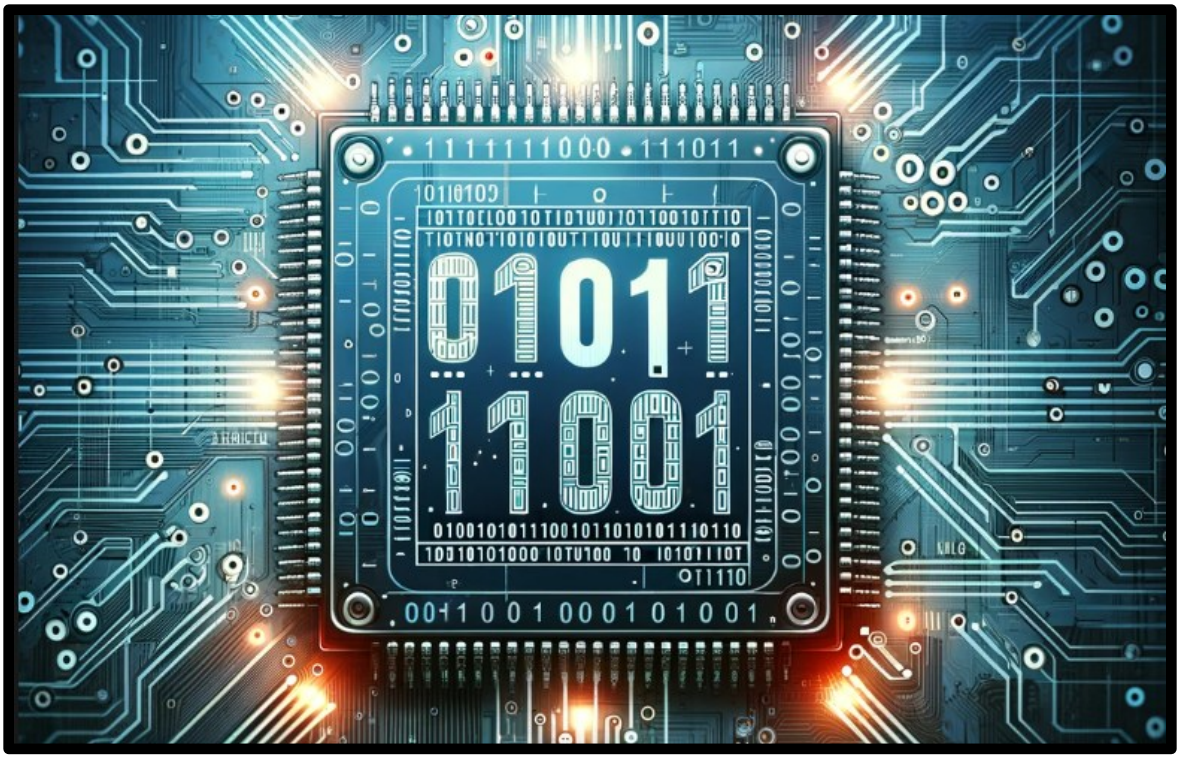
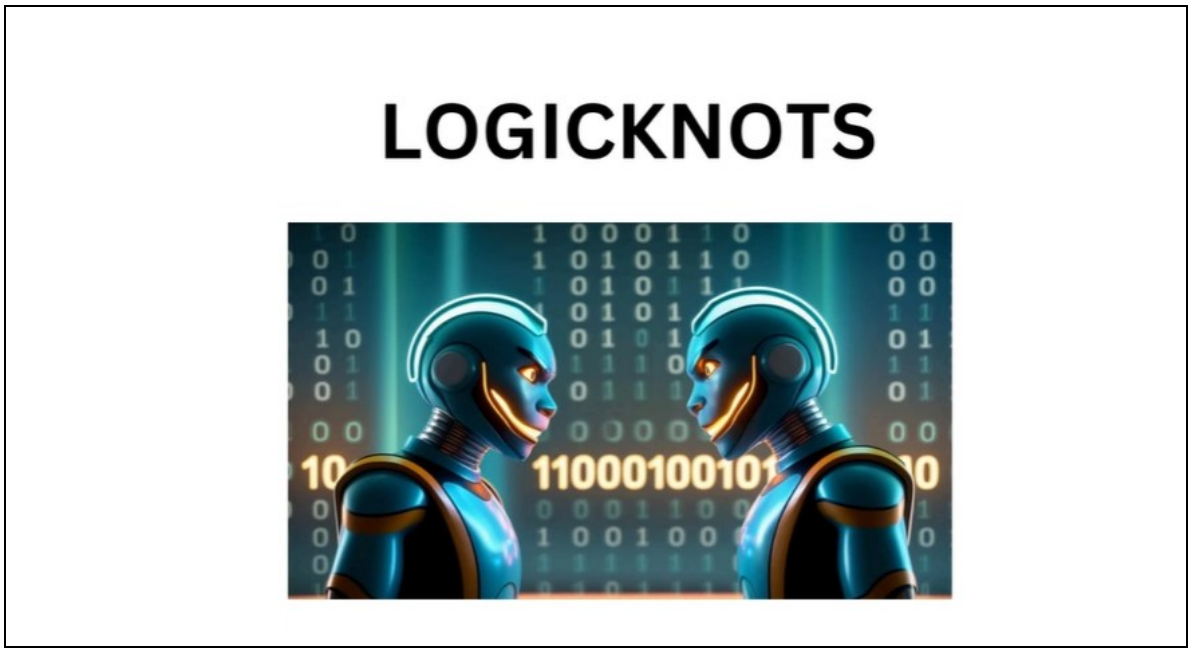


Binary Number System and Its Importance

Overview & Purpose

In this chapter, we will understand number systems, especially the binary number system, and how it is related to digital electronics.

Number System



On the screen, there's an English word written — Logicknots — which we can all read and understand. That's because we recognize the letters and their combinations.

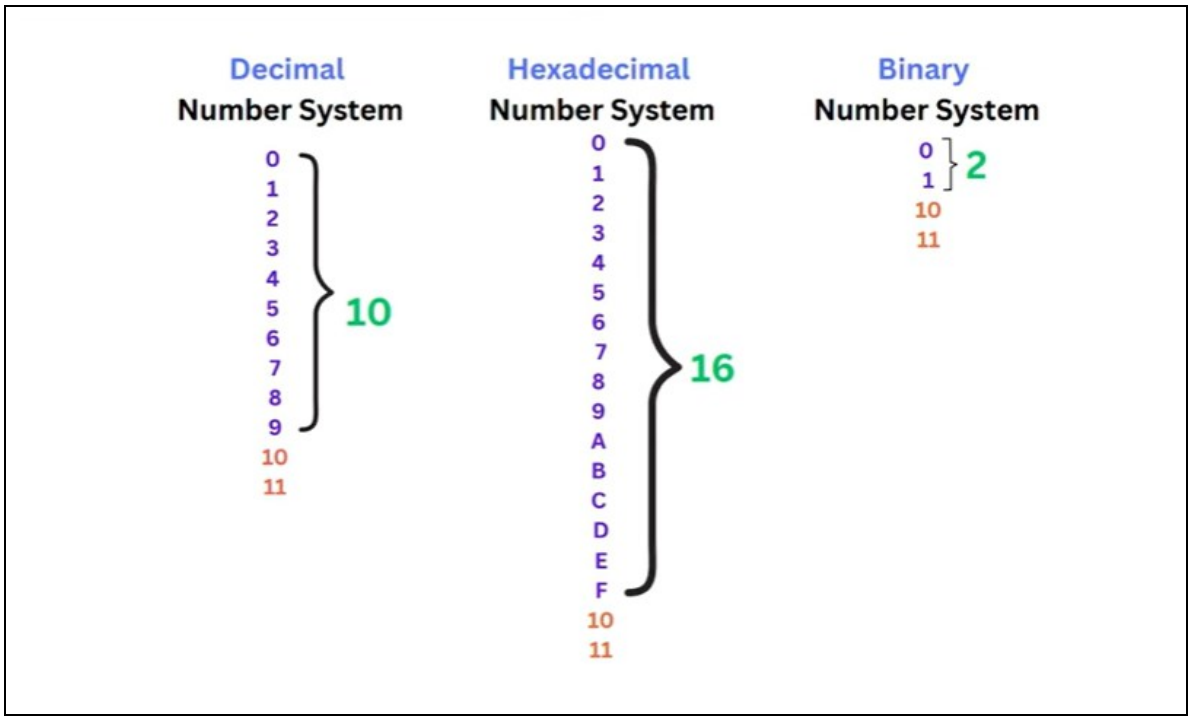
Now imagine that we don't know any letters and only understand two symbols — 0 and 1.

In such a situation, we will have to use only these two symbols (0 and 1) to communicate. If we want to create any word, we'll do it using just these two symbols.

If we have to say something, we'll have to use only 1 and 0. Any words we create will also be formed using 1 and 0 only.

Fortunately, that's not the case, and we have many words for communication which we can use to understand and talk.

But this is not the case with semiconductor microchips. Semiconductor microchips understand only two things — 0 and 1. So it's important to know this. And since there are only two symbols, the system is called a binary system.



🔢 Decimal Number System

On the left of the image, we see the Decimal Number System.

This is the number system we most commonly use in our daily lives. We are all very familiar with it.

In the decimal system, we start counting from 0:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Here we use a total of 10 different symbols (digits) — that's why it's called "decimal."

The word "decimal" comes from the Latin word 'Decem', which means ten.

Now think, when we count up to 9, we have no new symbols left. So what do we do?

We reuse the available digits but by adding a new digit.

So after 9, the next number becomes:

10 (which we call 'ten')

Then the counting continues: 11, 12, 13, ..., 19, 20, 21, ...

And this continues forward — until we reach 99.

Now when we reach 99, we've already used all the available digits.

So to go further, we add another new digit, and the next number becomes:

100 (which we call 'hundred')

Thus, in the decimal number system, we can create infinite numbers using only 10 symbols (0 to 9) — by adding new digits and placing them at different place values.

🔠 Hexadecimal Number System

Just like the decimal number system has 10 symbols, there is another number system called the Hexadecimal Number System.

This system has 16 symbols:

10 symbols from 0 to 9, and then A, B, C, D, E, F

(Here, A to F represent the values from 10 to 15 respectively.)

So, the complete set is: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F

This is why it is called "Hexa" (meaning 6) + "Decimal" (meaning 10) = Hexadecimal, a base-16 number system.

How is counting done in Hexadecimal?

Counting starts as: 0, 1, 2, ..., 9, A, B, C, D, E, F

Now we've used all 16 symbols. Since no new symbol is left, what do we do?

We add a new digit, just like we did in decimal after 9.

So after F, comes: 10 (which is equal to 16 in decimal)

Then the counting continues: 11, 12, 13, ..., 1F (where 1F equals 31 in decimal)

Then: 20, 21, ..., 2F, and so on.

⚙️ Binary Number System

Another number system is called the "Binary Number System."

In binary, we have only two symbols: 0 and 1

That’s why it’s called a base-2 number system.

How is counting done in Binary?

Counting starts: 0, 1

Now we’ve used all the available symbols (0 and 1). To move forward, we add another digit, just like we did in decimal and hexadecimal.

So the next number will be: 10 (which equals 2 in decimal)

Then: 11 (which is 3 in decimal)

Now again, the two symbols are used up. So we add a new digit again:

100 (in decimal, 4)

101 (decimal 5)

110 (decimal 6)

111 (decimal 7)

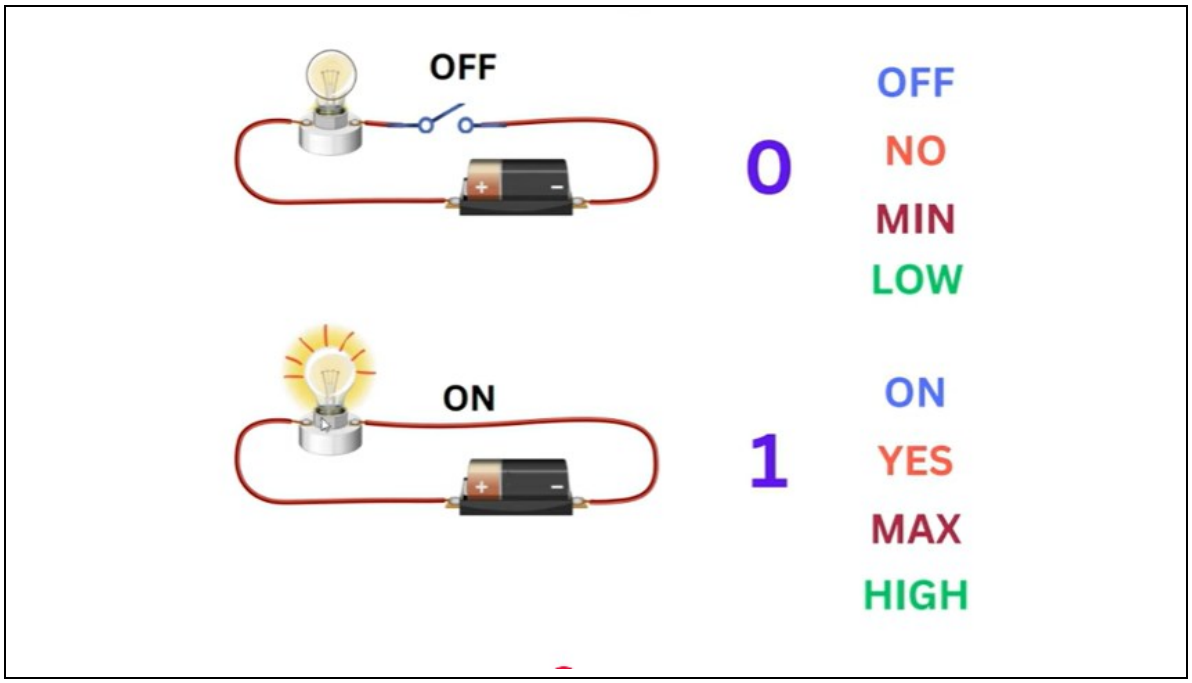
The binary system is the core language of computers, as computers understand only 0 and 1 — these are read as ON (1) and OFF (0).

💡 Importance of Binary in Digital Electronics

Let’s now focus on the binary number system because it is the most important for understanding digital electronics.

Digital electronics are directly related to semiconductor chip design. Within chips, all communication, signal transmission, and processing happens — all in the form of binary numbers.

Why Binary?



Because in electronics, it is very easy to create and detect only two states:

ON (meaning current is flowing, bulb is on)

OFF (meaning no current, bulb is off)

We can name them anything, like:

ON = High, 1, Yes, Max

OFF = Low, 0, No, Min

So the binary system (Binary = 2 states) is the easiest and most suitable for electronics. That's why computers, mobiles, and chip designs use binary numbers.

Power and Bits in Binary

Now an important question:

If I have n bits, how many different numbers can I make?

The formula is: Total Possibilities = 2^n

Examples:

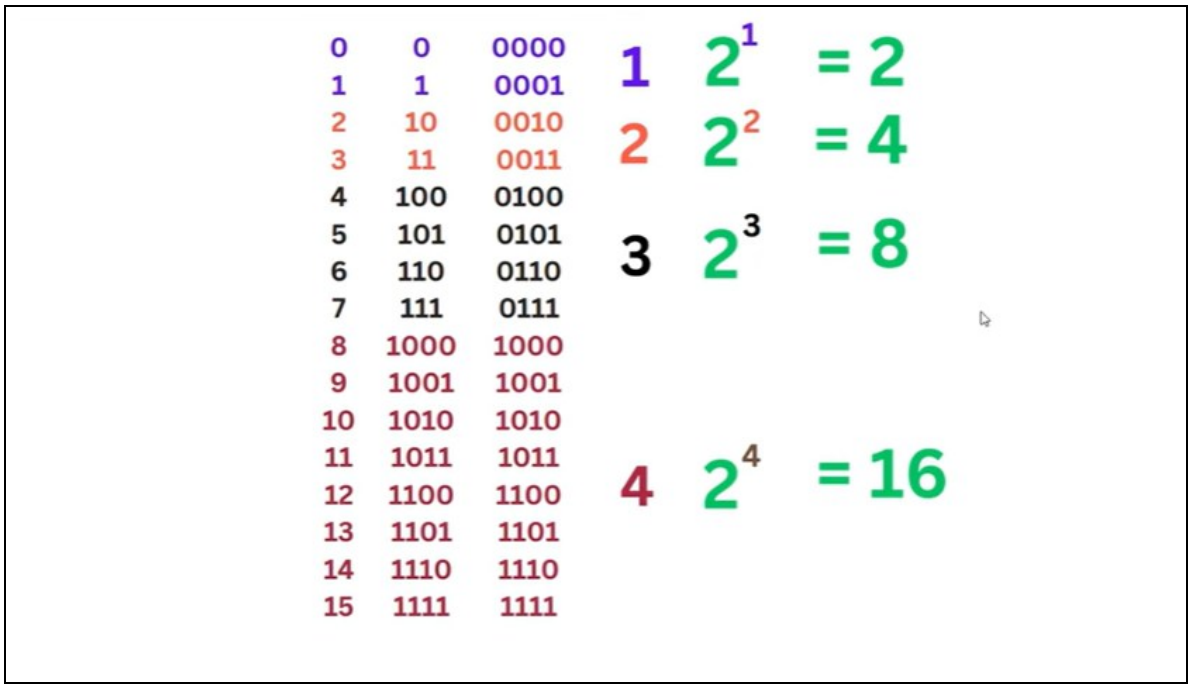
1 bit → 2^1 = 2 numbers: 0, 1

2 bits → 2^2 = 4 numbers: 00, 01, 10, 11

3 bits → 2^3 = 8 numbers

4 bits → 2^4 = 16 numbers

So if a number has 4 bits, we can make a total of 16 different binary numbers.



In binary, each "digit" is called a bit.

When several bits combine to form a number, we can call it a binary word.

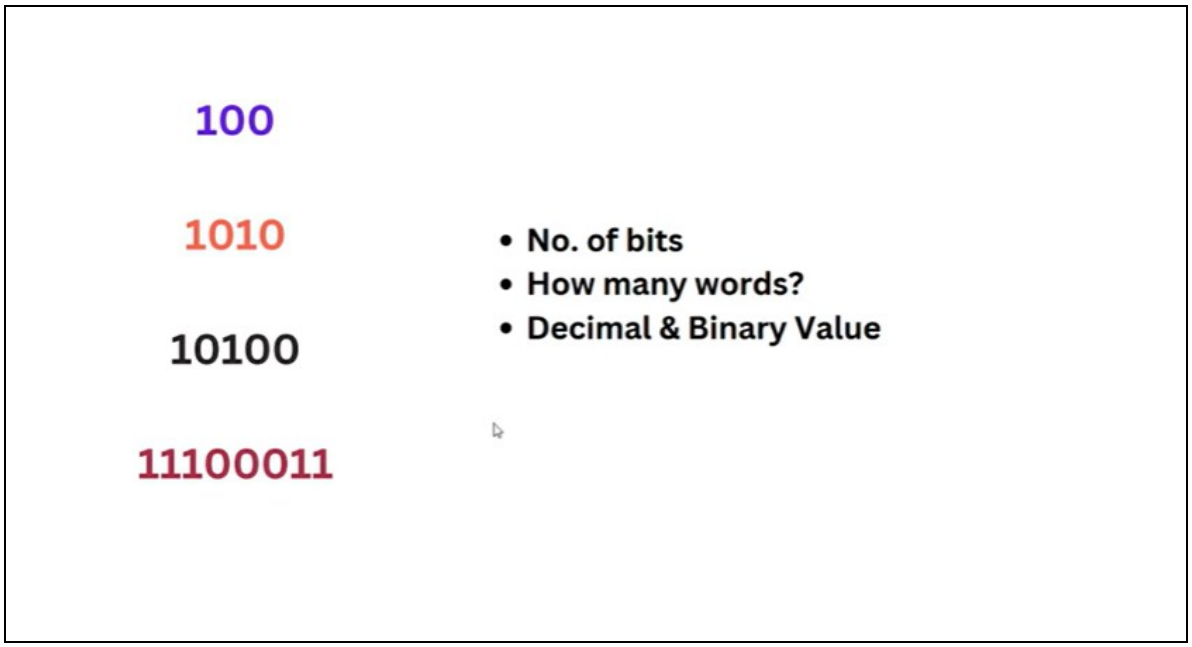
Examples:

Binary number 100 → 3 bits

Binary number 1010 → 4 bits

10100 → 5 bits

11110000 → 8 bits (which can be called an 8-bit word)



How to Convert Binary to Decimal?

Suppose we have a binary number: 1010

To convert it to decimal, we apply power of two position-wise from right to left:

$1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 8 + 0 + 2 + 0 = 10$

Similarly:

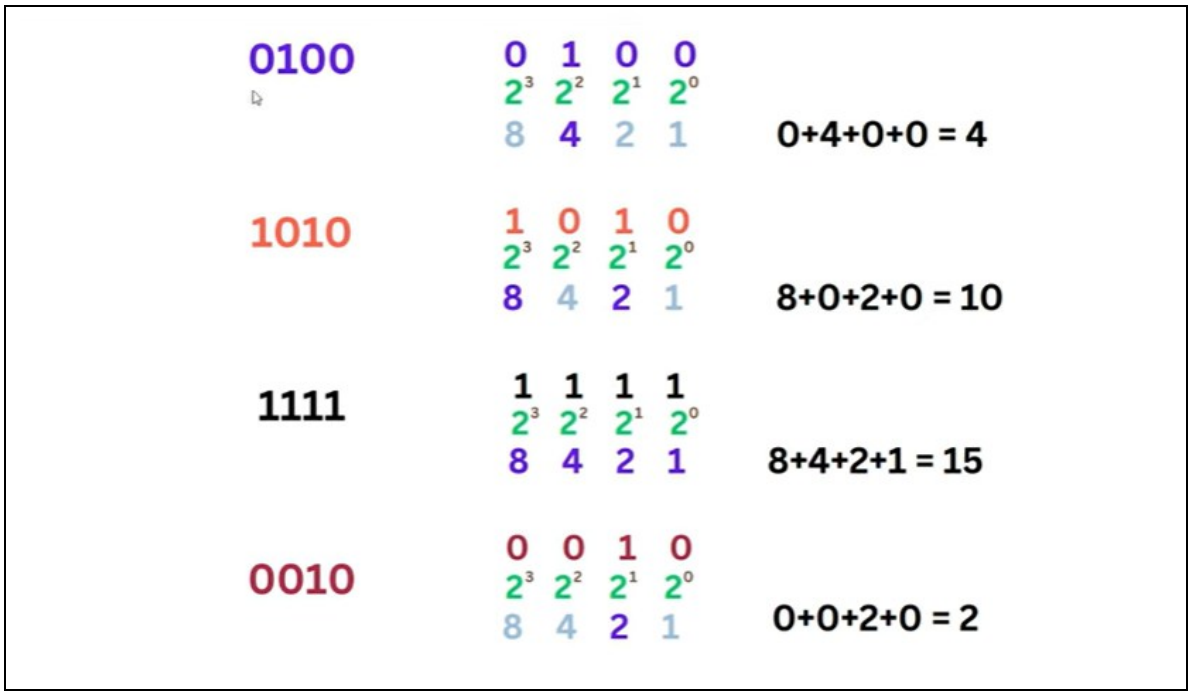
$1111 = 8 + 4 + 2 + 1 = 15$

$0100 = 0 + 4 + 0 + 0 = 4$

If we have 1 bit: $2^1 = 2$ possibilities

2 bits: $2^2 = 4$ possibilities (00, 01, 10, 11)

4 bits: $2^4 = 16$ possibilities

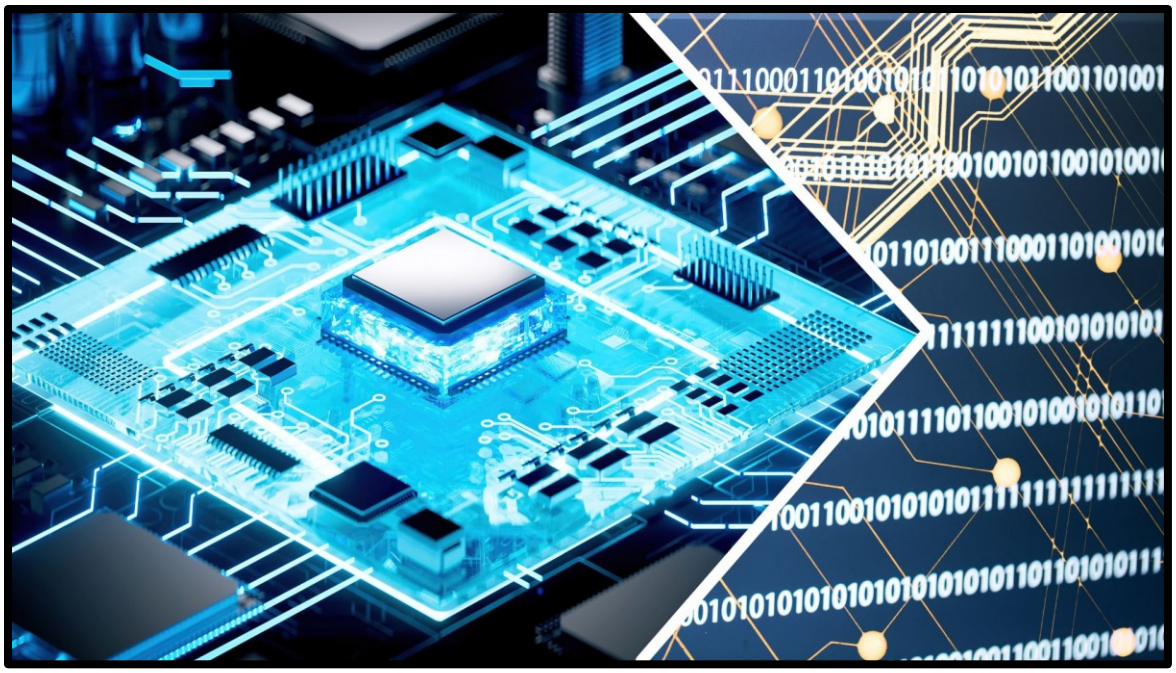


END Summary

The binary number system is the core foundation of digital electronics.

Digital electronics = semiconductor chips, and chips communicate in the language of binary numbers.

So, understanding binary is necessary to understand the digital system.



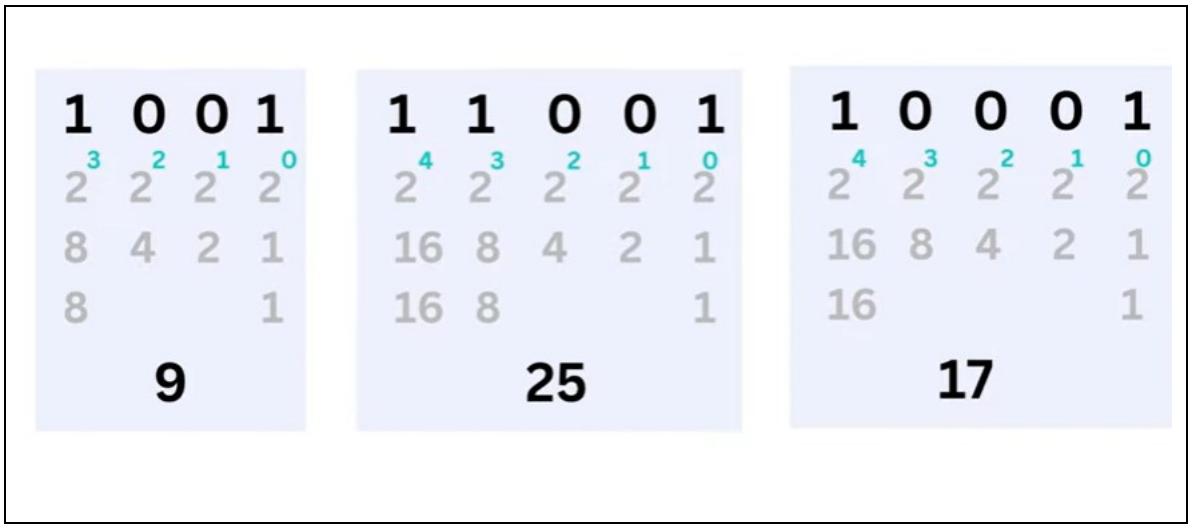
Binary Decimal Conversion

Overview & Purpose

In this chapter, we will talk about:

- Binary to Decimal Conversion
- Decimal to Binary Conversion
- How Binary and Decimal numbers are represented
- How Binary number addition and multiplication is done

🔢 Binary to Decimal Conversion



Suppose you have a binary number —

👉 1001

We need to find its equivalent in decimal.

Step-by-step method:

Write powers of 2 under each digit, starting from the right:

Binary: 1 0 0 1
Powers: 3 2 1 0
Values: 8 4 2 1

Now wherever the binary digit is 1, take the value at that position. Ignore where it's 0. In 1001, only positions 3 and 0 have 1 → i.e., take 8 and 1.

Now add these values:
8 + 1 = 9

So, the decimal equivalent of binary 1001 is 9

👉 11001

Binary: 1 1 0 0 1
Powers: 4 3 2 1 0
Values: 16 8 4 2 1

Take only the values at positions where binary has 1 → 16, 8, and 1

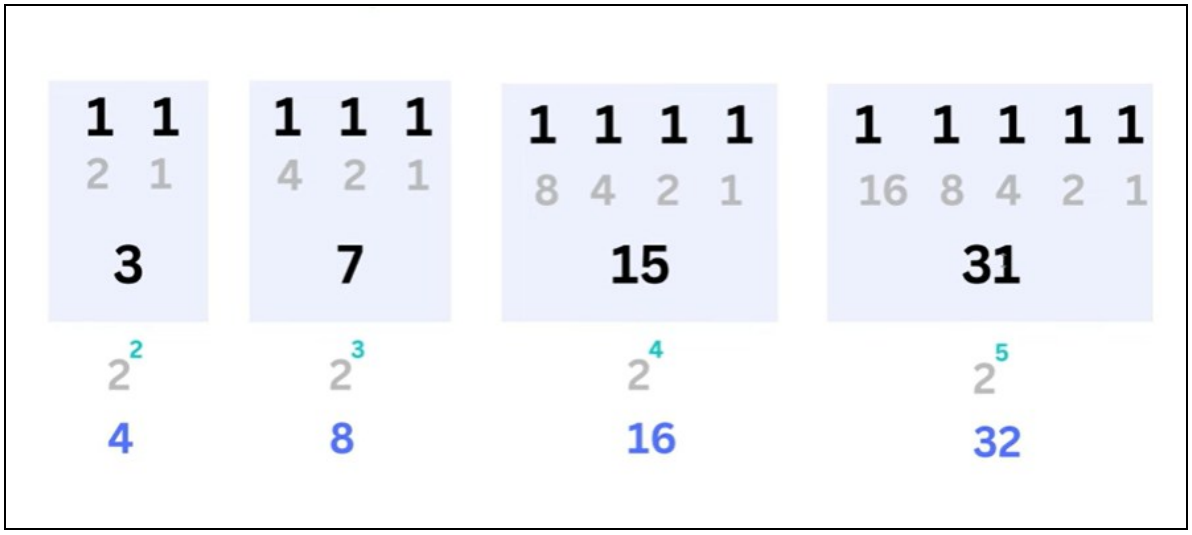
Add: 16 + 8 + 1 = 25 → Decimal equivalent of 11001 is 25

👉 10001

Binary: 1 0 0 0 1
Powers: 4 3 2 1 0
Values: 16 8 4 2 1

Only 16 and 1 are taken (rest are zero) → 16 + 1 = 17 → Decimal is 17

📊 Binary Number Capacity (Bit Length vs Count)



This image shows an important concept — the relationship between binary and decimal number counting capacity.

For example:

11 (2-bit binary number) → decimal equivalent is 3
So with 2 bits, we can count from 0 to 3 → $2^2 = 4$ values

111 (3 bits) → count from 0 to 7 → $2^3 = 8$

1111 (4 bits) → count from 0 to 15 → $2^4 = 16$

11111 (5 bits) → 0 to 31 → $2^5 = 32$

If we want to convert a decimal number to binary, it's important to know the minimum number of bits required to represent it.

Example:

To represent decimal number 40 in binary — 5 bits can only go up to 31. But 40 is greater, so we need to add one more bit.

Thus, we need at least 6 bits for 40 because:

$2^6 = 64 \rightarrow$ i.e., we can represent decimal numbers from 0 to 63 with 6 bits.

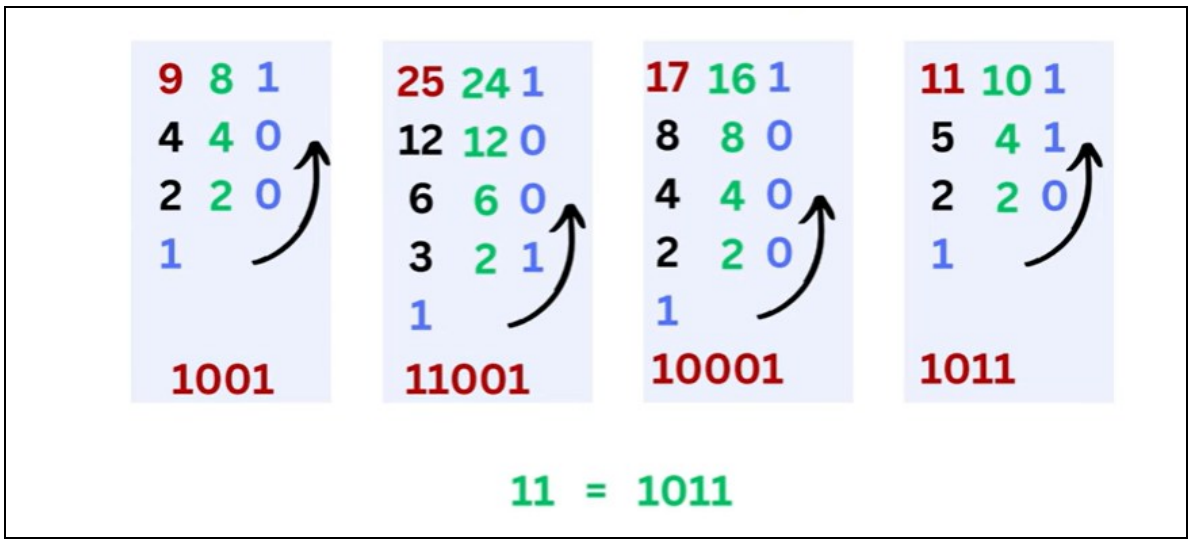
The more bits we have in binary, the more decimal numbers we can count. To represent larger numbers, we need more bits.

Examples:

2 bits → $2^2 = 4$ numbers (0 to 3)
3 bits → $2^3 = 8$ numbers (0 to 7)
4 bits → $2^4 = 16$ numbers (0 to 15)

So if you want to represent decimal number 40, 5 bits won't work (max is 32). You'll need 6 bits since $2^6 = 64$.

🔢 Decimal to Binary Conversion



Here we learn a very simple and easy way to convert a decimal number to binary. This method just needs two things — the table of 2 and division.

Suppose we have a decimal number — 9

Now we convert it to binary:

Steps:

9 ÷ 2 = 4 remainder 1
4 ÷ 2 = 2 remainder 0
2 ÷ 2 = 1 remainder 0
1 ÷ 2 = 0 remainder 1

We stop when quotient becomes 0.

Now read the remainders from bottom to top: 1 0 0 1
So, binary of 9 is — 1001

Another example — 25:

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

Now read from bottom to top: 1 1 0 0 1 → Binary of 25 is 11001

We had converted 11001 to decimal earlier, and it was 25. That means this method is correct and reliable.

Similarly, converting 17 gives 10001, and 11 gives 1011

So we can write:

11 = 1011

Now from just looking, one can't tell which is decimal and which is binary. Because "11" is a valid decimal number. But in binary, "11" is also valid (equal to decimal 3).

Likewise, "1011" could be 1011 in decimal or 11 in binary.

So it's difficult to identify which number belongs to which number system.

🔍 Solving Confusion Between Decimal and Binary

Every number system is clarified by writing its base:

Decimal numbers are represented as base 10: $(11)_{10}$
Binary numbers are represented as base 2: $(1011)_{2}$
Octal numbers are represented as base 8

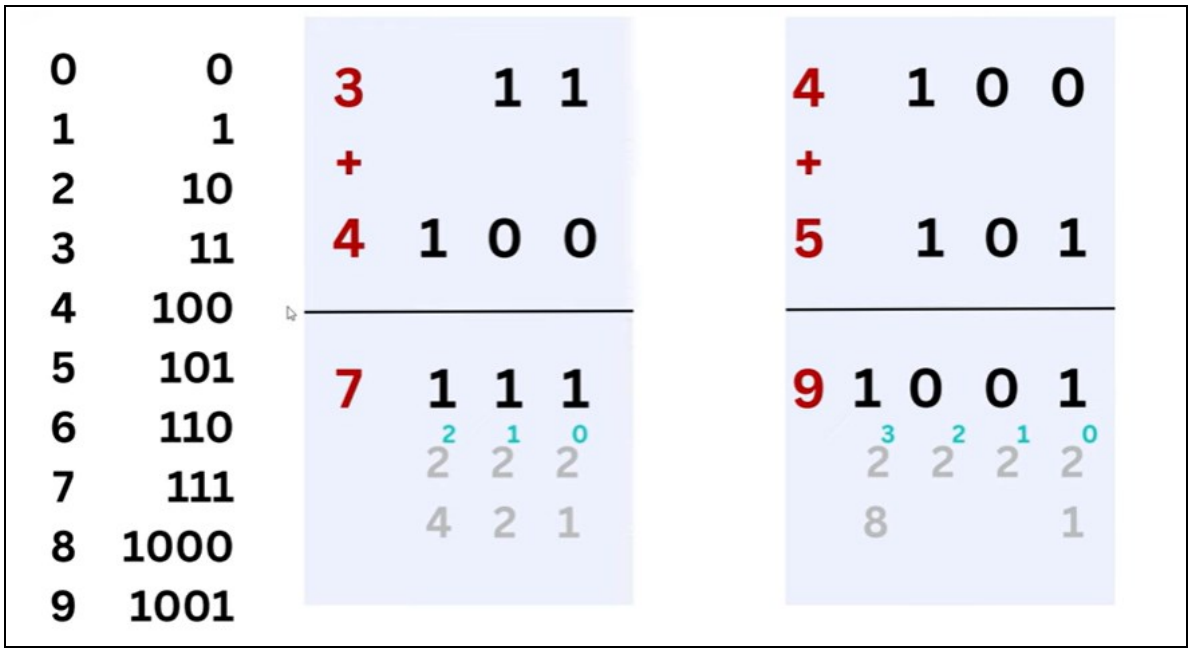
So if we write:

$(11)_{10} = (1011)_2$

Then anyone can understand that 11 is decimal and its equivalent binary number is 1011

So always remember, when comparing binary and decimal numbers, **always write the base** to avoid confusion.

➕ Binary Number Addition



Now let's understand how to add binary numbers.

Binary addition is very easy because it only has two digits - 0 and 1. That means there are fewer possible combinations. Binary has only four basic additions:

0 + 0 = 0
0 + 1 = 1
1 + 0 = 1
1 + 1 = 10 (i.e., 0 with a carry of 1)

Using these four rules, we can easily add any two binary numbers.

Example 1: 3 + 4 = 7

In binary:

3 = 011
4 = 100

Addition:

011
+ 100
———————
111 (Decimal = 7 → so it's correct)

Example 2: 4 + 5 = 9

Binary:

4 = 100
5 = 101

Addition:

100
+ 101
———————
1001 (Decimal = 9 → again correct)

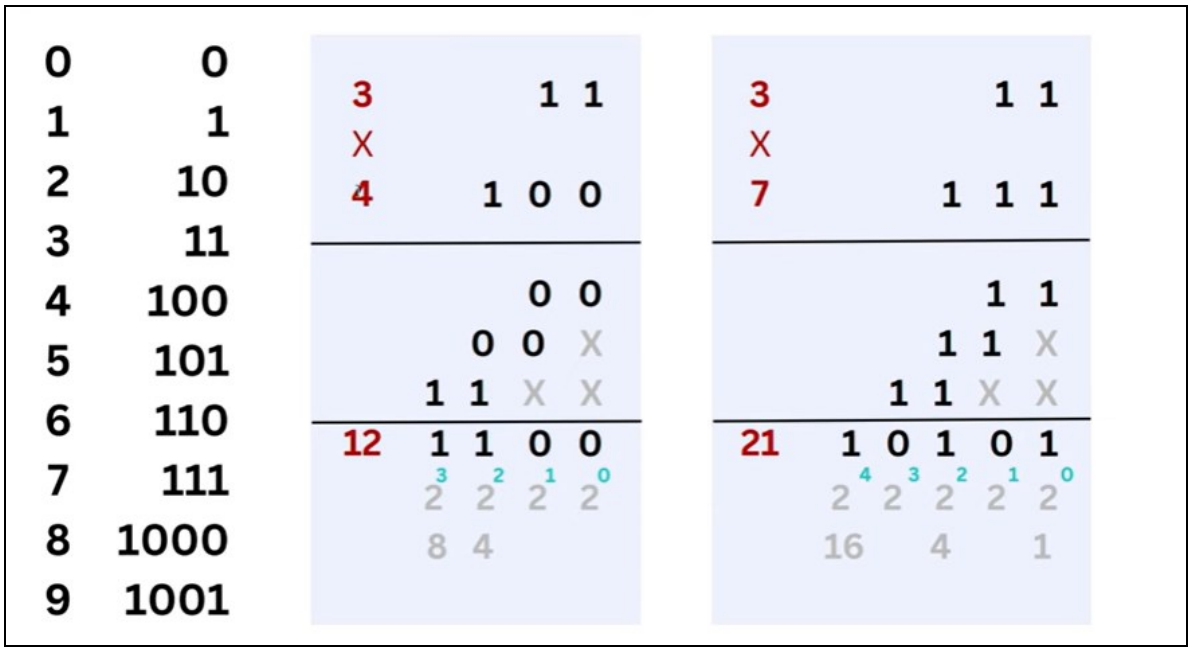
Note:

When we add 1 + 1 in binary, the result is 0 and a carry is generated — just like in decimal when 9 + 8 = 17 (we write 7 and carry 1).

In binary: 1 + 1 = 0 (write 0, carry 1)

If we need to add the carry to the next digit, we handle it carefully.

✖ Binary Number Multiplication



Now we will learn how to multiply binary numbers. The process is very simple — just like in decimal, except here we only use 0 and 1.

🧠 Remember a simple rule:

0 × anything = 0
1 × anything = the same number

Example 1: 3 × 4 = 12

In binary:

3 = 011
4 = 100

Multiplication:

```
011   (3)
  × 100   (4)
  ——————————
    000   ← multiplied by 0
   000X   ← again by 0
  011XX   ← multiplied by 1 (shifted 2 places)
  ——————————
  01100  = 12 (in decimal)
```

Example 2: 3 × 7 = 21

In binary:

3 = 011
7 = 111

Multiplication:

```
011   (3)
   × 111   (7)
  ———————————
    011     ← by 1
   011X     ← by 1 (shifted 1 place)
  011XX     ← by 1 (shifted 2 places)
  ———————————
  10101     = 21 (in decimal)
```

Semiconductor Chip Specs and Logic Gates

Overview & Purpose

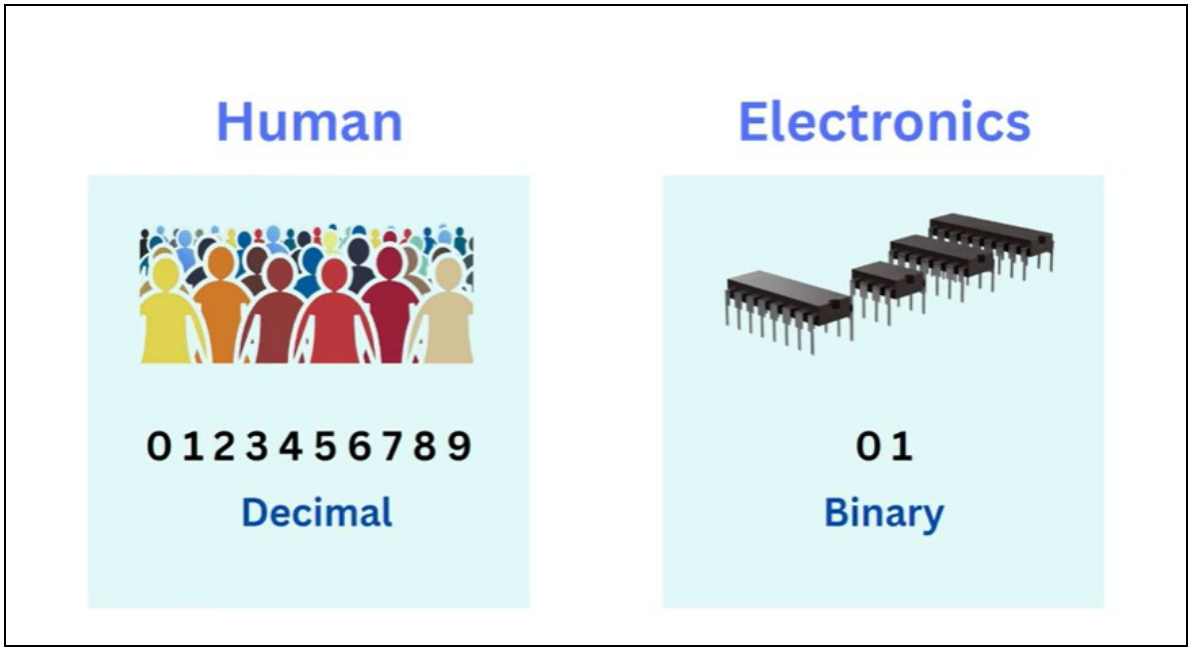
In the previous two chapters, we learned about:

- Decimal Numbers and Binary Numbers
- How Binary Numbers are used in electronics and why they are so important

Now, in this chapter, we will go a step further and understand how binary numbers are actually applied and how they work inside a semiconductor chip.

Just like we use decimal numbers (0 to 9) in our daily lives for counting, calculations, and communication, the world of electronics — especially in devices like chips, processors, and semiconductors — depends entirely on binary numbers. Binary, which consists of only two digits, 0 and 1, forms the very foundation of how electronic systems operate, store data, and perform tasks. Understanding how these binary numbers are processed inside a chip will help

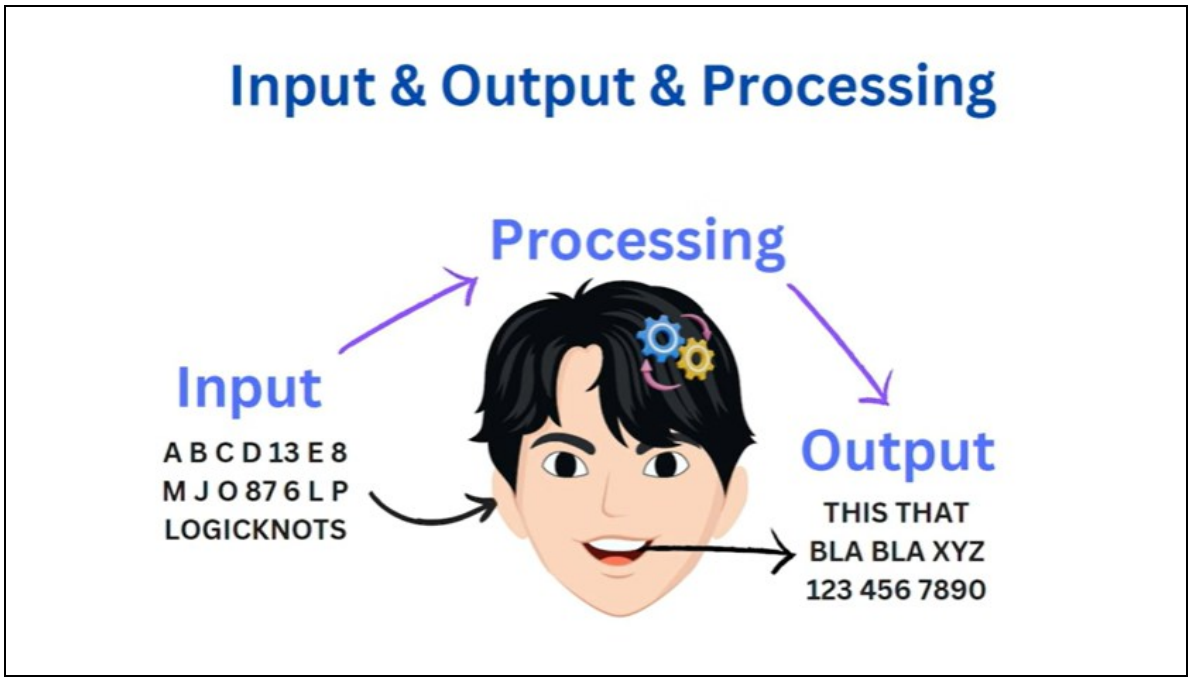
you see how all modern electronic devices work, from simple calculators to advanced computers.



Input-Processing-Output: A Simple Cycle

Let's now take a deeper look at how Input and Output work in both humans and machines.

The diagram below shows how humans use numbers, alphabets, or other forms of information as input and then produce output by following a simple three-step cycle:



Input:
We receive input through our sense organs — mainly through our ears (by listening) or eyes (by seeing or reading). Anything we hear, see, or read is taken in as information by our brain.

Processing:
Once the information is received, our brain starts working on it. It thinks, analyzes, or calculates what needs to be done with that information. This is called processing.

Output:
After processing the input, we respond by performing an action — like speaking, writing, moving, or reacting in some way. This is the output produced by our brain and body.

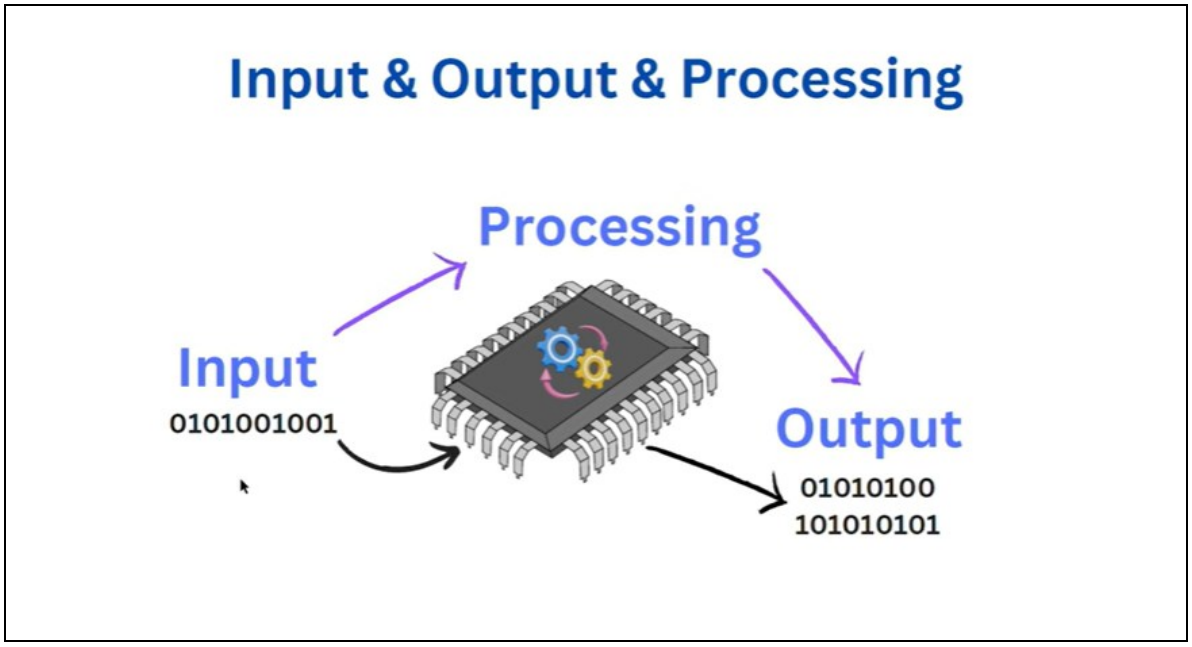
This forms a simple and continuous cycle:
Input → Process → Output

Example:
When someone asks you, “What is 2 + 2?”, you:

1. Hear the question (Input)
2. Think about the answer (Process)
3. Reply “4” (Output)

Just like humans follow this cycle to communicate and respond, chips and semiconductor devices follow the same fundamental steps to perform tasks — with one key difference: instead of using language or emotions, they use binary numbers (0s and 1s) to handle input, processing, and output.

How This Works in Semiconductor Chips



Semiconductor ICs (Integrated Circuits) also follow the same basic principle as humans: they take input, process that input, and provide output.

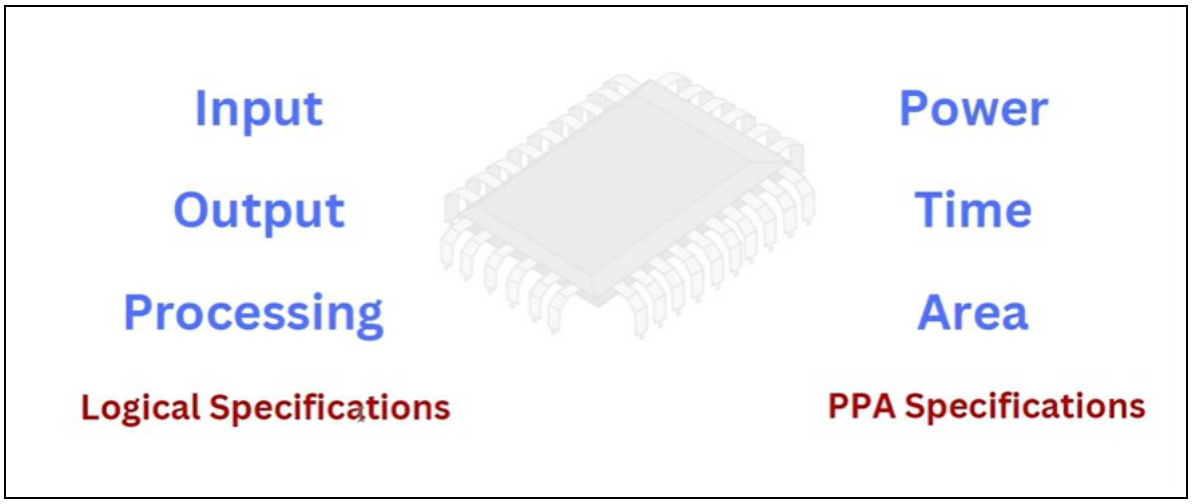
The main difference lies in the language of understanding:
Humans work with alphabets, words, sentences, and decimal numbers (0 to 9). That's how we read, speak, and think.
But computers and semiconductor chips can only understand binary numbers — 0s and 1s. This is the digital language in which all their operations happen.

So, when it comes to computing devices like chips:

1. Processing always begins with **binary input** — a stream of 0s and 1s representing the information.
2. The chip executes logical operations using this input to make decisions or calculations.
3. Finally, the chip generates a **binary output**, which can later be converted into forms that humans can understand, like text, sound, or images.

These three steps — Input → Processing → Output — together form the Logical Specification of the chip or electronic system.

⚙ Semiconductor Chip Design Parameters



🔁 Logical Specifications

Input → Process → Output

This entire process is called logical because it follows a well-defined set of rules. It's like asking a question: "If the Input is X, then what should the Output be?" The clear connection between input and output, based on logic, is what we refer to as Logical Specification.

In summary, Logical Specification is a blueprint that tells the semiconductor device how it should behave when specific input is given. Everything that happens inside a computer or chip follows this predictable, logical pattern.

👤 Who designs the chip?
A client or customer — such as a company, organization, or individual — places an order for the chip based on their specific needs or application. The client clearly defines two important things:

- What kind of Inputs will be provided to the chip
- What kind of Outputs should be produced in response to those Inputs

Example:

"If I give input 100, the output should be 11001."
This specific requirement is called the Input-Output Specification. It acts like an instruction sheet that tells what the chip should do with each input.

👷 What does the designer do?

Once the client provides the Input-Output Specification, the chip designer comes into action. The designer's job is to create the processing logic inside the chip so that when a particular input is given, the output matches exactly what the client expects.

In simple terms, the designer must figure out:

- What kind of processing should take place inside the chip?
- How should the internal circuits and logic gates be arranged to get the correct output every time?

This carefully planned processing system is known as the Logical Design of the chip.

So in summary:

- The client defines what should happen.
- The designer decides how it should happen inside the chip using logical circuits and binary processing.

This collaboration between the client's requirement and the designer's technical knowledge forms the foundation of chip design

⚡ Power, Performance, Area (PPA) Specifications

When designing a chip, it's not enough to just focus on the logic and how the input is converted into the output. Three more very important factors must be considered to make the chip useful and efficient in the real world:

🔋 1. Power

- Just like humans need food and energy to function, chips need electrical power to work.
- We carefully measure how much power the chip will consume while performing its tasks.
- Lower power consumption is better, especially for devices like smartphones and laptops that run on batteries.

⏱️ 2. Performance (or Delay/Time)

- Performance refers to how quickly the chip can produce the output after receiving the input.
- This time taken is also called delay.

- Faster response = Better performance.
- For example, when you press a key on your phone, you expect it to respond immediately — that's possible because of good chip performance.

📐 3. Area

- Area means how much physical space the chip occupies on the silicon wafer.
- Smaller chips are preferred because they save space, reduce material costs, and can fit into compact electronic devices.
- For example, smaller chips are used in smartwatches or wireless earphones.

🔍 PPA: A Perfect Balance

These three factors together are called PPA Specifications:
Power, Performance, and Area.
When a client orders a chip, they don't just give Input-Output requirements — they also specify:

1. How much power should be consumed
2. How quickly the output should be produced
3. How compact the chip should be

The chip designer must keep all of these requirements in mind while creating the Logical Design of the chip. It's a balancing act — improving one factor often affects the others, so finding the right balance is key.

✅ Conclusion:

A good chip is one that:

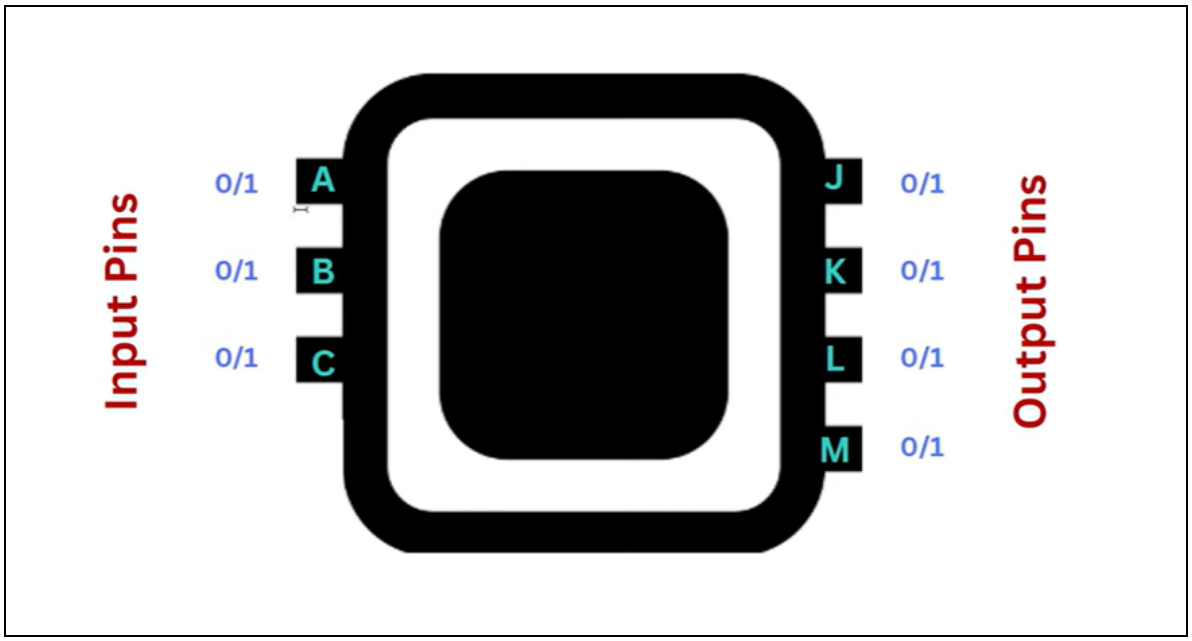
1. Gives the correct output (Accuracy)
2. Produces output quickly (High Performance)
3. Consumes as little power as possible (Low Power)
4. Takes up minimal space (Compact Area)

Achieving the right balance of PPA ensures the chip works efficiently in any electronic device, whether it's a simple calculator or an advanced smartphone

Input and Output Pins in Chip Design

When designing a digital chip, one of the first things to decide is the number of Input and Output pins the chip will have. These pins are the points where external devices can send signals into the chip (Input) or receive signals from the chip (Output). Each of these pins works with binary values, meaning they can only carry either a 0 (representing OFF/Low) or a 1 (representing ON/High).

Let's take a simple example to understand this better.



Suppose we have 3 Input pins named A, B, and C.
Since each pin can only be 0 or 1, we can create multiple unique combinations by varying these 0s and 1s across the three pins.

For 1 pin, there are 2 combinations → 0 or 1

For 2 pins, → $2^2 = 4$ combinations

For 3 pins, → $2^3 = 8$ combinations

So with 3 Input pins (A, B, C), we can create 8 unique binary input combinations.

These combinations are:

000 001 010 011 100 101 110 111

This means the chip can receive 8 different possible input cases.

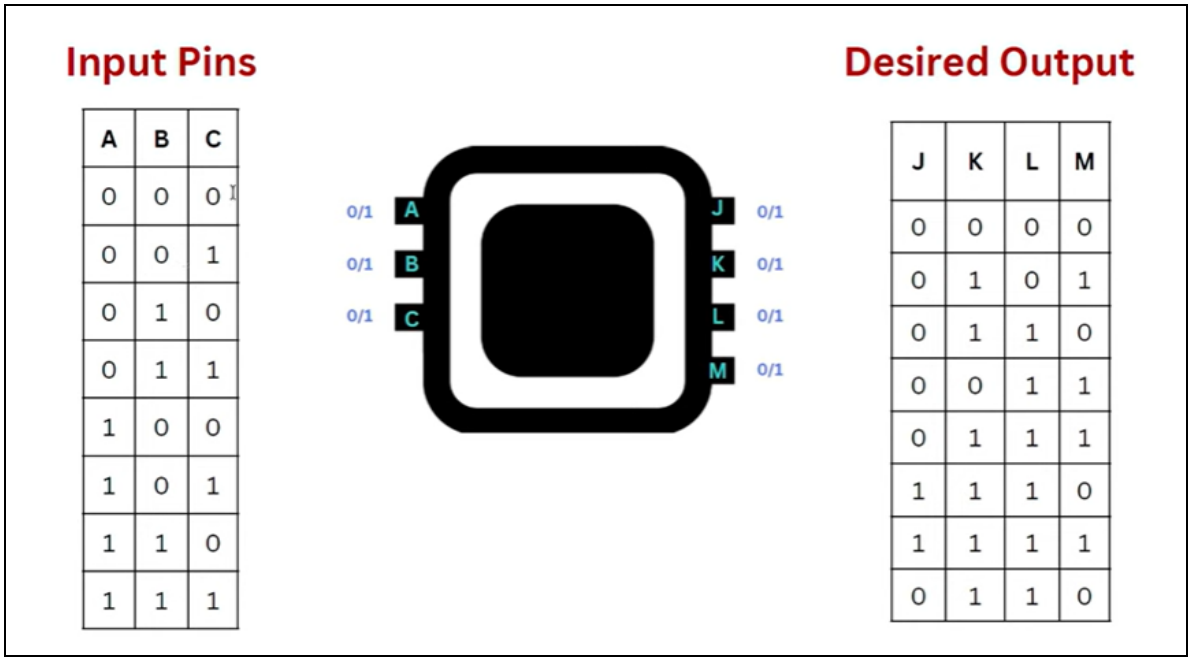
Now let's talk about Output pins. Suppose we have 4 Output pins named J, K, L, and M.
For each input combination, we can decide what values should be present on each Output pin. These outputs, too, will be binary — combinations of 0s and 1s.

Example:

If A=0, B=0, C=0 → Output could be: J=0, K=0, L=0, M=0 → Which is 0000

If A=0, B=0, C=1 → Output could be: J=0, K=1, L=0, M=1 → Which is 0101

This way, for every one of the 8 possible input combinations, the output pins can be set to any binary value, depending on the client's requirement.



The number of Input and Output pins is not fixed — it completely depends on what the client or system designer wants.

More Input pins mean more possible combinations, making the system more complex.

More Output pins allow for more information or actions to be conveyed from the chip to the external world.

Each Input combination will have an Output associated with it, and this entire mapping is decided by the client depending on the specific task the chip has to perform.

This entire Input-Output mapping is implemented as logic inside the chip. The way these outputs are generated from the given inputs is designed using logic gates like AND, OR, NOT, NAND, NOR, XOR, etc.

The process of building this entire mapping with the required logic is called **Digital Circuit Design** or **Logical Design**.
This design tells what exact output will be produced for every possible input, making the chip function exactly as desired.

The best part is — this design is fully customizable.

If the client needs different output behavior, the logical design can be changed to match.

If the chip is being built for a specific machine or application (like a washing machine, a remote control, or a calculator), the entire logic is built specifically for that use.

When a chip is designed specifically to meet unique requirements like this, it is known as a **special-purpose chip**. These chips are different from general-purpose processors because they are **tailor-made for a particular job**, resulting in faster and more efficient performance for that task.

🔢 Standard Logic Gates

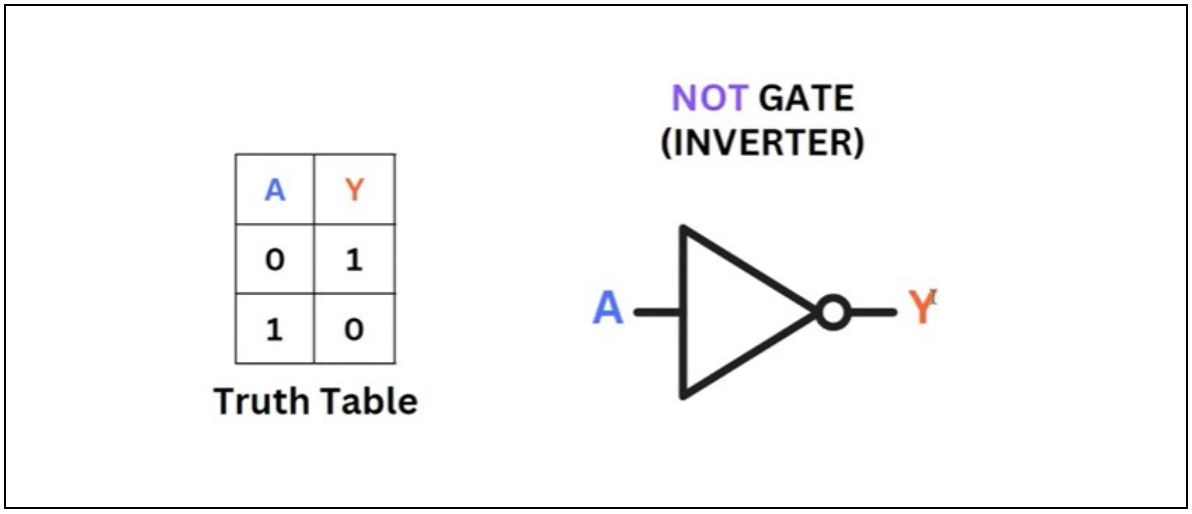
In digital electronics, there are some widely used, pre-designed chips known as Standard Logic Gates. These are called “standard” because their Input-Output behavior is fixed and well-defined. No matter where or how they’re used, they always perform the same logical operation. These gates are the basic building blocks of all digital circuits, including computers, calculators, smartphones, and more.

Let’s go over them one by one with more detail:

1. NOT Gate (Inverter)

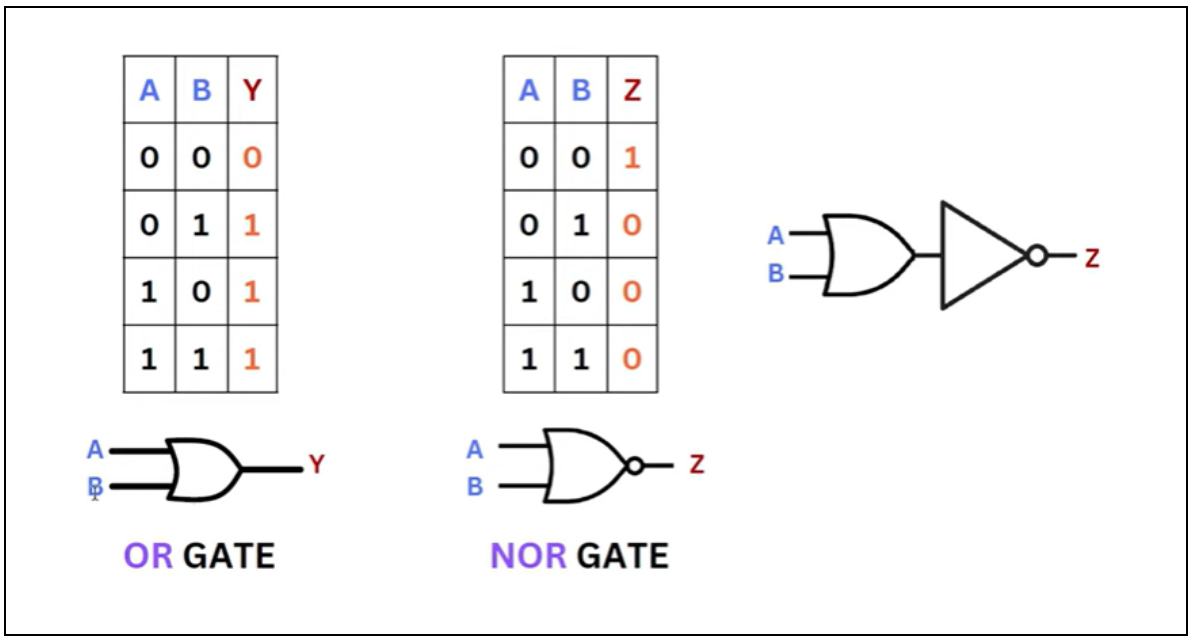
- Number of Inputs: 1

- This is the simplest logic gate. It reverses the input — if you give it 0, it outputs 1; if you give it 1, it outputs 0.
- Because it produces the opposite of the input, it is also called an Inverter.
- Example: Input = 1 → Output = 0



2. OR Gate

- Number of Inputs: Usually 2 (but can have more)
- The output is 1 if either of the inputs is 1. If both are 0, then the output is 0.
- Think of it like: "If A OR B is true, output is true."
- Example: A = 0, B = 1 → Output = 1

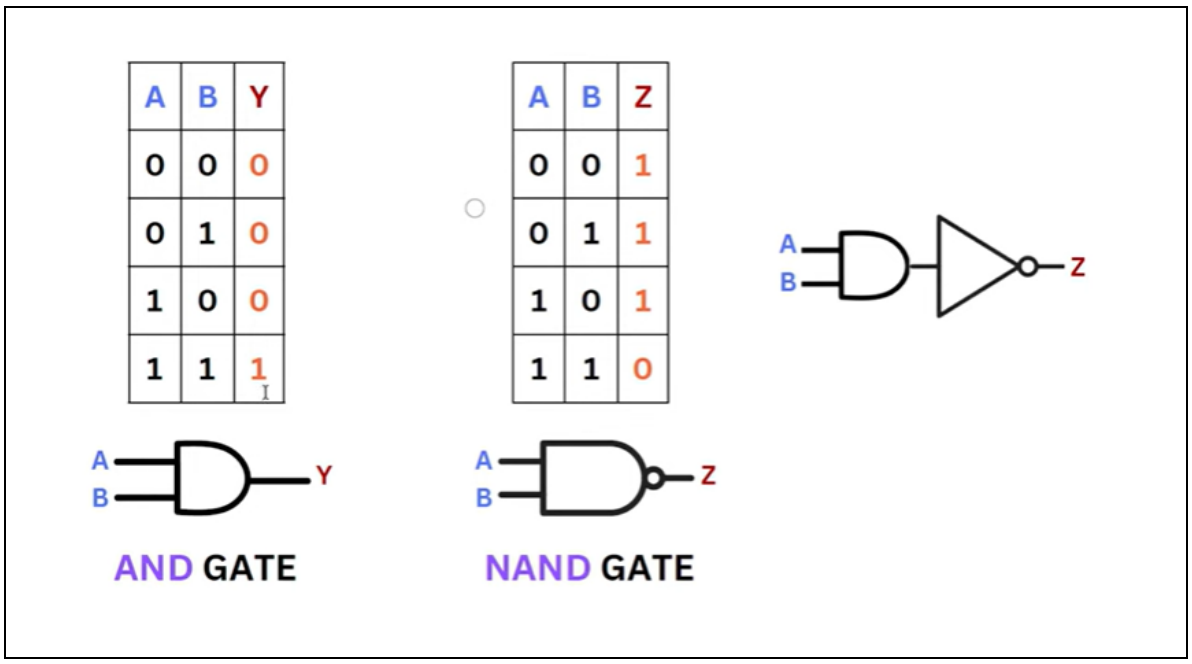


3. NOR Gate (OR + NOT)

- This is the inverted version of the OR Gate.
- You can build a NOR Gate by connecting a NOT Gate to the output of an OR Gate.
- The output is 1 only if both inputs are 0.
- Example: A = 0, B = 0 → Output = 1

4. AND Gate

- Number of Inputs: Usually 2 (but can have more)
- The output is 1 only when both inputs are 1. If either input is 0, the output will be 0.
- Example: A = 1, B = 1 → Output = 1

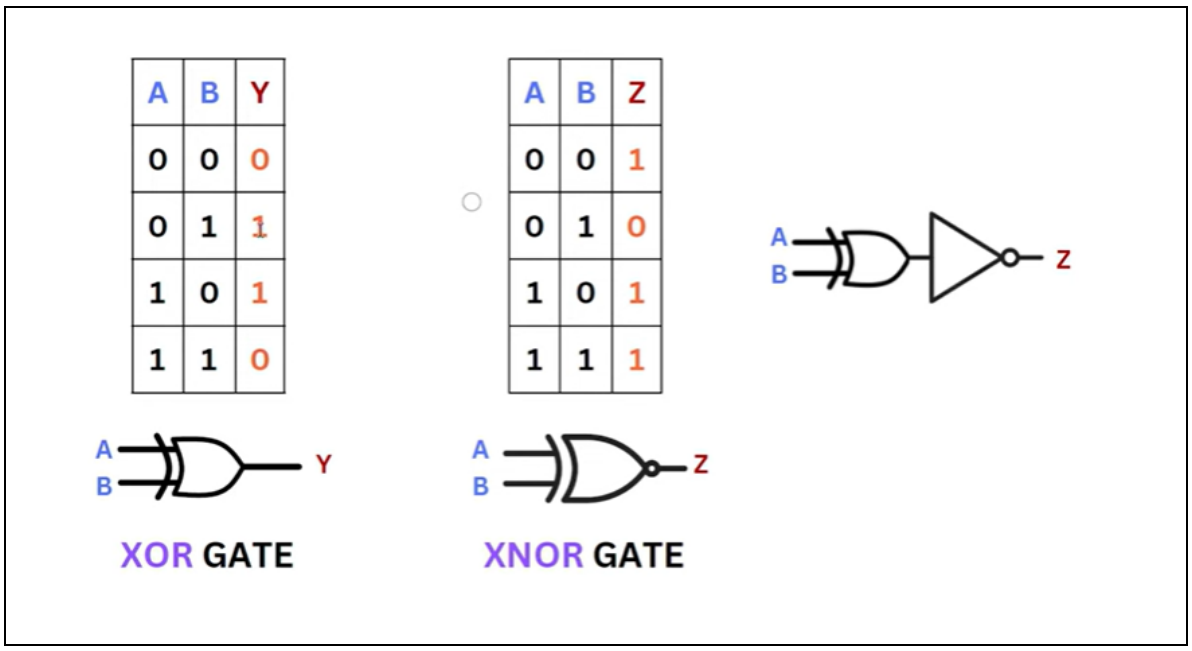


5. NAND Gate (AND + NOT)

- The NAND Gate is the inverted version of the AND Gate.
- It is formed by adding a NOT Gate at the output of an AND Gate.
- Whenever the AND Gate would produce 1, the NAND Gate produces 0 instead.
- Example: A = 1, B = 1 → Output = 0

6. XOR Gate (Exclusive OR)

- The output is 1 only when the two inputs are different.
- If the inputs are the same (both 0 or both 1), the output is 0.
- Example: A = 1, B = 0 → Output = 1
- Common Use: XOR gates are often used in adders and error detection circuits.

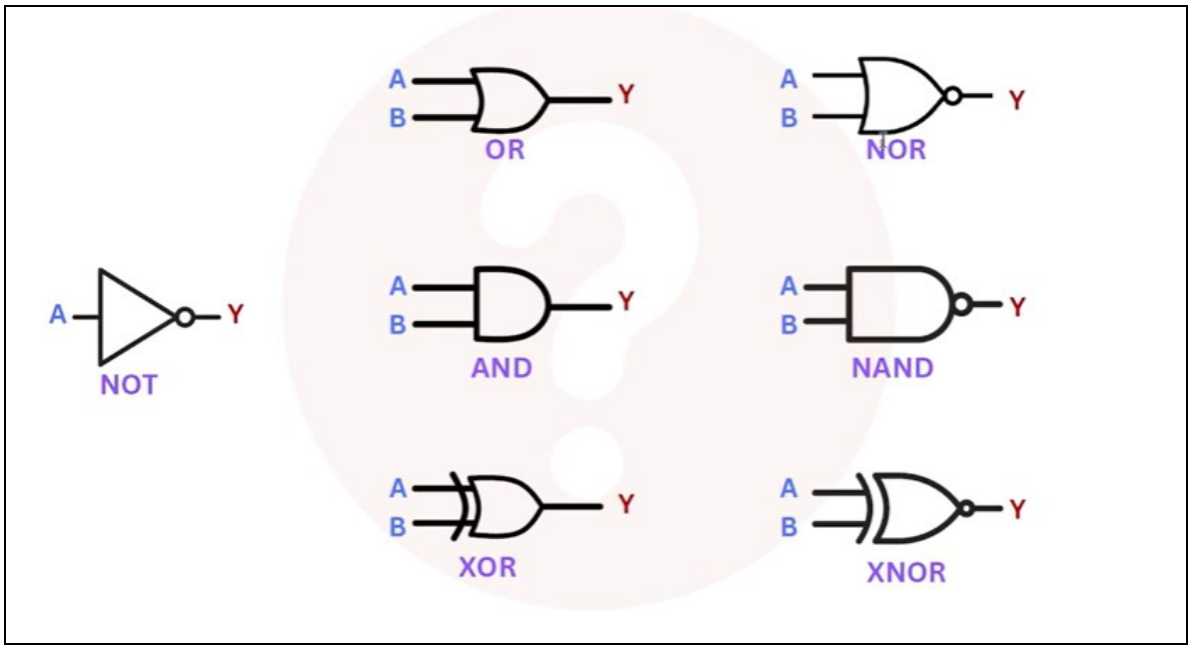


7. XNOR Gate (Exclusive NOR)

- This is the inverted version of the XOR Gate.
- The output is 1 only when both inputs are the same (00 or 11).
- Example: A = 1, B = 1 → Output = 1

🔄 Relationships Between Gates

By adding a NOT Gate to the output of other gates, we can easily create their inverted versions:



OR → NOR

AND → NAND

XOR → XNOR

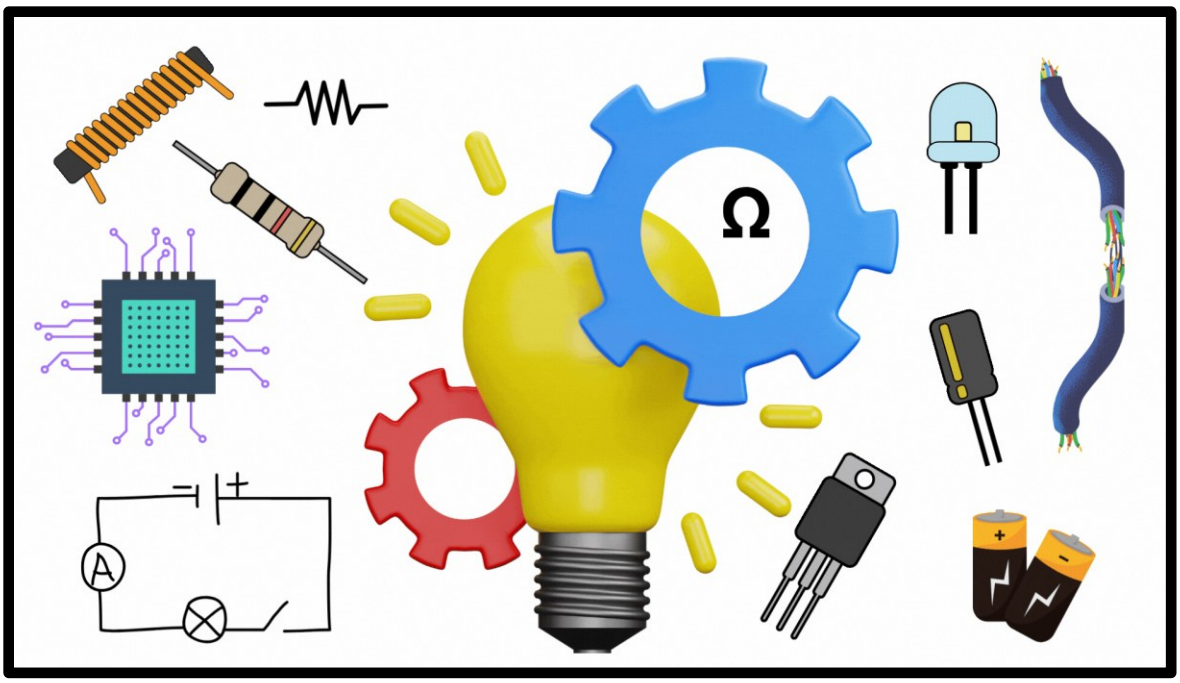
This relationship shows how NOT Gates play an important role in building more complex circuits from simple ones.

By now, we have learned their symbols, behavior, and truth tables, which tell us exactly what output to expect for every input combination.

In the next chapter, we will dive deeper and explore how these logic gates are actually built inside a chip —

What circuits are used inside them?

How do transistors work together to create these logical operations?
That's where the connection between electronics and digital logic becomes even more interesting.



Foundation of Electronics for Chip Design

Overview & Purpose

From this chapter onward, we will officially begin our journey into Chip Design and Circuit Design — the core of how modern electronic devices are built. But before we can dive deep into designing chips, circuits, and complex systems, there's something very important we must do first.

Before learning advanced concepts, we need to build a strong foundation by understanding the basic fundamentals of electronics. Without these basics, it would be difficult to fully grasp the logic behind how chips work, how circuits behave, or how signals flow through electronic components.

That's exactly what we are going to focus on in this chapter.
We will learn the essential Basics of Electronics and Circuit Design, which are the building blocks for everything that comes next.

In this part, we will clarify several important basic concepts of electronics, such as:

- What is current and voltage?
- How do electronic components like resistors and capacitors work?
- How do we create circuits using these components?

These fundamentals are very important because they will help you clearly understand how digital circuits are designed inside semiconductor chips, how those circuits process binary data (0s and 1s), and how everything fits together to build complete electronic systems.

Think of this chapter as the foundation of a building — the stronger the foundation, the stronger and taller the building can be.
With this understanding in place, you'll be fully prepared for the more advanced chip and circuit design topics that are coming up next.

❁ Chip Design Specifications

When we talk about designing a chip, there is always a client or customer who gives instructions about what kind of chip they need. Chip designers don't decide on their own what a chip should do — it always starts with a requirement from the client.

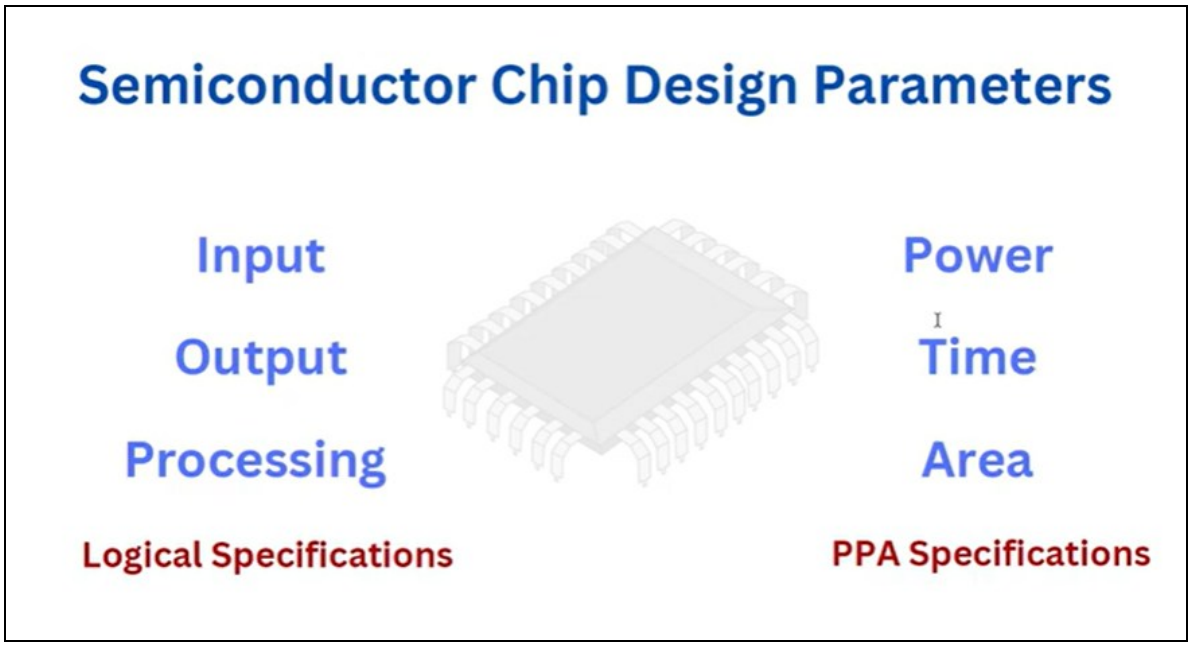
👉 The client explains:
"Design a chip for me that should have:

- Some input pins — into which binary numbers (0s and 1s) will be sent. These are like the entry points for information.
- Some output pins — from which the result or output will be received, again in the form of 0s and 1s."

👉 Along with this, the client also provides clear instructions about what Output should come for which Input.
For example:

- If input is 000 → Output should be 1011
- If input is 111 → Output should be 0100

All these input-output instructions together are called Logical Specifications. These specifications define exactly how the chip should behave — what result should be produced for each possible input combination.



But designing a chip is not just about connecting input to output.
While designing, engineers have to focus on three important factors to make sure the chip works efficiently:

1. Power

    - How much electrical energy (power) the chip uses while working.
    - Lower power consumption is better, especially for mobile devices like smartphones, where battery life matters.

2. Performance (Time/Speed)

    - How much time the chip takes to provide the output after getting the input.
    - Faster response means better performance, especially in high-speed processors.

3. Area

    - How much physical space the chip takes on the silicon wafer.
    - Smaller size means we can fit more chips on a single wafer, which reduces cost and improves efficiency.

Together, these three key aspects are known as PPA – Power, Performance, Area.
A good chip design tries to balance all three —

- Low Power
- High Performance (fast output)
- Compact Area (small size)

So, as you can now see, there are two types of specifications in chip design:

1. Logical Specifications —
   What output should be generated for each input combination (behavior of the chip)
2. Physical Specifications (PPA) —

    - How much power will the chip consume
    - How quickly will it give the result
    - How much physical space will it occupy

Now the question is — how is all this possible? How can we know how much time it will take or how much power will be required?

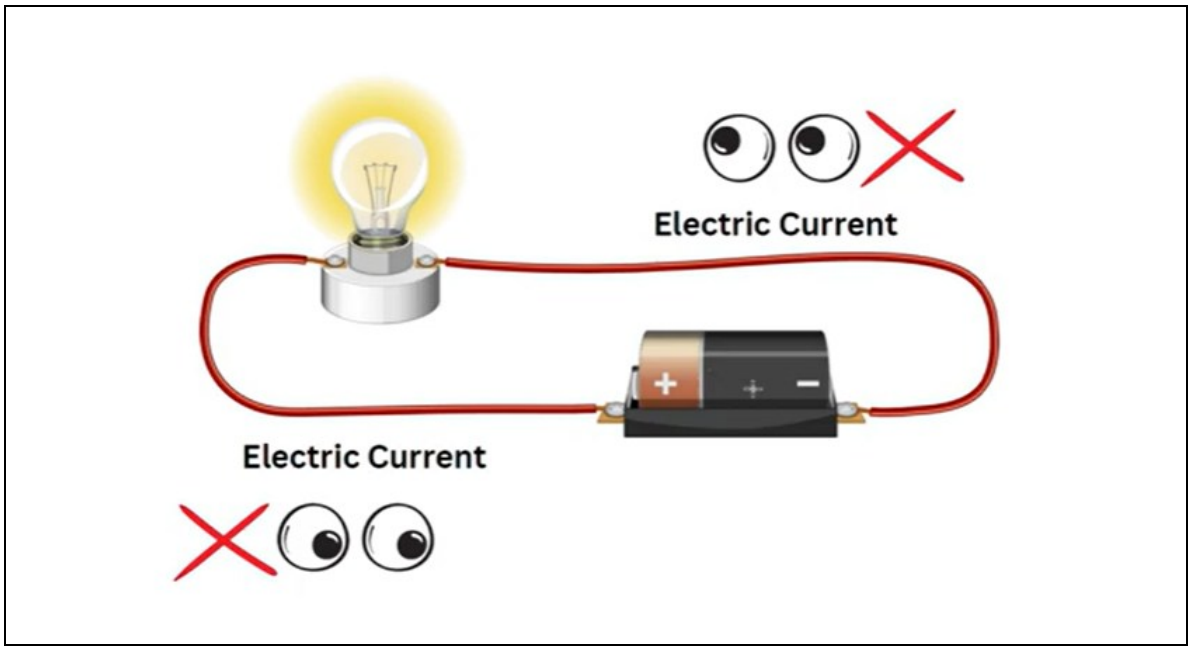
For that, we need to understand the basic principles of electronics.
Without understanding these concepts, we can't move forward in chip design.

That's why, in this chapter, we will begin with one of the most fundamental concepts of electronics — Electric Current, because this is what actually moves binary signals (0 and 1) in inputs and outputs.

Let's now explore and understand what Electric Current is, how it flows in circuits, and what role it plays in making a chip function properly.

Electric Current

Electric current can sometimes feel a bit tricky or confusing to understand, especially for beginners. One of the main reasons for this confusion is that when we create or observe a circuit — for example, connecting a battery, a bulb, and some wires — we can see the bulb light up, but we cannot see the electric current actually moving inside the wires.



It’s just like watching a magic show — you see the result, but you don’t fully understand what’s happening behind the scenes. The current is invisible, and that makes it harder to imagine what’s really going on or in which direction it is flowing.

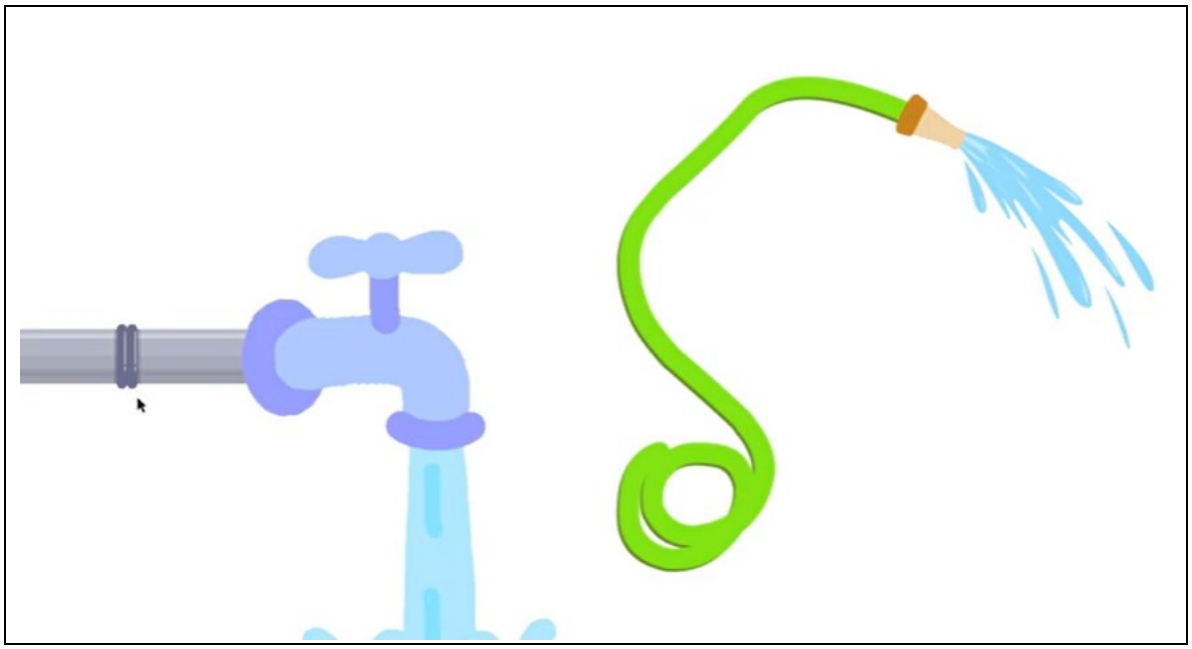
Since we cannot directly see the current moving, understanding how it behaves and flows in a circuit can feel slightly tricky or abstract. This is why many students find the concept of electric current a little difficult in the beginning.

But don’t worry — in this chapter, we will break it down using very simple, real-life comparisons and practical examples. Wherever possible, we’ll compare electric current to things you already know, like water flowing through pipes to make the explanation clear and relatable.

The goal here is just one — that you clearly understand what electric current actually is and how it works.

Water Flow Analogy

To understand electric current better, let’s take a very simple and relatable example — the flow of water.



We've all seen how water flows through a pipe. When we open a tap, water starts moving in a particular direction inside the pipe. The best part is that we can actually see the water flowing, which makes it very easy to understand how the flow works and how pressure or blockage can affect it.

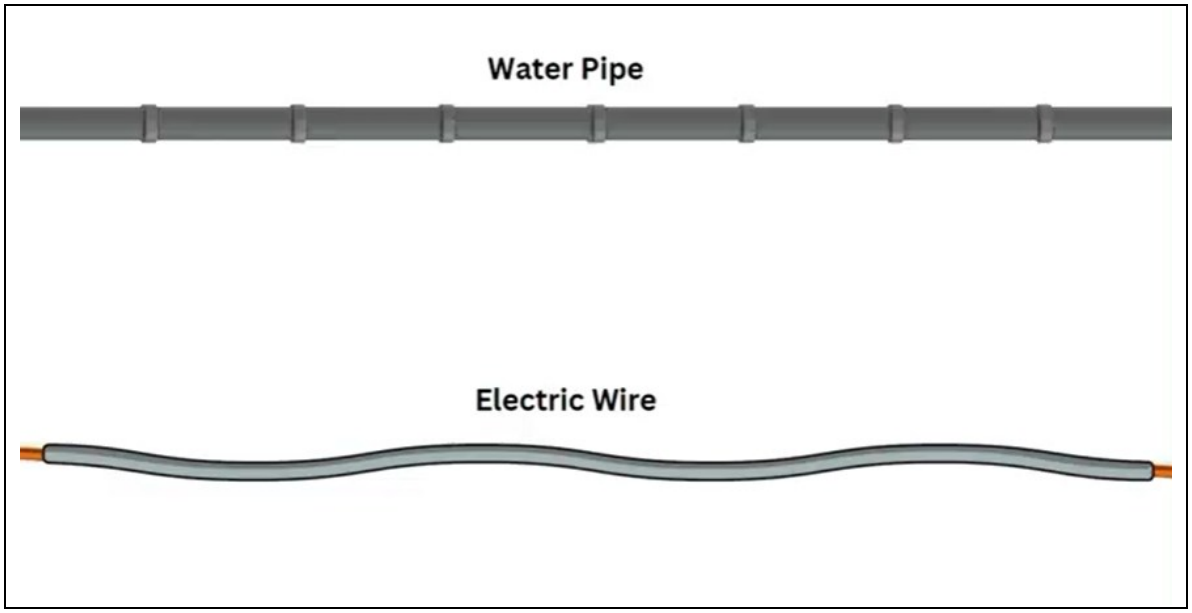
In the same way, electric current is also a kind of flow — but instead of water, it's the flow of tiny particles called electrons. The only challenge is that we can't see electrons moving with our eyes. This invisibility is what makes understanding electric current slightly difficult at first.

That's why we'll use this water flow analogy (comparison) to make things clearer. By comparing the flow of electric current to the flow of water, we can easily understand important concepts like:

- How current flows in a wire
- What role a battery plays in the circuit
- How energy gets transferred from one place to another inside a circuit

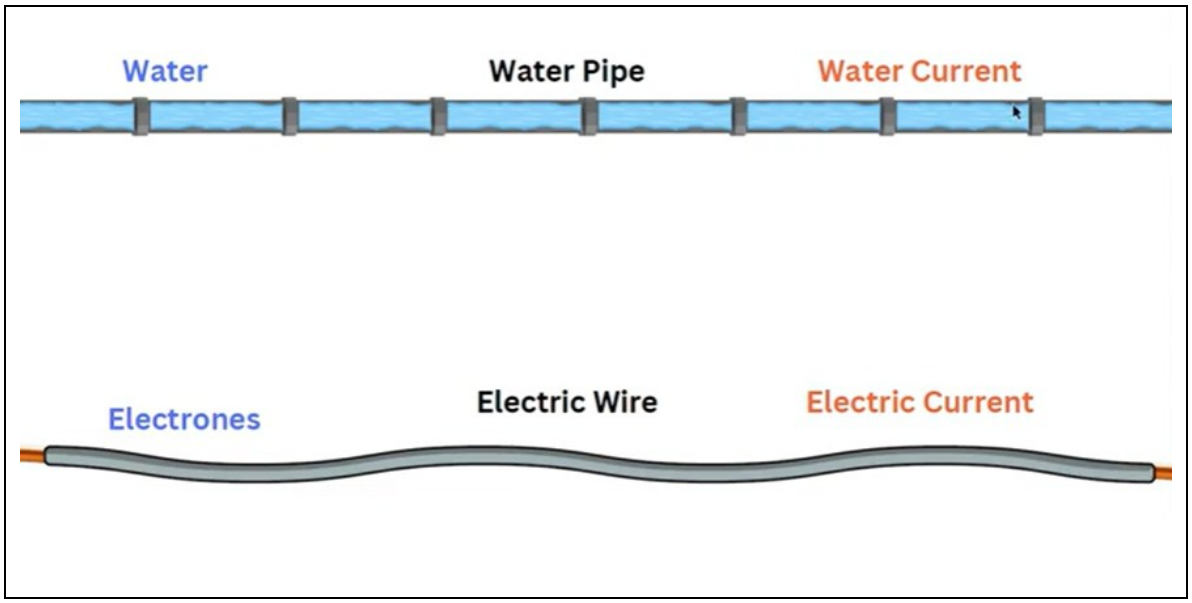
With this simple and familiar comparison, the idea of electric current will start to feel much easier and more understandable.

Water Current vs Electric Current



Imagine there's a pipe with no water inside it — at that moment, nothing is moving, and there's no flow happening. In this situation, we say that there's no water current at all.

Now, let's compare this to electricity. If we have an electric wire but no flow of electricity through it, then we can simply say that the electric current is zero. Just like a dry, empty pipe has no water flowing through it, an unused wire has no electrons moving, meaning no electric current.

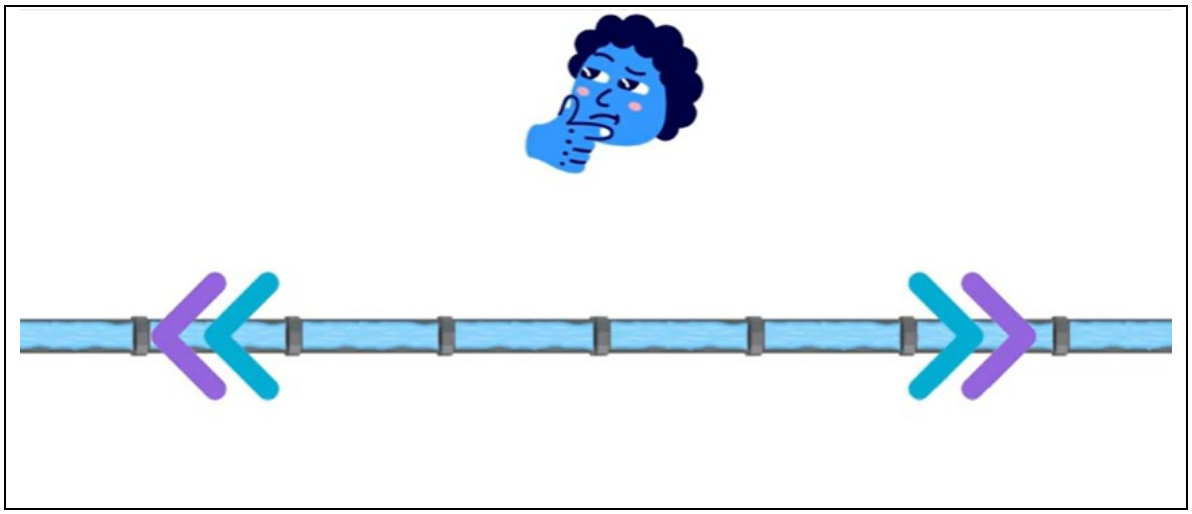


Now, imagine that we start filling the pipe with water, and the water begins to flow in a particular direction. As soon as that happens, we can say that there is water current inside the pipe.

Similarly, when electrons start moving inside a wire in one direction, we call that movement electric current. These tiny moving particles (electrons) carry energy from one place to another, just like flowing water carries force or pressure through pipes.

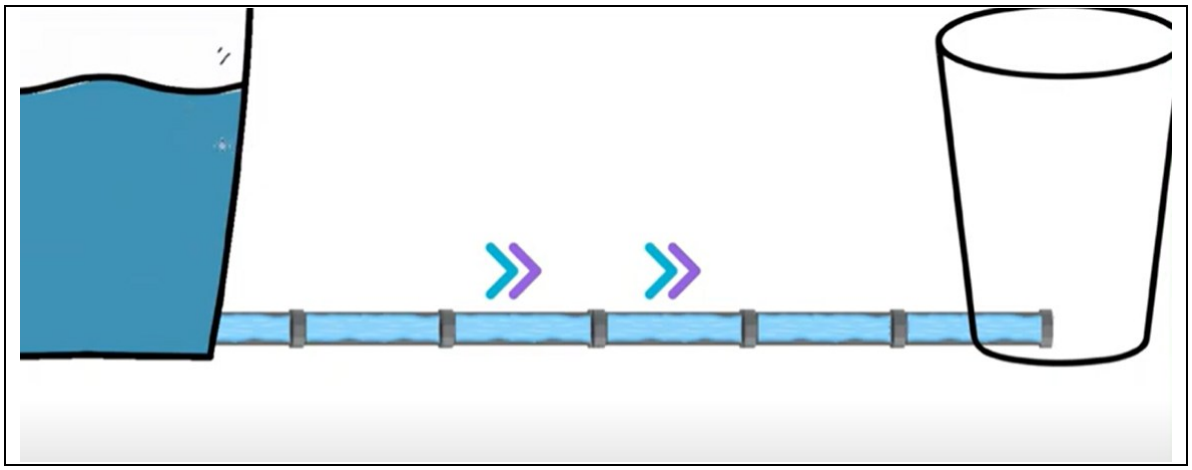
So, in short:
Water Flow → Water Current
Electron Flow → Electric Current

Potential Difference



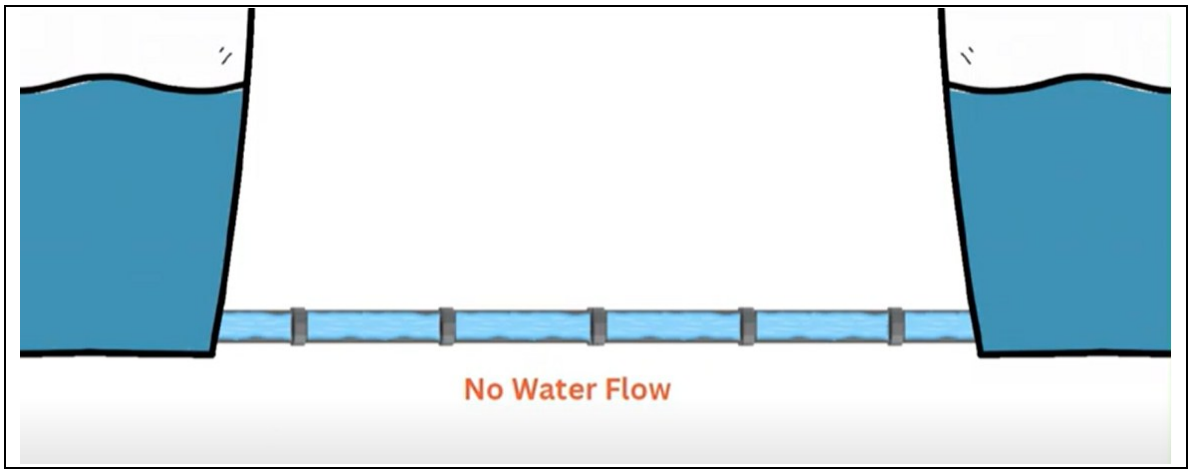
Can we tell just by looking at this pipeline whether the water is flowing from left to right or from right to left? It's quite hard to say just by looking at the image, because the flow itself is invisible. We can see the pipe, but we can't see the direction in which the water is moving.

Case 1:



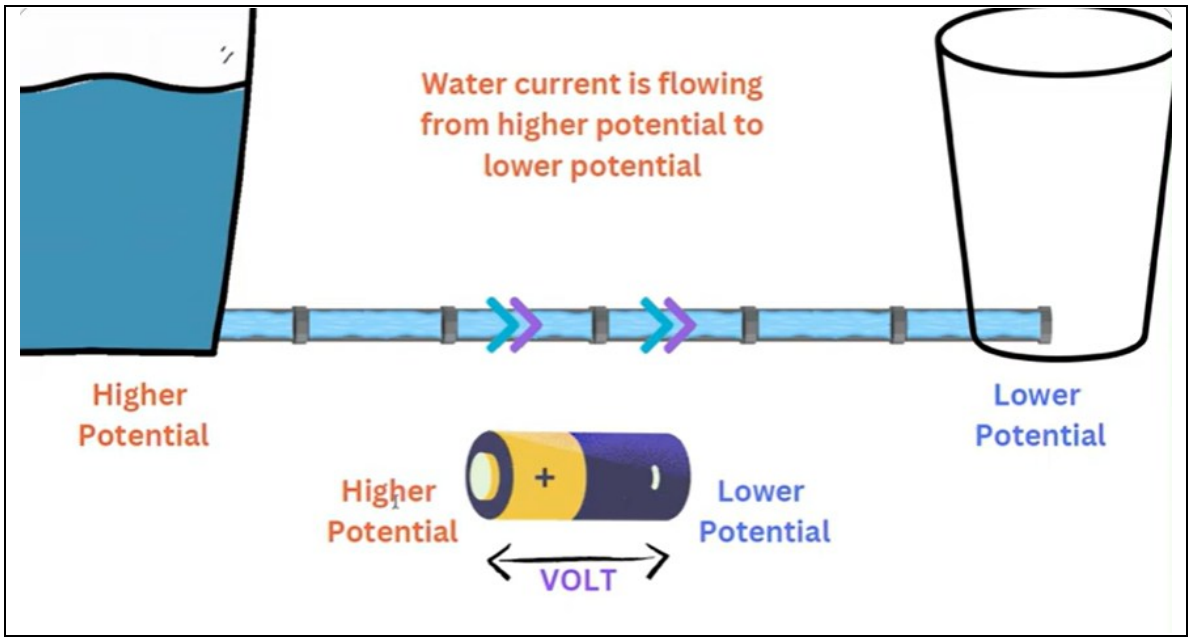
Now, imagine we see an image where a pipeline is connected between two buckets — one bucket on the left side has more water, while the other bucket on the right side is almost empty. In this situation, we can easily say that the water will naturally flow from the fuller bucket (left side) to the empty bucket (right side). Why can we say this so confidently? Because we have all seen this happen in real life — whenever we connect two containers with a pipe, water always flows from the fuller one to the emptier one. It's a natural behavior of fluids, flowing from where there's more to where there's less.

Case 2:



Now imagine a different case. If both buckets have equal amounts of water and are connected by a pipe, then no water will flow from one to the other. Why? Because for water to flow, there must be a difference — one side must have more water (higher potential), and the other must have less water (lower potential). As long as the water level on both sides is exactly the same, the flow will remain zero. There's simply no reason for the water to move.

Just like water needs a higher level on one side and a lower level on the other to flow, electric current also needs a difference to flow. In the case of electricity, this difference is called potential difference.

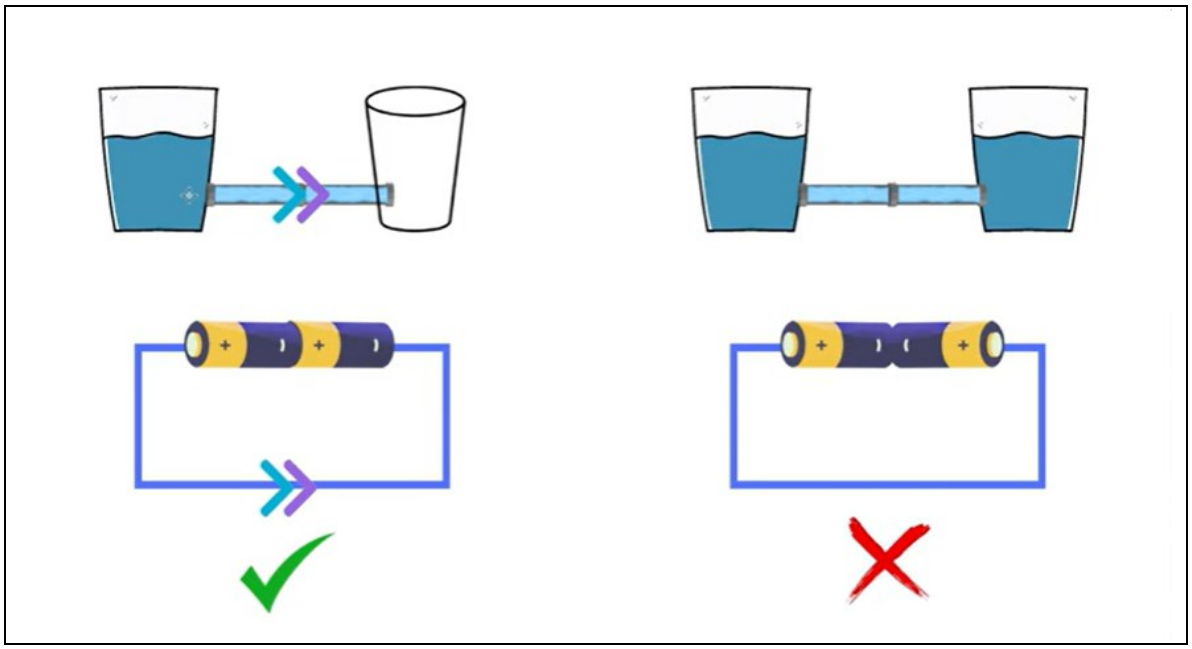


Now, what creates this potential difference? A battery acts as a power source that pushes electrons through a circuit.

If one end of the battery has higher potential (more electrical push) and the other end has lower potential, and we connect these two ends using a wire — only then will the electrons start flowing, which we call electric current.

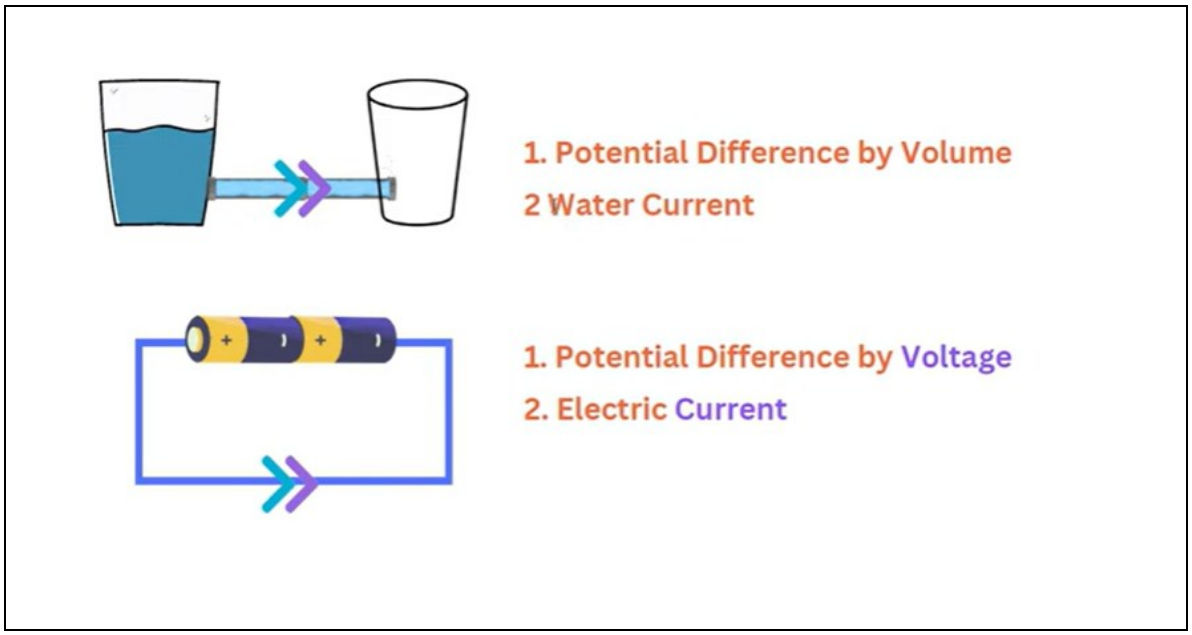
If there is no potential difference between the two sides of the battery, the electrons have no reason to move — and the current will be zero. It's exactly like the case of two buckets filled equally with water — nothing moves if there's no difference.

We measure this potential difference using a unit called **volts**. The greater the voltage of the battery, the stronger the push on the electrons, meaning more current can flow through the circuit. A higher voltage is like having a bigger height difference in a waterfall — the water falls more forcefully.



To summarize, there are two important conditions to help us clearly understand when water or electric current will flow:

1. Different volumes of water at both ends of the pipe: One end has high volume (high potential), and the other end has low volume (low potential). This difference creates water flow.
   Similarly, electric current will flow only when there is a potential difference (voltage difference) between the two terminals of a battery. One terminal must be at a higher potential, and the other at a lower potential. Once these two ends are connected by a wire, current will start flowing through the circuit.
2. Equal water levels at both ends: If both ends have equal volumes — meaning equal potential — then no water will flow.
   Similarly, if a battery is connected in such a way that both terminals have the same potential (for example, high potential on both sides), then no current will flow, even if connected with a wire.



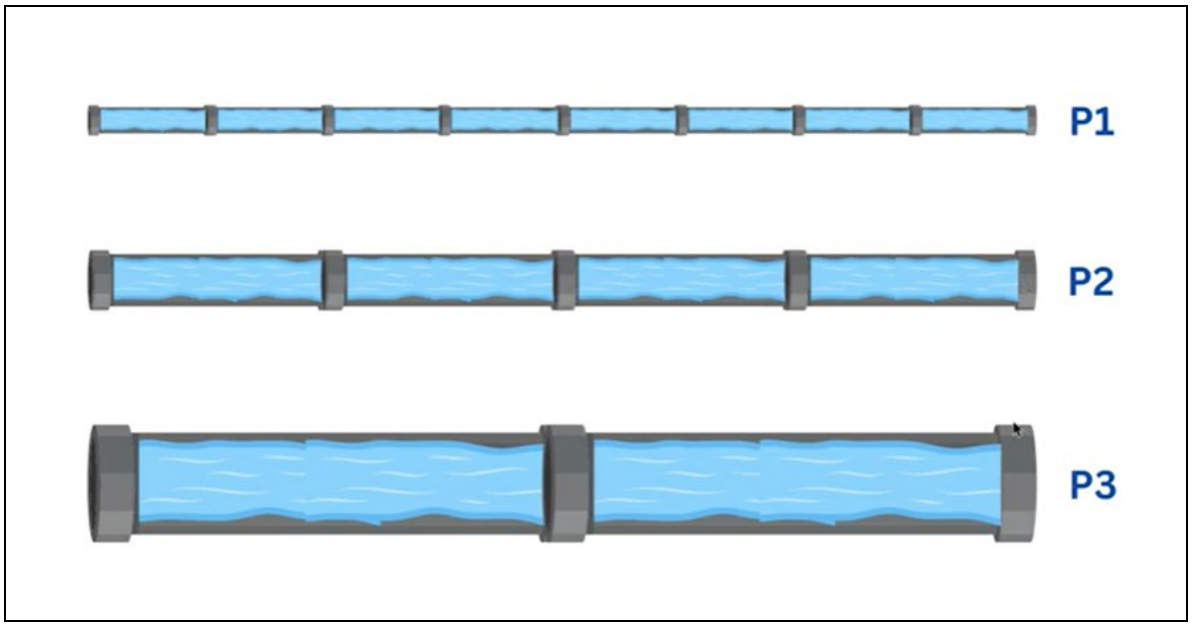
But there's still one more important thing we cannot ignore — and that's the pipeline in the case of water or the wire in the case of electricity.

If there is no pipeline for the water to travel through, the water will remain stuck where it is, no matter how full one bucket is. Likewise, if there is no wire to connect the two terminals of the battery, the electrons won't have any path to travel, and no current will flow.

That's why it's very important to understand not only the potential difference but also the role of the wire in completing the circuit. Without a proper wire (or conducting path), electric current simply cannot flow at all.

Electric Resistance

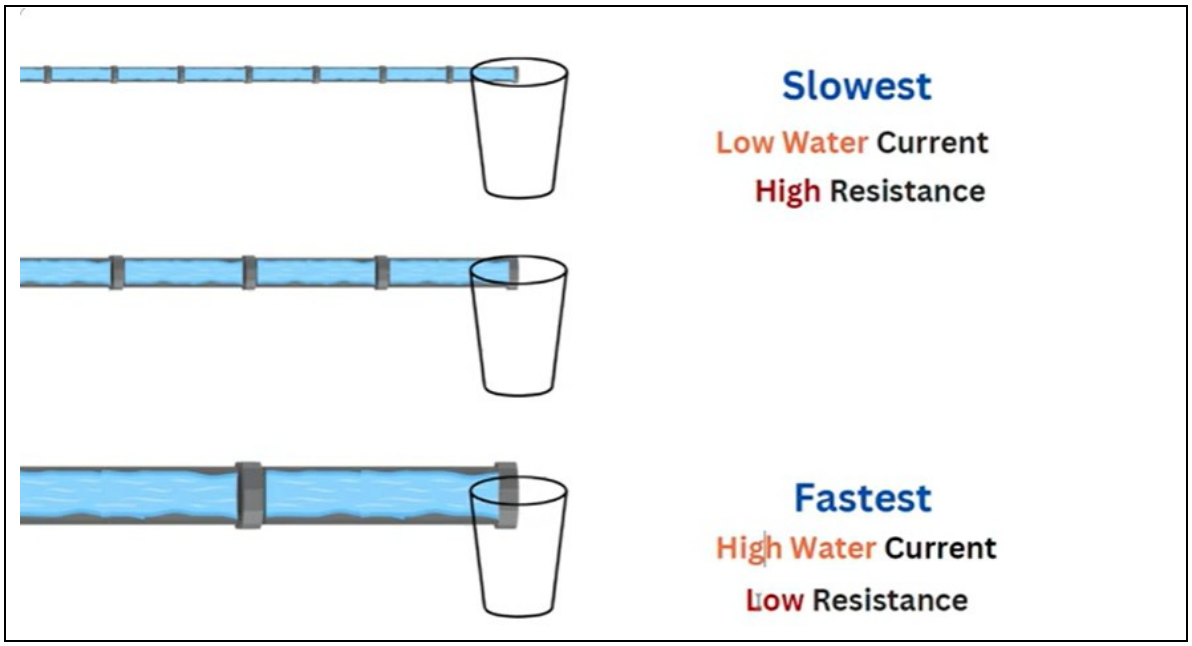
Imagine that we have three different types of pipes in front of us, each with a different thickness:



1. P1 — A very thin pipe
2. P2 — A medium-sized pipe
3. P3 — A very thick pipe

Now, let's perform a small experiment in our imagination. Suppose we are trying to fill a bucket with water using each of these pipes one by one. What will happen?

- If we use P1 (the very thin pipe), only a very small amount of water can pass through at a time. As a result, the bucket will take a very long time to fill. This means that the flow of water is slow, and we can say that this pipe is offering high resistance to the flow of water.
- Next, if we use P2 (the medium-sized pipe), the water flow will be faster than the thin pipe but slower than the thick one. The bucket will fill at a moderate speed, meaning this pipe has medium resistance.
- Finally, when we use P3 (the very thick pipe), the water will flow very easily and quickly, and the bucket will be filled in no time. Here, we can say that the pipe is offering very little resistance to the water flow.



Now, let's connect this idea to electricity.

Just like water flows through pipes, electric current flows through wires. And just like the size of the pipe affects how fast water can flow, the thickness of the wire affects how easily electrons (which make up electric current) can flow through it.

Here's the comparison:

- Thin wire → High resistance → Slower electric current flow
- Medium wire → Medium resistance → Moderate current flow
- Thick wire → Low resistance → Faster electric current flow

Why does this happen? Because the thinner the wire, the less space there is for electrons to move through. As electrons try to pass, they collide with particles inside the wire, creating obstruction or resistance. The more collisions, the harder it is for current to pass through.

So we can say:

➡️ Thin wire = More obstruction to electron flow = High resistance
➡️ Thick wire = Less obstruction to electron flow = Low resistance

In simple words, resistance is like a traffic jam for electrons.

- Thin wire = Heavy traffic → Slow movement
- Thick wire = Empty road → Fast movement

This is why, in electrical circuits where we want more current to flow easily, thicker wires are used. And when we want to limit or control the current, thinner wires or special components called resistors are used to provide resistance.

So, just as we choose the pipe size carefully depending on how fast we want to fill the bucket, engineers choose wire thickness carefully depending on how much current needs to flow in a circuit.

Understanding this connection between wire thickness and electric resistance is a key step in learning how real circuits are designed.

Ohm's Law

Now that we've explored the concepts with examples and analogies, let's summarize and focus on the three most important concepts we've learned so far in understanding how electricity works in circuits:

1️⃣ Voltage (V)

Voltage represents the potential difference between two points in a circuit. You can think of it as the force or pressure that pushes electrons through a wire, just like water pressure pushes water through a pipe.

Higher voltage → Stronger push → More potential for current to flow

Without voltage, there's no driving force for the electrons to start moving. This is why a battery or some kind of power source is always required in a circuit.

2️⃣ Current (I)

Current is the actual flow of electrons through a wire or circuit, just like the flow of water inside a pipe. It tells us how many electrons are flowing per second through the wire.

Higher current → More electrons flowing → More work being done (like lighting a bulb, running a fan, etc.)

The more pressure (voltage) you apply and the easier the path (low resistance), the greater the current that will flow.

3️⃣ Resistance (R)

Resistance is like the opposition or obstruction faced by electrons as they try to move. Just like narrow or clogged pipes slow down water flow, certain materials or thin wires slow down the flow of electrons.

Higher resistance → Harder for electrons to flow → Less current

Lower resistance → Easier for electrons to flow → More current

Materials like copper and aluminum have low resistance, which is why they are commonly used to make electric wires.

🔗 How Are Voltage, Current, and Resistance Connected?

Now comes the interesting part — these three things are mathematically connected through a very important rule in electronics known as Ohm's Law. This law helps us calculate how much current will flow in a circuit based on the voltage and resistance.

The formula is:

$I = V / R$

Where:

I = Current (in Amperes or A)

V = Voltage (in Volts or V)

R = Resistance (in Ohms or Ω)

This formula means:

If Voltage increases → Current increases (if resistance stays the same)

If Resistance increases → Current decreases (if voltage stays the same)

Sometimes you'll also see Ohm's Law written like this:

$V = I \times R$

Same relationship, just rearranged to find different values based on what you need to know.

Let's connect it back to the water pipe analogy:

Voltage = Water pressure

Current = Water flow

Resistance = Narrowness of the pipe

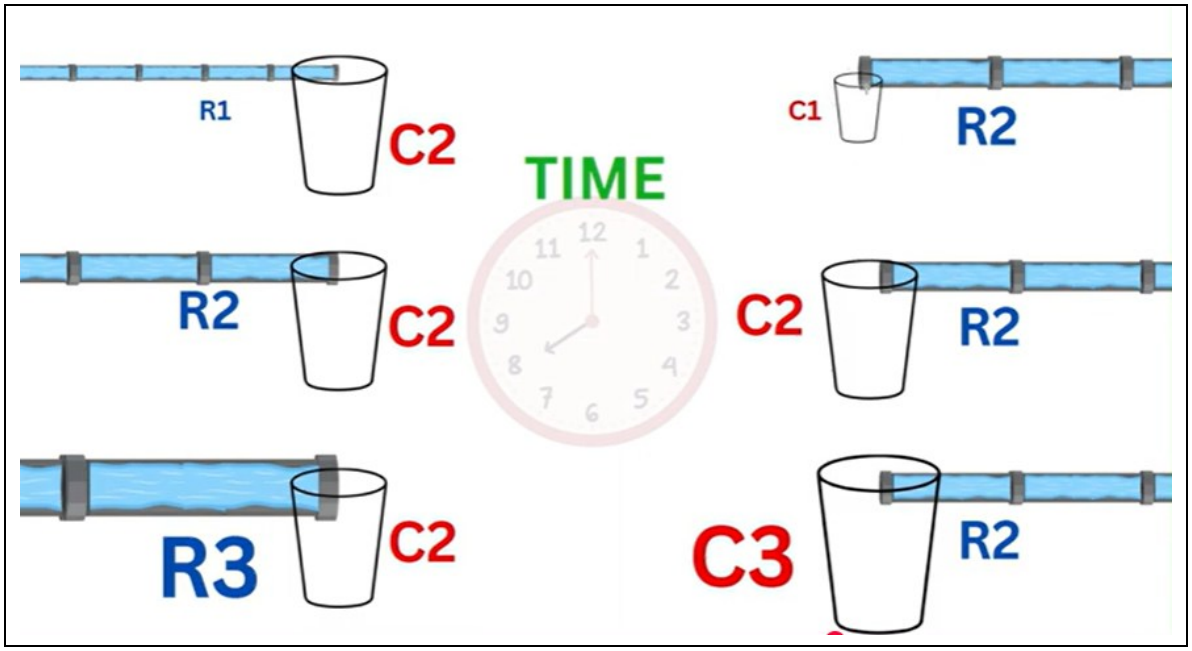
If you increase the water pressure (voltage), more water will flow (higher current), unless the pipe is very narrow (high resistance), which slows it down.

Understanding Ohm's Law is a fundamental building block for learning circuit design and electronics. It helps engineers calculate the right values for components like resistors, wires, and power supplies to ensure circuits work properly.

⏳ Time Constant

Now, let's explore a very important concept related to time in electronics — this is called the Time Constant.

To understand this easily, let's go back to our water flow analogy.



Suppose we have several pipelines of different thicknesses with different resistances (R1, R2, R3), and each one is connected to a bucket of the same capacity (C2) as shown on the left side of the image. If we start filling these buckets with water, which bucket will fill up faster?

The bucket connected to the thickest pipe will fill up the fastest. Why? Because the thick pipe has less resistance, which means more water can flow per second.

So far, so good. Now, let’s change the scenario a bit.

Now imagine that we use the same-sized pipe for each setup (as shown on the right side of the image), but change the size of the buckets instead.

In this case, the smallest bucket will fill up the fastest because there’s less space to fill.

So from this, we can see that the time taken to fill a bucket depends on two things:

1. The thickness of the pipe → Which controls how easily water can flow → This represents Resistance (R) in an electric circuit.
2. The size of the bucket → Which controls how much water needs to be filled → This represents Capacitance (C) in an electric circuit.

In electronics, this whole setup is equivalent to charging a capacitor with electric current.

- The bucket = Capacitor (C) → Stores electric charge (like the bucket stores water)
- The pipe = Resistor (R) → Controls how quickly electrons can flow into the capacitor (like the pipe controls water flow)

And here’s the most important formula that connects these two:

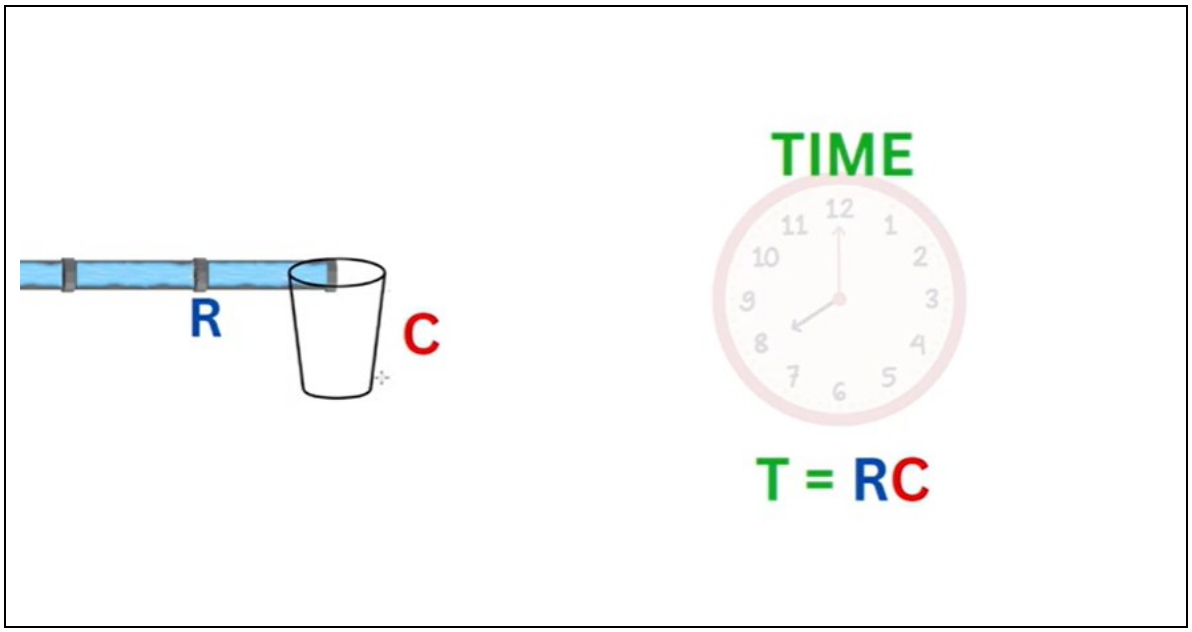
τ (Time Constant) = R × C

Where:

τ (tau) = Time constant (measured in seconds)

R = Resistance (in ohms, Ω)

C = Capacitance (in farads, F)



The Time Constant (τ) tells us how quickly a capacitor charges up or discharges in a circuit.

Smaller τ → Faster charging/discharging

Larger τ → Slower charging/discharging

Example:

High resistance (thin pipe) + Large capacitor (big bucket) → Takes more time to fill

Low resistance (thick pipe) + Small capacitor (small bucket) → Takes less time to fill

When designing chips and circuits, understanding the time constant is very important for performance. Why? Because we need to know how quickly parts of the circuit can respond when signals change.

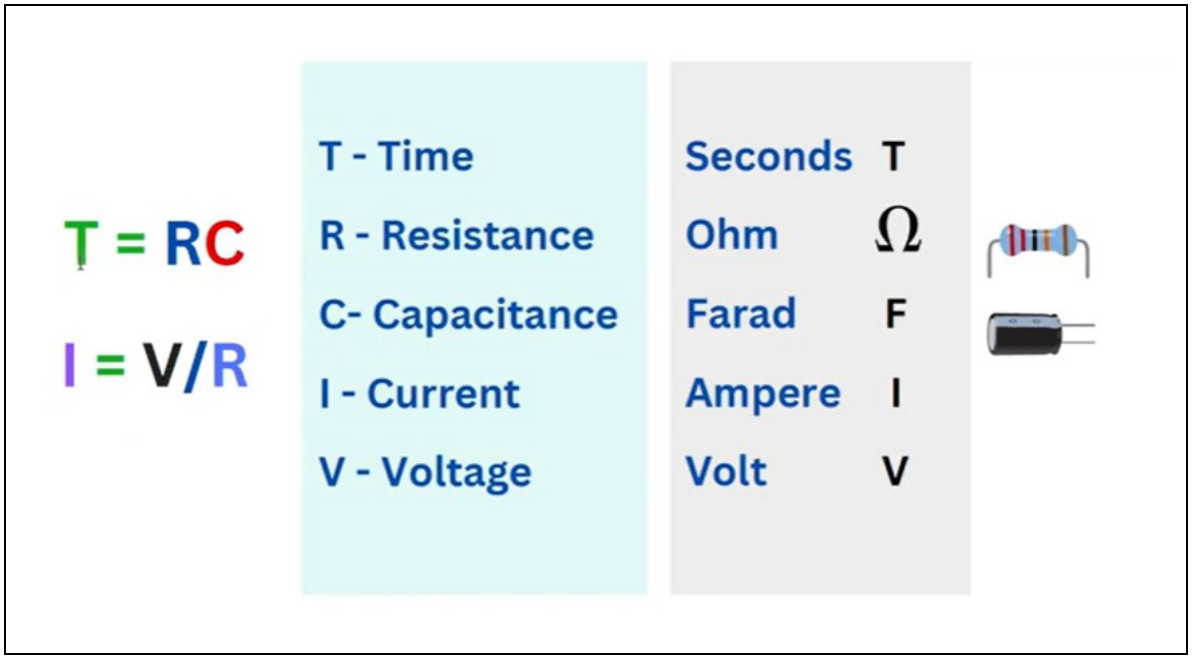
For example, in high-speed processors or digital systems, fast charging and discharging of capacitors ensures quick switching between binary signals (0s and 1s).

We will explore capacitors and their behavior in more detail in later chapters. But for now, just remember this key takeaway:

Time Constant (τ) = Resistance × Capacitance (R × C) → It controls how fast a capacitor charges or discharges in a circuit.

🔍 Concept Recap + Units

Let's now recap what we've learned so far along with their symbols and units, so that everything stays clear in your mind:



1. Time Constant (τ)
   Formula: τ = R × C
   Unit: seconds (s)
   It tells us how quickly a capacitor charges or discharges.

2. Resistance (R)
   The obstruction to the flow of electrons
   Unit: ohm (Ω)
   The higher the resistance, the slower the current will flow.

3. Capacitance (C)
   The ability of a component (like a capacitor) to store charge
   Unit: farad (F)
   Bigger capacitance means it can store more charge.

4. Current (I)
   The actual flow of electrons
   Unit: ampere (A)
   Higher current means more electrons flowing per second.

5. Voltage (V)
   The push or force that drives electrons
   Unit: volt (V)
   Higher voltage means a stronger push for electrons to flow.

Example: If you look at a small battery, you might see something like 1.5V written on it. This means the battery provides 1.5 volts of potential difference.

Sometimes, you'll also see current ratings written, like 500mA or 1000mA (mA = milliampere), which tells you how much current it can supply safely.

6. Ohm's Law
   I = V / R

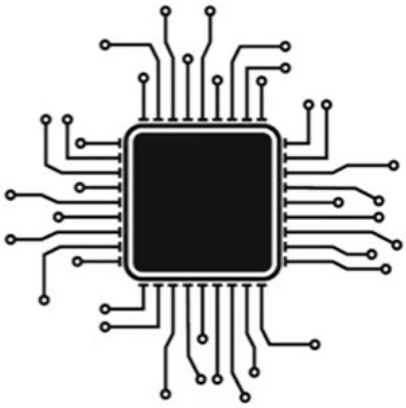
🕒 Inside a Chip: Why Delay Happens

We've explored an important question — why don't we get an output instantly when we provide an input? Why does it take some time for the input signal to convert into output?

The answer lies in two fundamental electronic components: Resistance (R) and Capacitance (C). Every electronic circuit, especially inside a chip, contains these two elements. Together, they form what is called an **RC Network**.

As soon as you provide an input to a circuit, this RC Network causes a small delay before the output is generated. This is why there's always a slight gap between giving an input and receiving an output.

Water Network of a City



R C Network inside a Chip

Let’s take an easy-to-understand example — the water supply system of a large city:

Imagine a big water tank (like a battery) that supplies water to different homes. The water travels through pipes (similar to wires in a circuit) to reach smaller overhead tanks in individual homes (which work like capacitors).

When you open your tap to fill a glass of water, it takes a few seconds for the water to flow, depending on two things:

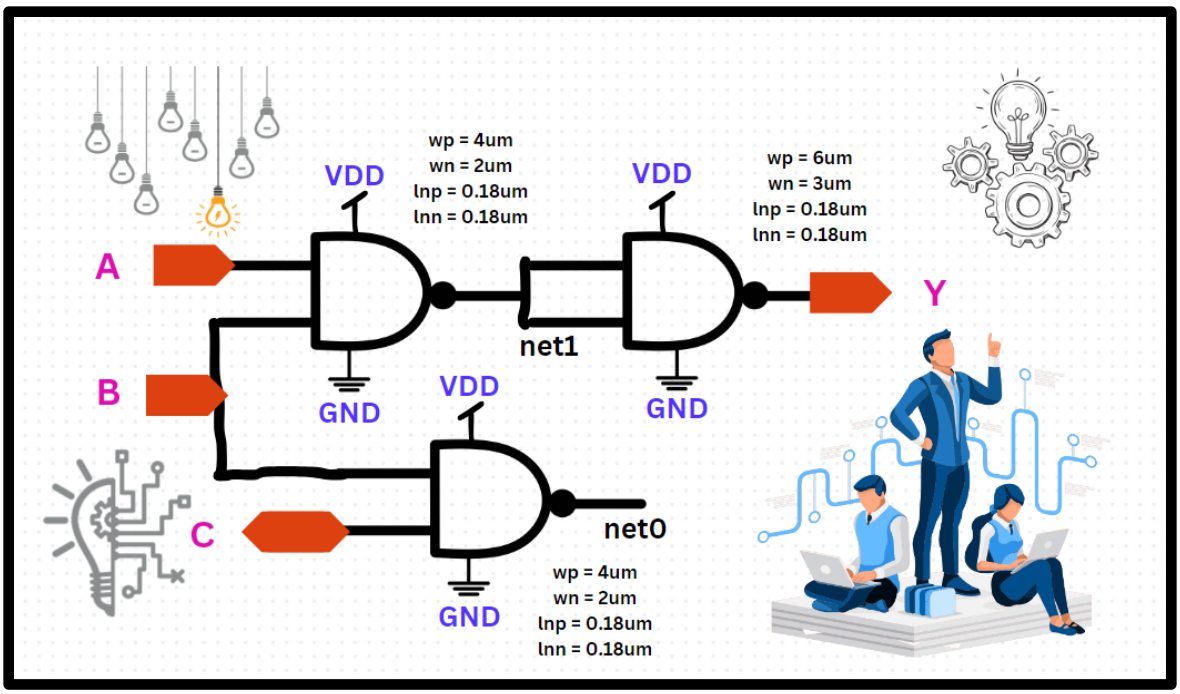
- Pipe Thickness → Thicker pipes = faster water → This is like Resistance (R) in a circuit.
- Tank Size → Smaller tanks fill faster → This is like Capacitance (C) in a circuit.

If the pipes are very narrow or the tanks are large, water takes more time to reach your glass. Similarly, in a circuit, higher resistance and higher capacitance lead to greater delay in producing output.

The exact same principle applies inside electronic chips. Whenever a signal travels inside a chip, it has to pass through this RC Network, causing a time delay.

Less delay = faster chip = better performance

That’s why chip designers always work to reduce these delays for better speed.



Functionality of Logic Gate using CMOS

Overview & Purpose

In this chapter, we will learn how to design Standard Logic Gates at the component level. Logic gates are the building blocks of all digital circuits, and understanding their design is essential for anyone interested in electronics or chip design.

We will explore how these gates work, study their behavior, and most importantly, dive deeper into their internal structure at the component level. This means we won't just see them as black boxes giving output for given inputs — we will actually learn how they are built using CMOS.

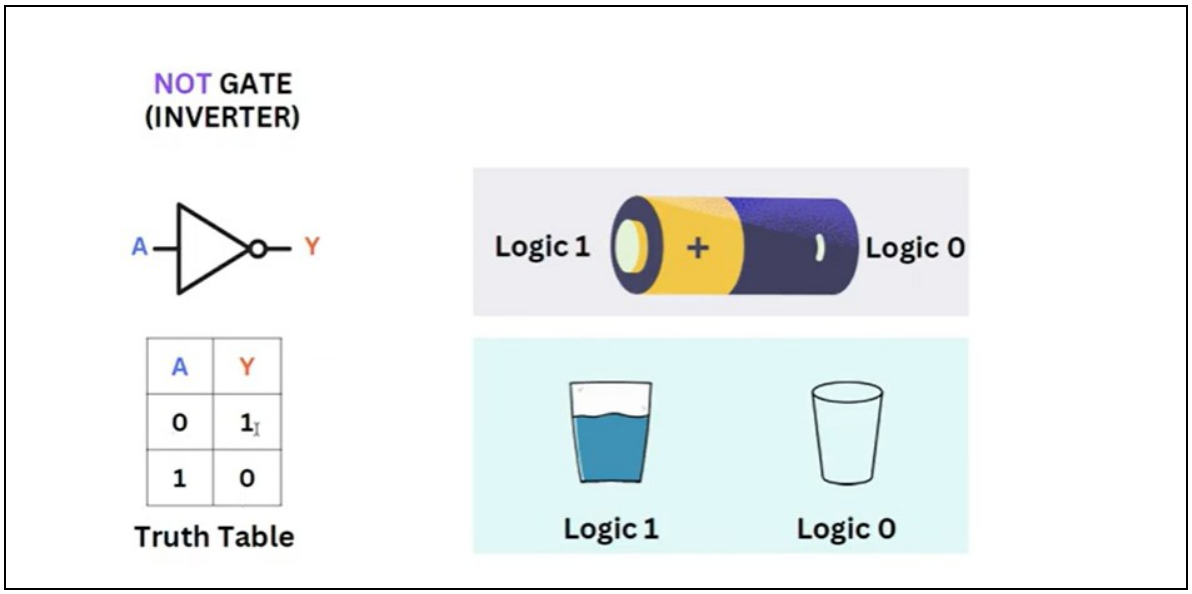
By the end of this chapter, you will have a clear understanding of how logic gates function internally, how they process signals, and how they are designed inside electronic circuits.

This foundational knowledge is crucial for moving ahead in the field of digital electronics and semiconductor chip design.

NOT Gate

Let's start with the NOT Gate. We've already seen its behavior and functionality in detail in the previous chapter. The truth table of a NOT Gate (also called an Inverter) is quite simple:

- If the input is 0, the output is 1
- If the input is 1, the output is 0



But what does it actually mean to give 1 or 0 at the input? And what does it physically mean to get 1 or 0 at the output?
Is there some actual number "0" or "1" moving through the wire? → No.

As we've already discussed in earlier chapters, 0 and 1 are just symbols or representations.
They can stand for things like True-False, ON-OFF, Yes-No — depending on the context. But in electronics, 0 and 1 specifically correspond to voltages:

- 0 → Lower potential → Ground → 0 volts
- 1 → Higher potential → Positive voltage

For example:

- In a circuit with a 5V battery → Logic 1 = 5V, Logic 0 = 0V (ground)

- With a 1.5V battery → Logic 1 = 1.5V, Logic 0 = 0V

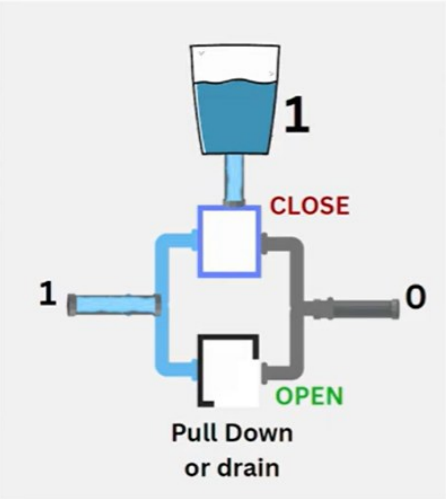
So in digital electronics:
1 = High voltage, 0 = Low voltage

We use 0 and 1 for simplicity because they are easy to write and think about. Just like in the water bucket example we used earlier:

- Empty bucket → Logic 0
- Full bucket → Logic 1

Now that we've refreshed this concept, let's move ahead to understand how the NOT Gate works internally. We'll first use the water and pipeline analogy to simplify the concept and then see its actual electronic explanation with components.

Pull-Up and Pull-Down Networks



In this image, we are explaining the NOT Gate using the analogy of water pipes and control boxes (valves).

Imagine: there is a large tank filled with water placed at the top. Below this tank, there's a pipeline system with two possible paths — one path leads downward to a drain, and the other path leads forward to the output.

Now, this system is arranged in such a way that it automatically controls where the water should go, based on whether water is already present in the pipeline or not:

- If water is present in the pipeline → the blue box (valve) at the top will automatically close, and the black box (valve) at the bottom will open.
- If there is no water in the pipeline → the black box will close, and the blue box will open, allowing water to flow forward toward the output.

Let's now observe how the NOT Gate behaves under both conditions of input:

- Input = 1 (Water is present):

When we provide Input = 1, it means that water has entered the pipeline.

As soon as this happens, the upper box (blue valve) closes, blocking any further water flow from the tank to the output.

Simultaneously, the lower box (black valve) opens, creating a way for the water in the pipe to drain downward.

Result: The pipeline becomes empty, and Output = 0. Even though water entered, the system drained it out, leaving nothing to reach the output. Hence, the output is zero.

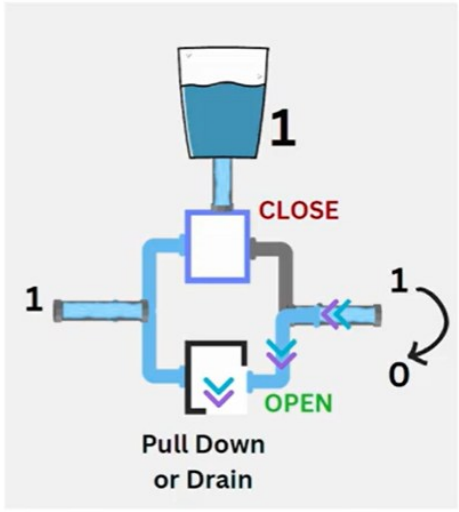
- Input = 1 (Water is present), Output = 1 (Initial Condition):

What does this case mean? It refers to the moment when water initially enters the pipe.

As the pipe starts filling with water, the blue box (top valve) closes immediately, stopping any further incoming water.

The black box (bottom valve) opens, allowing the water already present to drain out through the lower path. Once the water is drained, the pipeline is empty once again.

Result: Output becomes 0, because the path to the output has no water left in it anymore.



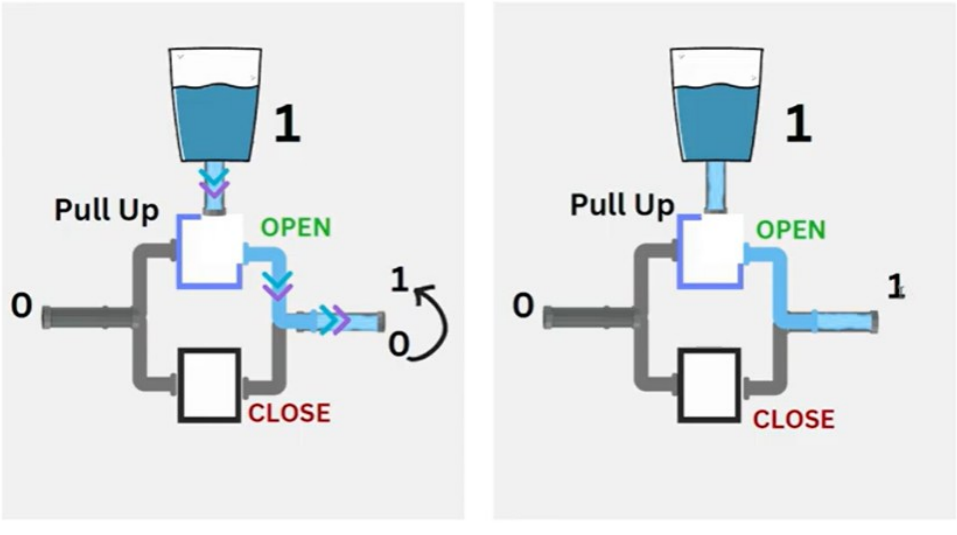
- Input = 0 (No water present):

Now let's consider when the input is 0 — meaning no water is present in the pipeline.

As soon as this condition is met, the blue box (top valve) opens, allowing fresh water to come down from the tank toward the output.

The black box (bottom valve) closes, preventing any chance of the water draining out before reaching the output.

Result: Water smoothly flows directly to the output, making Output = 1.



Now, these two valves in our water analogy are equivalent to two important electronic components:

The black box is called the **Pull-Down Network** or Pull-Down Box.

The blue box is known as the **Pull-Up Network**.

When we talk about buckets and flowing water, it's just a way to help visualize the concept. In real electronic circuits, there is no physical bucket or water; instead, electric current and voltage levels are used.

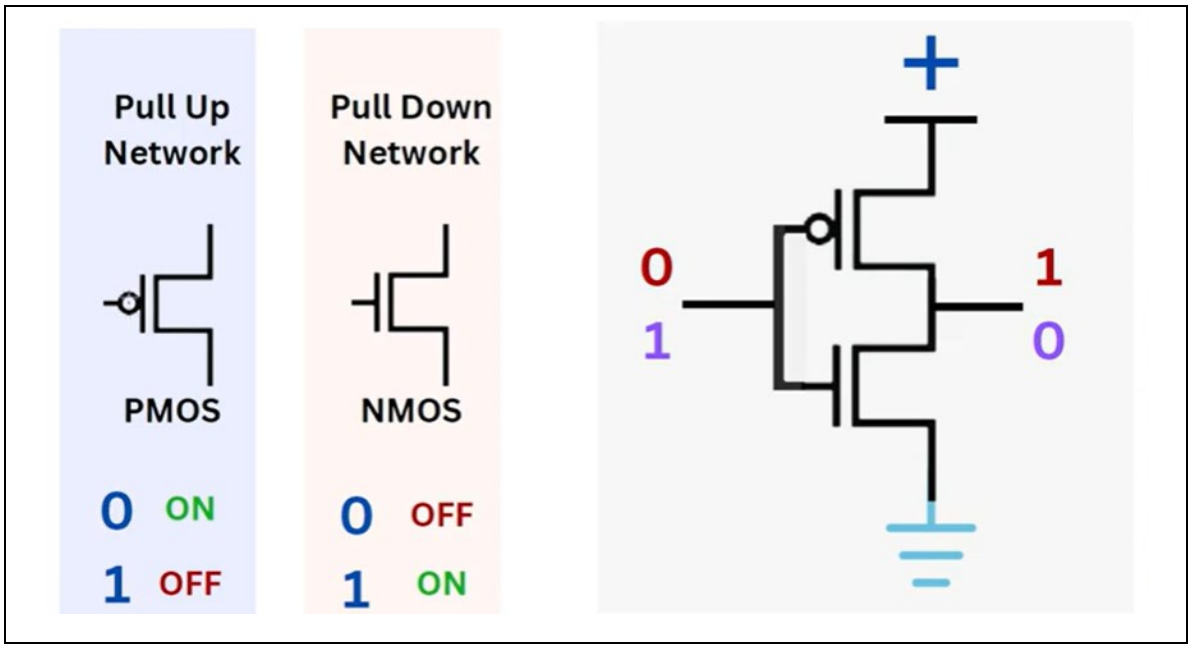
The tank at the top can be thought of as being connected to a battery's positive terminal — meaning that positive voltage (logic 1) comes from above.

🟦 Pull-Up Network = PMOS

- The blue box (upper valve) represents the Pull-Up Network, known in electronics as PMOS.
- A helpful way to remember this is that "Pull-Up" and "PMOS" both start with the letter P. PMOS stands for P-type Metal Oxide Semiconductor.
- Just like the blue box opens when Input = 0, PMOS also turns ON when Input = 0.

⬛ Pull-Down Network = NMOS

- Similarly, the black box (lower valve) represents the Pull-Down Network, called NMOS in electronics.
- NMOS stands for N-type Metal Oxide Semiconductor.
- Just like the black box opens when Input = 1, NMOS also turns ON when Input = 1.



PMOS + NMOS = CMOS

When we combine one PMOS and one NMOS together to form a complete circuit, that combined structure is called CMOS, short for Complementary Metal Oxide Semiconductor. CMOS forms the foundation of modern digital chip design. It is used in building logic gates, processors, memory chips, and almost all forms of digital electronics.

🧬 CMOS - Complementary Metal Oxide Semiconductor

When we combine a PMOS and an NMOS, we form a very efficient and reliable circuit known as CMOS, which stands for Complementary Metal Oxide Semiconductor. This is the basic building block of most digital electronic devices today, including processors, memory, and logic gates.

Connections:

- The PMOS is always connected to the positive voltage supply (Vdd) from the top.
- The NMOS is always connected to ground (0V) from the bottom.
- Both of these transistors are connected together at one common point — and this is where we take the Output from.

So, the Output wire is shared between PMOS and NMOS, and depending on the input signal, either PMOS or NMOS will control what voltage appears at the Output.

Case 1: Input = 0:

- When the Input = 0, the PMOS turns ON (remember: PMOS is ON when input is 0).
- The ON PMOS creates a direct connection between the positive voltage (Vdd) and the Output wire.
- At the same time, the NMOS turns OFF, which disconnects the Output from ground.
- Result: Output is directly connected to Vdd, so Output = 1 (logic high).

Case 2: Input = 1:

- When the Input = 1, the PMOS turns OFF, so it disconnects the Output from the positive supply.
- Meanwhile, the NMOS turns ON (NMOS is ON when input is 1).
- The ON NMOS connects the Output directly to ground (0V).
- Result: Output is pulled to 0V, so Output = 0 (logic low).

Input	Which is ON?	Output
0	PMOS	1 (Positive voltage)
1	NMOS	0 (Ground)

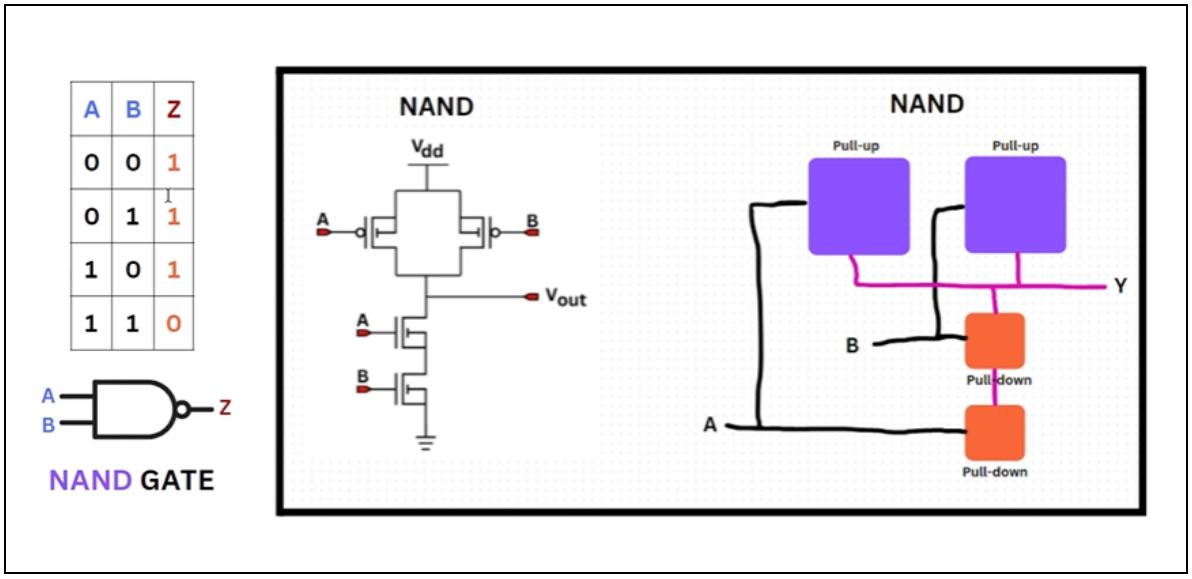
Creating NAND Gate with CMOS

We know that the truth table of a NAND Gate is as follows:

A (Input)	B (Input)	Output
0	0	1
0	1	1
1	0	1
1	1	0

According to the truth table, when both inputs are 1, the output will be 0; otherwise, the output will always be 1.

To achieve this functionality, we need to create a circuit like the one shown in the image.



To build a CMOS NAND Gate, we need:

- 2 PMOS (Pull-Up)
- 2 NMOS (Pull-Down)

👉 On the upper side (Pull-up Network – PMOS):
Two PMOS transistors are placed on the upper side and are connected in parallel.
One PMOS is connected to input A, and the other to input B. These two are combined together at the Output (Vout).
If either A or B is 0, at least one PMOS will turn ON, and Output will be 1.

👉 On the lower side (Pull-down Network – NMOS):
Two NMOS transistors are placed on the lower side and are connected in series.
The upper NMOS is connected to the Output (Vout), and the lower NMOS is connected to ground (GND).
The gates of both NMOS transistors are also connected to inputs A and B, exactly the same as for the PMOS transistors above.
These inputs (to NMOS and PMOS) are connected together → (A is connected to A, and B is connected to B).

Only when A = 1 and B = 1 will both NMOS turn ON and connect the output to ground, making the Output = 0.

🔍 Step-by-Step Working:

✅ Case 1: A = 0, B = 0
Both PMOS are ON ⇒ Output = 1
Both NMOS are OFF ⇒ No connection to ground below

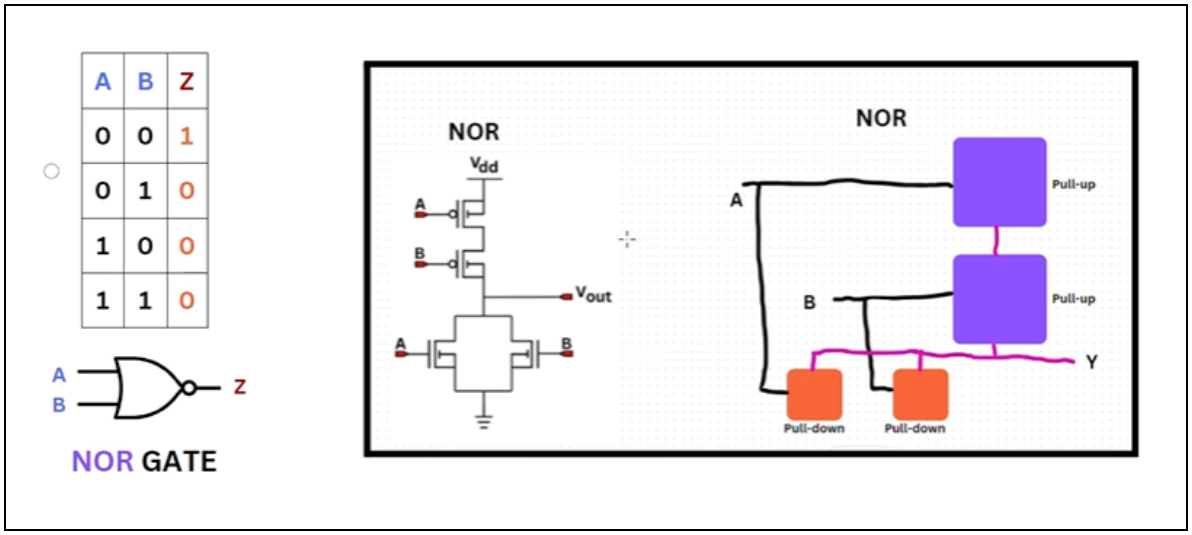
✅ Case 2: A = 0, B = 1 (or vice versa)
One PMOS is ON ⇒ Output = 1
One NMOS is OFF ⇒ Output cannot be drained

✅ Case 3: A = 1, B = 1
Both PMOS are OFF
Both NMOS are ON ⇒ Output drains to ground ⇒ Output = 0

This circuit works like a NAND Gate.

🧩 Creating NOR Gate with CMOS

Let's look at another example of a NOR Gate.



In this circuit, there are two PMOS transistors at the top which are connected in series, and two NMOS transistors at the bottom which are connected in parallel. Now, what does this arrangement mean?

Among the NMOS transistors at the bottom, if even one of them turns ON (meaning if either input A or B is 1), then whatever voltage is present at the output will immediately drain downward to 0.

This setup ensures that as soon as we apply 1 at any input, the output will drop to 0.

Now let's understand this clearly with the help of the Truth Table:

A (Input)	B (Input)	Output
0	0	1
0	1	0
1	0	0
1	1	0

If A = 1 or B = 1, then at least one NMOS will turn ON, and the output will instantly become 0 because it gets a direct path to ground.
There's only one special condition where the output becomes 1: when A = 0 and B = 0.

At that time, both NMOS remain OFF, meaning there's no path to ground, and both PMOS turn ON (because PMOS always turns ON when its input is 0).
Only in this case does the output connect to Vdd (positive voltage), and the output becomes 1.

That's why in a NOR Gate, the output is 1 in only one condition — when both inputs are 0, and the entire current path is completed through the PMOS transistors to Vdd.

🌐 Universal Gates

The reason for explaining the functionality of NAND and NOR gates in detail is because these two gates are known as Universal Gates in digital electronics.

With the help of these Universal Gates, we can design or create any other logic gate used in digital circuits and systems.

For example:

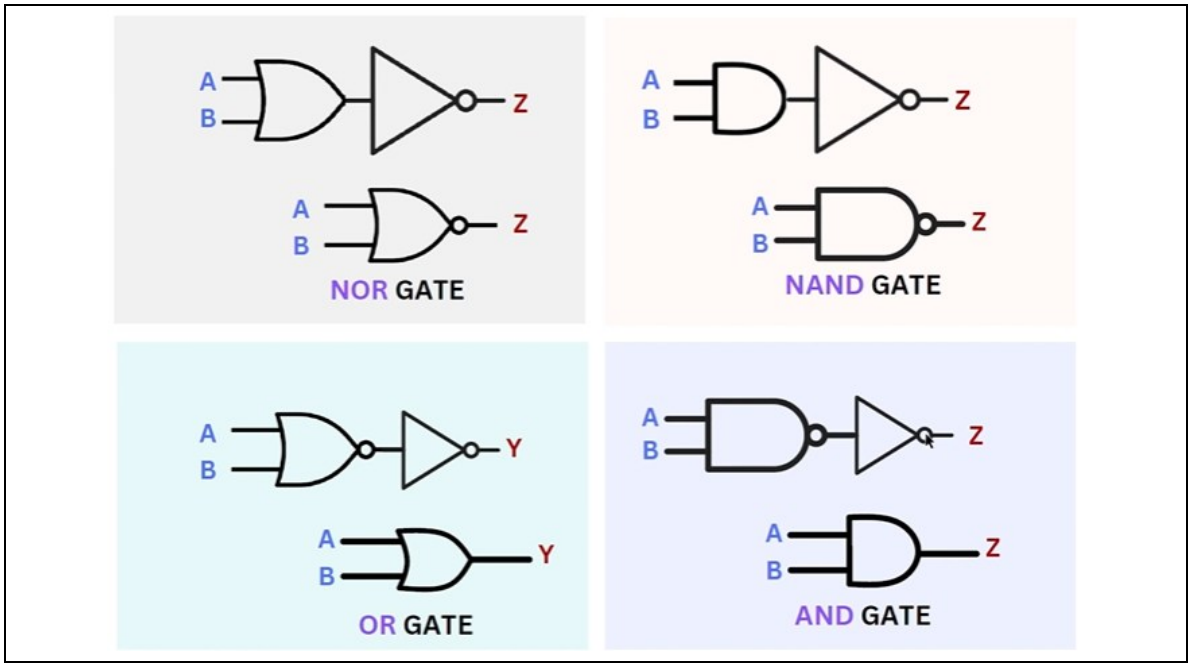
- If we connect an inverter (NOT Gate) after a NOR gate, the overall combination behaves exactly like an OR Gate. So, even though we don't have a direct OR Gate, we can create one using a NOR + NOT arrangement.
- Similarly, if we connect an inverter after a NAND gate, the combination behaves exactly like an AND Gate. Again, NAND + NOT = AND.

In the same way, by using just NAND and NOR gates in different combinations, we can build all other gates like NOT, AND, OR, XOR, XNOR, and so on.

That's why NAND and NOR are called Universal Gates — because they alone are sufficient to build the entire logic family used in digital electronics.

In fact, many real-world digital circuits and chips are built using only NAND gates or only NOR gates, because it simplifies the manufacturing process.

This is why NAND and NOR are so important in digital electronics.



❁ Logic Gates inside a Chip

One very important thing to note is that until now, we have explored the world of digital electronics only up to the component level.

We have understood the symbolic representation of basic gates like NAND, NOR, and NOT, and also learned how they function, along with some basic-level circuit designing concepts.

However, when we step into the world of chip design — meaning actually fabricating a circuit onto a silicon wafer — the entire scenario changes.

Things are no longer just about symbols or simple diagrams on paper.

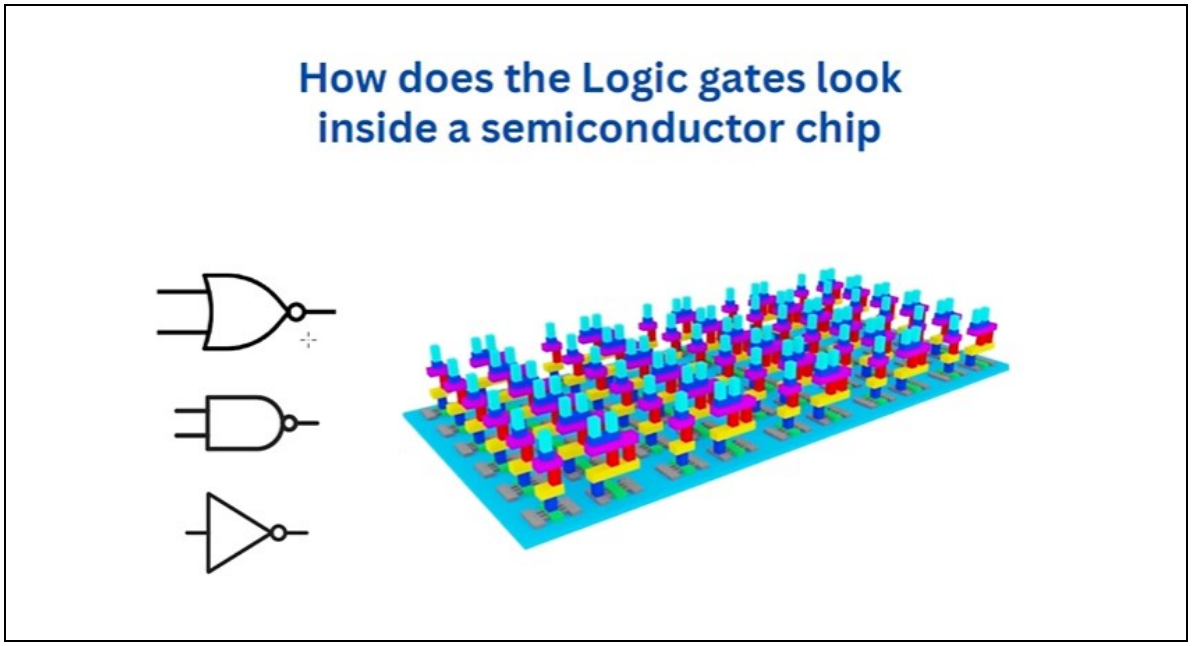
For example:

The NAND gate you've been seeing until now as a simple symbol or the PMOS-NMOS diagrams you studied — these are all schematic representations made for understanding and designing circuits on paper or in software.

But when that same NAND gate is actually built on a silicon chip, it doesn't appear in symbolic form at all.

Instead, it takes the shape of complex 3D structures and layered geometries, formed by depositing and etching various materials like polysilicon, metal, and silicon dioxide on the silicon surface.

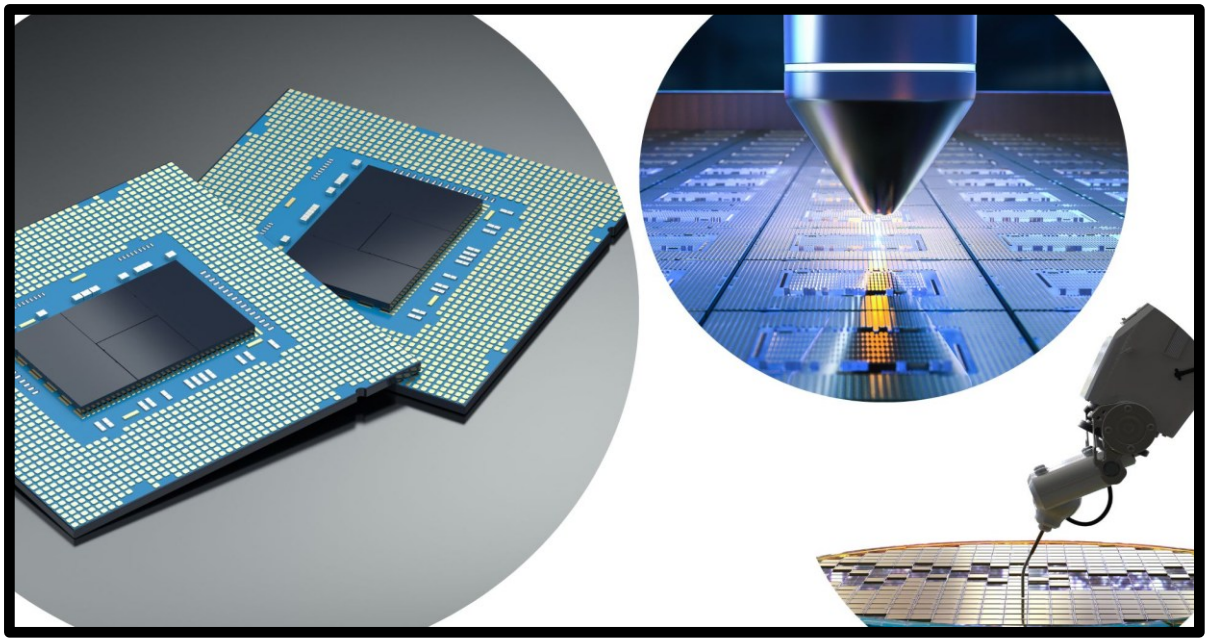
The 3D view shown below is just an example to help visualize how real chips look from the inside.



When designing actual chips, engineers work with multiple layers, where each layer represents a different material, pathway, or electrical function, combined in a highly organized 3D structure.

In fact, chip designing involves several levels of abstraction: schematic diagrams, layout designs, and finally, the 3D fabrication itself.It's a process that connects electronic theory with real-world semiconductor manufacturing.

We will explore this fascinating world of chip-level design in much greater detail in the next chapter. There, you will learn how these logic gates — which we've so far seen as simple symbols — are physically created on silicon, forming the foundation of processors and memory chips we use in everyday devices



Chip Designing, Verification and Fabrication

Overview & Purpose

This chapter is of critical importance because it provides a comprehensive and holistic understanding of the entire process of chip development — beginning from schematic design, progressing through circuit design, illustrating how a chip is constructed on silicon, detailing the fabrication process, and explaining what layout and schematic design entail. In essence, it covers the complete journey from conceptualization to final realization.

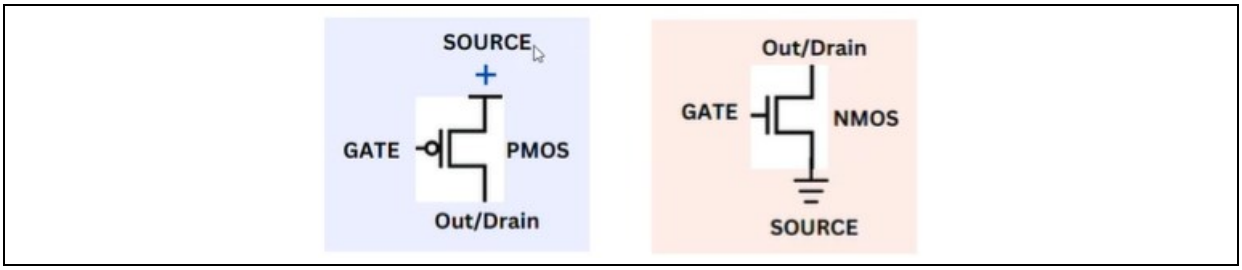
Semiconductor IC Design

Circuit Design

In the previous chapter, you were introduced to the fundamental concepts of NMOS (n-type metal-oxide-semiconductor) and PMOS (p-type metal-oxide-semiconductor) transistors, along with the standard symbols used to represent them. These symbols, whether depicted on a computer screen or drawn in a notebook, serve as abstract representations that help us visualize and communicate circuit elements more easily. When we want to quickly describe or convey the behavior and structure of an NMOS or PMOS transistor to someone, these symbolic diagrams prove extremely useful.

These representations are also referred to as **schematics** or **symbols**. By utilizing a specific combination of these schematic symbols, we can construct logic gates — the building blocks of digital electronics.

It was also discussed previously how various types of logic gates — such as AND, OR, NOT, NAND, NOR, XOR, etc. — can be created using basic arrangements of NMOS and PMOS transistors. Each logic gate, much like the individual transistors, also has its own standardized symbol that represents its behavior and structure.



When multiple logic gates are interconnected systematically, they form a more complex and functional circuit. This entire process is known as **Circuit Design**.

In essence, circuit design involves interconnecting numerous NMOS and PMOS transistors — or alternatively, logic gate symbols — in such a way that the resulting circuit performs a defined, purposeful function. When all the individual components start working together cohesively to achieve a specific task or logic operation, the outcome is what we recognize as a complete and functional circuit.

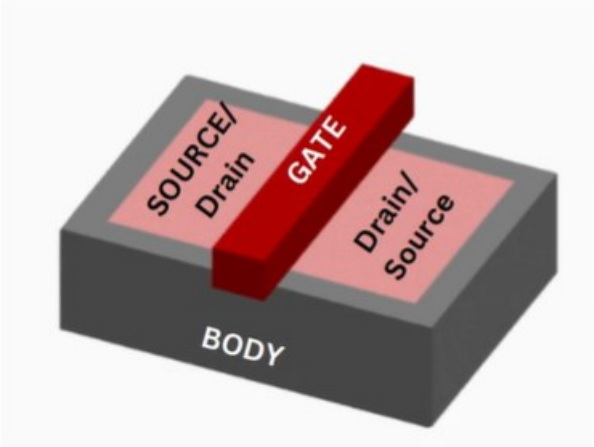
It is very important to emphasize a key point here — the schematic symbols we observe in textbooks, notebooks, or computer software tools are merely visual aids for understanding and conceptual representation. These symbols are not physically constructed on the actual semiconductor chip

Design on Silicon

When we talk about "silicon" in the context of chip design, we are referring specifically to the silicon wafer — a highly purified and flat slice of silicon material upon which the integrated circuits (ICs) are fabricated.

Contrary to the symbolic representations we use during circuit design, the actual implementation on silicon does not involve drawing these symbols directly onto the wafer. Instead, a precisely engineered three-dimensional (3D) physical structure is created on the surface of the silicon wafer. This 3D structure is designed to mimic and perform the same electrical behavior as the schematic symbols of transistors and logic gates.

In these physical structures, there is a Gate region located at the center (typically highlighted in red in illustrative diagrams), flanked on either side by the Source and Drain regions. Beneath this configuration lies a layer referred to as the Body or Substrate — which is, in fact, the actual silicon wafer itself.



Therefore, when we refer to chip design, it is crucial to understand that transistors such as PMOS and NMOS are not just theoretical symbols. Instead, they are physically realized as 3D structures on the silicon wafer through a highly precise fabrication process. These structures look significantly different from the schematic symbols and are built using specialized manufacturing techniques.

However, before we can move on to this highly intricate fabrication stage, there is one more vital step that must be completed — a process known as **Layout Design**.

The Journey from Circuit Design to Chip

- Chip Manufacturing Process

The process of transforming a digital circuit design into a physical semiconductor chip involves several carefully orchestrated stages. Each phase plays a critical role in ensuring the final product functions as intended, both logically and physically. Below is a detailed breakdown of this end-to-end chip development pipeline:

1. Circuit Design:

   The very first step in chip manufacturing is circuit design. At this stage, the functional behavior of the chip is defined. Engineers determine what kind of logic needs to be implemented — which logic gates (such as AND, OR, NOT, NAND, NOR, etc.) will be used, and how these gates will be interconnected to perform a specific function.

   This phase is often referred to as logic-level design, where the goal is to create a circuit diagram that defines the logical flow of signals and operations within the chip. The circuit is typically represented using schematic symbols for NMOS and PMOS transistors or complete logic gates. This forms the foundation for all subsequent steps.

2. Layout Design:

Once the logical circuit is finalized, the next critical step is the layout design. Here, the circuit is translated into a physical representation that specifies how each component will be fabricated on the silicon wafer.

In layout design, each layer of the chip — such as metal layers (used for interconnections), polysilicon, and diffusion regions (used to create transistors) — is drawn with precision using specialized design software. This layered drawing is highly detailed and adheres to specific design rules. The layout serves as the blueprint or mask that guides the physical implementation of the circuit on silicon. It essentially defines the 2D geometry that, once fabricated, results in the 3D structures of the actual chip components.

3. Verification

   After the layout has been completed, it must be thoroughly verified to ensure accuracy and correctness. This step is known as Verification, and it ensures that the physical layout truly represents the intended circuit design and complies with all manufacturing rules.
   Two of the most important verification checks are: DRC and LVS. These checks are essential to prevent costly errors during fabrication and to guarantee that the chip will function as intended.

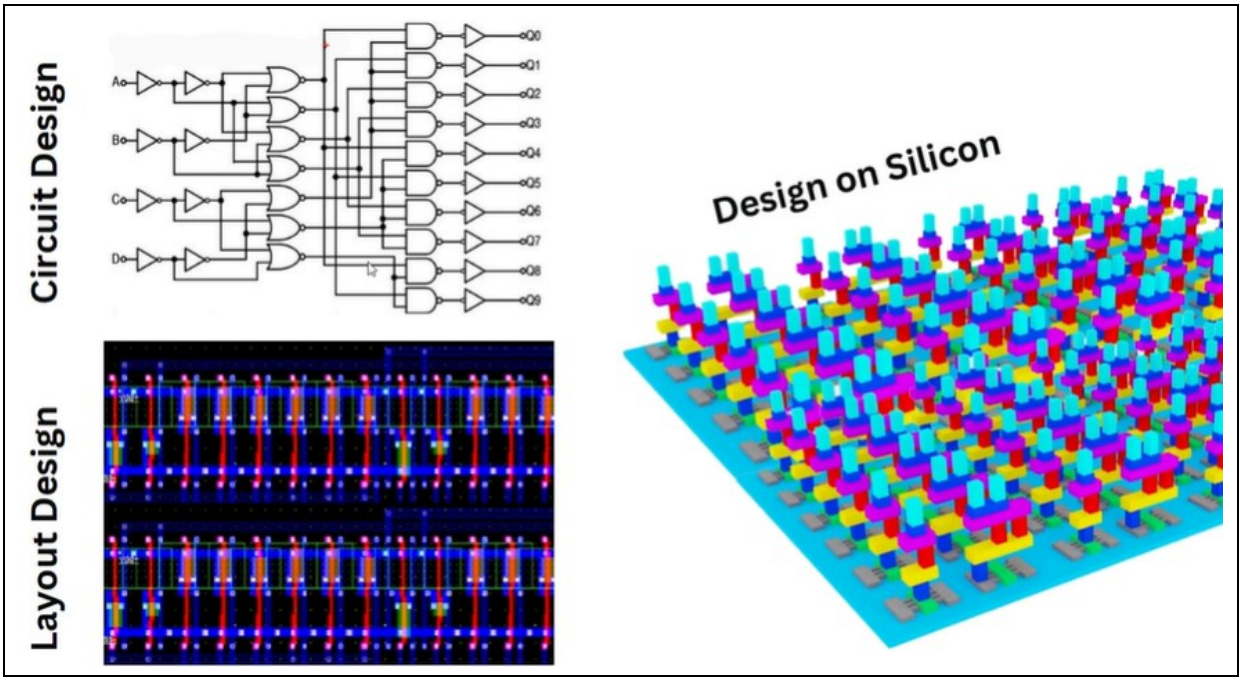
4. Fabrication

   Once the layout passes all verification checks, the design is considered ready for fabrication. In this final step, the verified layout is transferred onto a silicon wafer using a series of advanced photolithography and etching processes.

   The result is a physical chip where each component — from transistors to interconnects — is formed with nanoscale precision. This phase transforms the abstract, logic-based design into a real, working integrated circuit (IC).

This entire process — from circuit design to layout, verification, and fabrication — is what brings a conceptual design into physical reality, enabling the production of chips used in countless electronic devices around the world.

- A Simple Example



To gain a clearer understanding of the entire process, let us examine a simple, illustrative example. In the diagram shown here, you can observe what a basic circuit design looks like. This example represents only a small portion of what an actual circuit might be — real-world circuits can be significantly larger or even simpler, depending on their intended application. However, when designing circuits using computer-aided design (CAD) tools, the visual representation generally resembles what is shown in the example.

Corresponding to this circuit design is its layout design, which is depicted alongside it. This layout is a physical representation that will eventually guide the manufacturing process. It translates the abstract logical structure into a detailed pattern suitable for silicon fabrication.

When we design layouts, they are generated as 2D drawings on the computer screen. However, these 2D representations correspond to actual 3D physical structures once fabricated onto the silicon wafer.

To fully grasp this process, it is important to understand the three interconnected steps —

- First is Circuit Design, in which we define the functionality — like if we input 0, 0, and 1, what should the output be? To achieve that functionality, whatever logic gates and combinations are required — that is called Circuit Design.
- Then, based on the same circuit design, we create a Layout Design, which is a physical 2D representation of the circuit design.
- And from this layout design, the fabricated chip is created, which gives a 3D view on silicon.

Although the layout is created in 2D on the computer, next we will understand how 2D becomes 3D.

Memory chip example

How Complex is Making a Chip – An Interesting Fact!

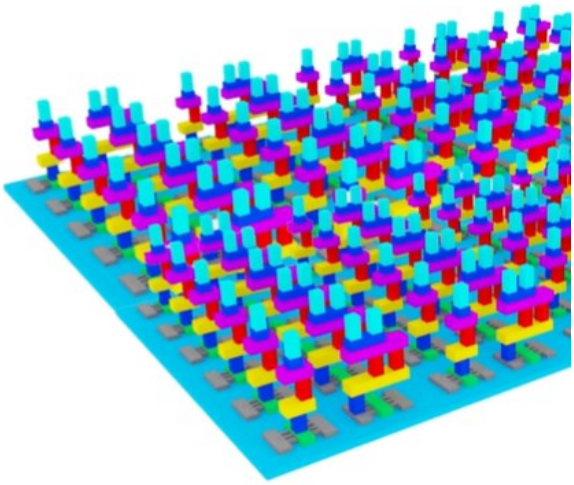
Here's an interesting and insightful point to help you grasp the scale and complexity involved in chip manufacturing — particularly when it comes to memory chips. This example will give you a better sense of just how large, dense, and intricate modern chips can be.

Most of us are familiar with memory cards or storage devices with capacities like 1GB, 10GB, or 16GB. These are commonly used in smartphones, cameras, and other digital devices.

Now, consider this: suppose one transistor — whether NMOS or PMOS — can store one bit of information. Based on that, let's explore the scale of transistors needed for different sizes of memory:

A 1GB memory card would need to store 1 gigabit (1,073,741,824 bits) of data. That means it would require approximately 1.07 billion transistors just to store that amount of data.

Capacity	unit	Crore
1bit	1	
1 byte	8	
1 KB	1024	
1MB	1048576	
1GB	1073741824	**107**
16GB	17179869184	**1718**



Let's break that down further:

- 1 Byte = 8 Bits → So, storing 1 byte would require 8 transistors.
- 1 Kilobyte (KB) = 1024 Bytes → 1024 × 8 = 8192 transistors.
- 1 Megabyte (MB) = 1024 Kilobytes → 8192 × 1024 = 8,388,608 transistors (over 8.3 million).
- 1 Gigabyte (GB) = 1024 Megabytes → Now we need 1,073,741,824 transistors — over 1.07 billion just for data storage.

And remember — these are only the transistors used to store memory.

In addition to storage, a memory chip requires many more transistors to perform other critical functions, such as READ Data, WRITE Data, SELECT Data and CONTROL Signals.

These functions require their own logic circuits, which in turn consist of thousands to millions more transistors. So, the actual transistor count on a chip is significantly higher than the raw storage requirement.

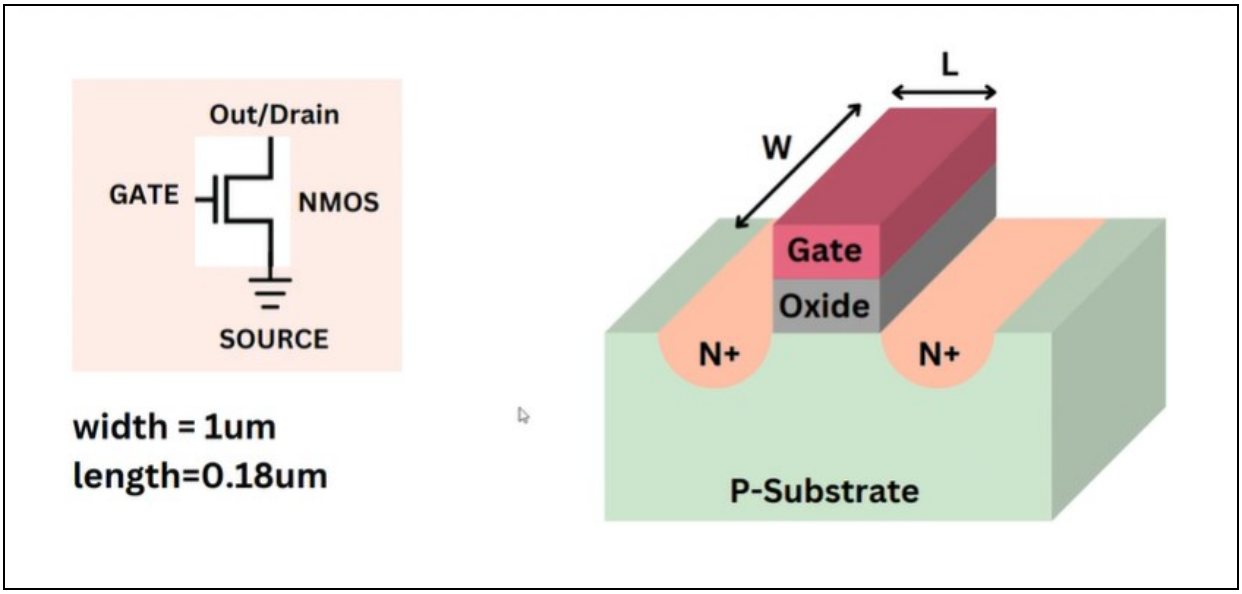
Now, imagine designing and building a 16GB memory chip. That would require over 17 billion transistors, all integrated onto a tiny silicon die measuring just a few square millimeters in size.

Constructing such a dense and precise network of microscopic components is a tremendous engineering feat which includes chemical, mechanical, electronic engineering.

NMOS Design Example

Structure

Let us now explore how the symbol of an NMOS transistor — typically drawn in a circuit diagram — is transformed into an actual 3D structure within a semiconductor chip



On the left, you might see the standard NMOS symbol used in schematics. However, when the same NMOS transistor is fabricated onto silicon, its physical appearance changes entirely — as illustrated on the right. This is a crucial step in the journey from design to fabrication.

In this transformation, two physical dimensions become especially important:

- Width – measured in the direction parallel to the gate.
- Length – measured in the direction perpendicular to the gate, representing the distance between the source and drain.

For example, in the illustration provided:

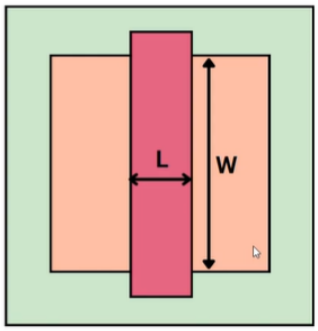
Width = 1 micrometer (μm) and Length = 0.18 micrometer (μm)

To give you context, 1 micrometer = 10^{-6} meters, or one-millionth of a meter — a scale that highlights just how incredibly small and precise these structures are

In the diagram, a thin oxide layer lies just beneath the gate, below the oxide is the substrate, which serves as the base material. Near both the source and drain regions, you'll observe N+ doped areas. These N+ regions are heavily doped to enhance conductivity — a topic we'll delve into shortly.

So the symbol that we draw on paper or on a computer is actually a very tiny 3D structure on the chip, whose dimensions are at the micrometer level. This is the foundation of chip fabrication.

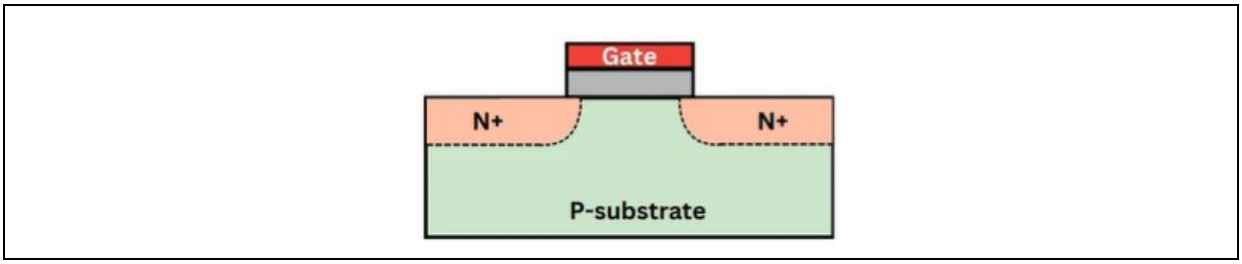
Before this 3D structure is built on silicon, it first exists in the form of a 2D layout — a crucial step in circuit design.



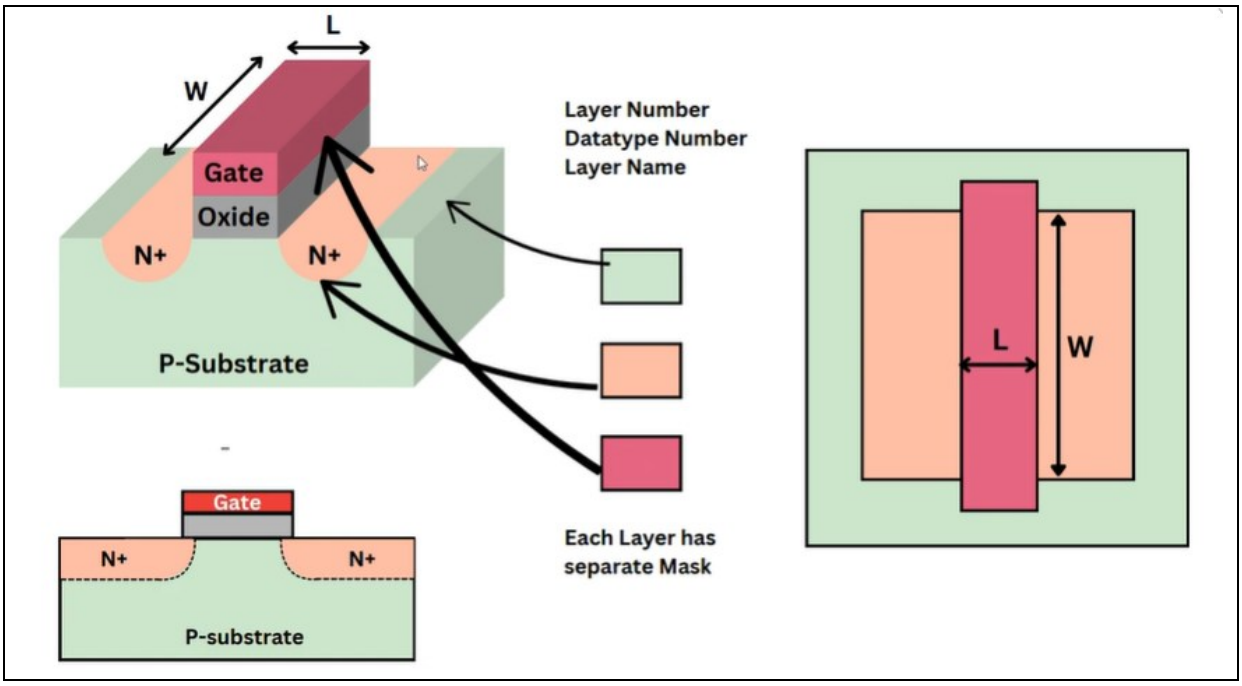
This layout is created on a computer using tools such as the Icurate Tool, available within our Learning Management System (LMS). The Icurate Tool allows you to design the transistor's physical structure with precise control over dimensions and layer definitions.

Once your design is complete, the layout is saved in a specialized format **called GDS (Graphic Data System)**. The GDS file is an industry-standard format used for chip fabrication, and it contains all the necessary geometric and layer data required to manufacture the chip.

The image shown below is a front view of the layout.



In this view, you'll observe three different colored rectangles — one green, one orange, and one red. These represent three different layers. Each layer has a name, a layer number, and a unique data type number.



During fabrication, individual masks (or dies) are created for each of these layers. These masks are used in the photolithography process to etch or deposit materials on the silicon wafer.

For example:

- The green mask defines the substrate a
- The orange mask creates the N+ doped regions.
- The red mask forms the gate structure.

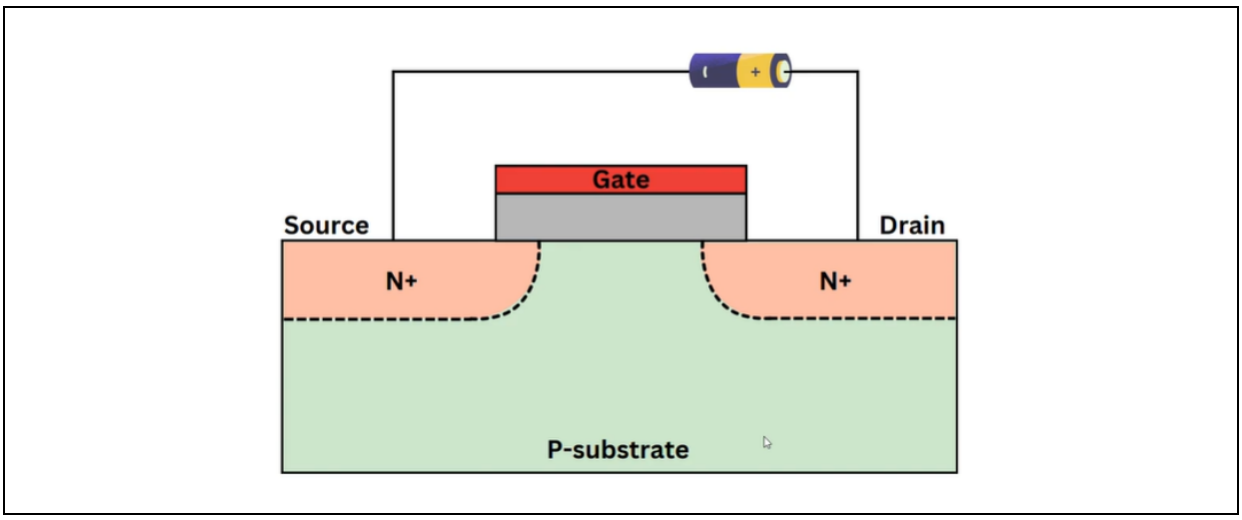
Using these layers together, a complete 3D NMOS structure is realized on silicon.

Now observe that when you look at this layout in 2D view, you will see width and length. If you're given a task like, "Create a transistor with 2 micrometer width," then in the Icurate tool, you can set the width of that rectangle to 2 micrometers — and its length will be whatever you draw vertically. That becomes the transistor's length.

Functionality

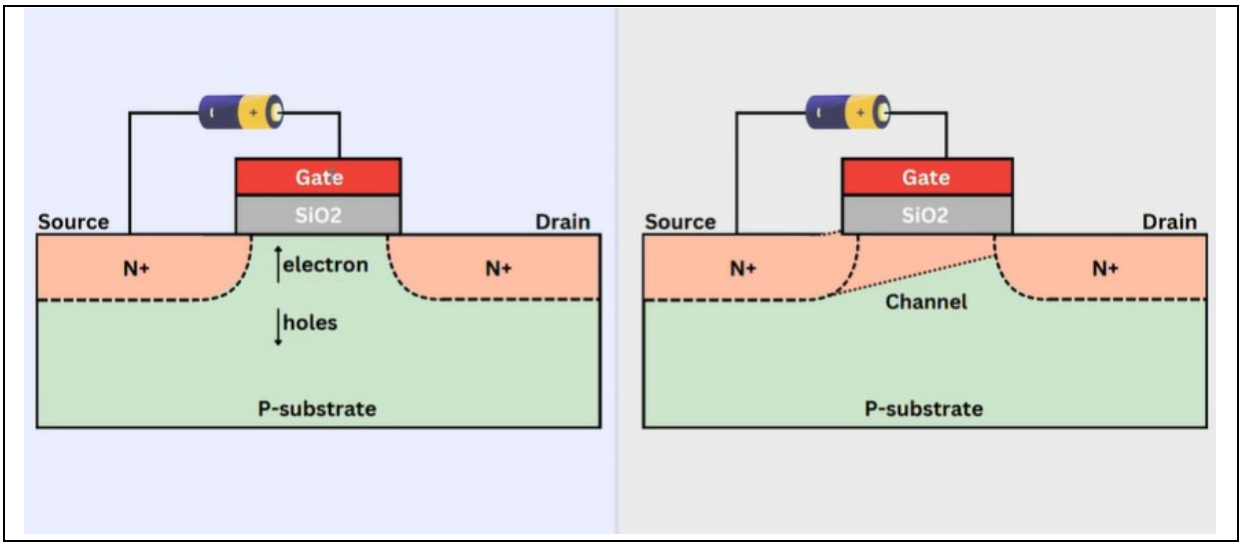
Let us now discuss the core functionality of NMOS and PMOS transistors — why they are designed in this particular way, and how they actually work within a circuit. As introduced in the previous chapter, NMOS and PMOS transistors function like switches. When a specific logic signal (such as logic 0 or logic 1) is applied to the gate terminal, the transistor either turns ON or OFF.

The gate is the control terminal that determines whether or not current will flow. For example, when a logic 1 (i.e., high voltage) is applied to the gate of an NMOS transistor, it turns ON — allowing current to pass through. Conversely, when logic 0 is applied, the NMOS remains OFF, and no current flows.



This leads us to an important question — from where to where does the current flow in an NMOS transistor?

The answer: Current flows from the Source to the Drain. However, this flow can only occur if there is a potential difference (i.e., voltage difference) between the Source and Drain terminals. To establish this potential difference, a voltage source (such as a battery) is connected across these terminals.



But simply applying voltage across Source and Drain is not enough. If there is no signal at the gate, the transistor remains OFF. This is because no conductive path (or channel) is formed between the Source and Drain.

However, when a positive voltage (logic 1) is applied to the gate, it induces the formation of a channel underneath the gate. This channel acts like a bridge, allowing electrons to flow from Source to Drain — thereby enabling current conduction.

This process involves complex physics, which we explore in depth in our next-level course on Semiconductor Physics. For now, understand this foundational concept:

Applying logic 1 to the gate of an NMOS creates a channel rich in electrons, forming a conductive path for current to flow from Source to Drain.

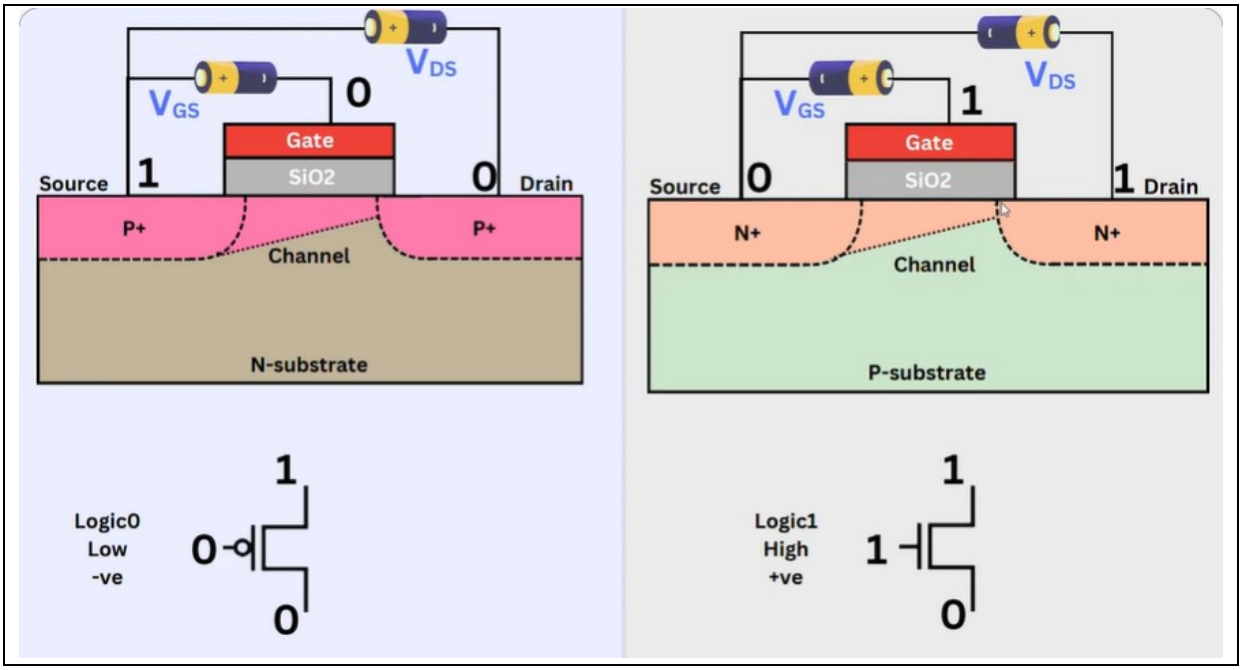
Now, pay close attention: A conductive channel — and thus current flow — will only occur when two specific conditions are met:

- There is a voltage difference between Source and Drain.
- There is a voltage difference between Gate and Source.

These two voltages are defined as:

- VDS – Voltage from Drain to Source
- VGS – Voltage from Gate to Source

Both VDS and VGS must be present for the NMOS to conduct. While we won't go into the detailed numeric values here, these will be thoroughly explained in advanced-level courses.



In the case of a PMOS transistor, the concept remains similar, but the logic is reversed. To turn ON a PMOS, you must apply logic 0 (i.e., low voltage) at the gate.

When logic 0 is applied and a potential difference exists between the Source and Drain, a channel is formed, enabling current to flow from Source to Drain.

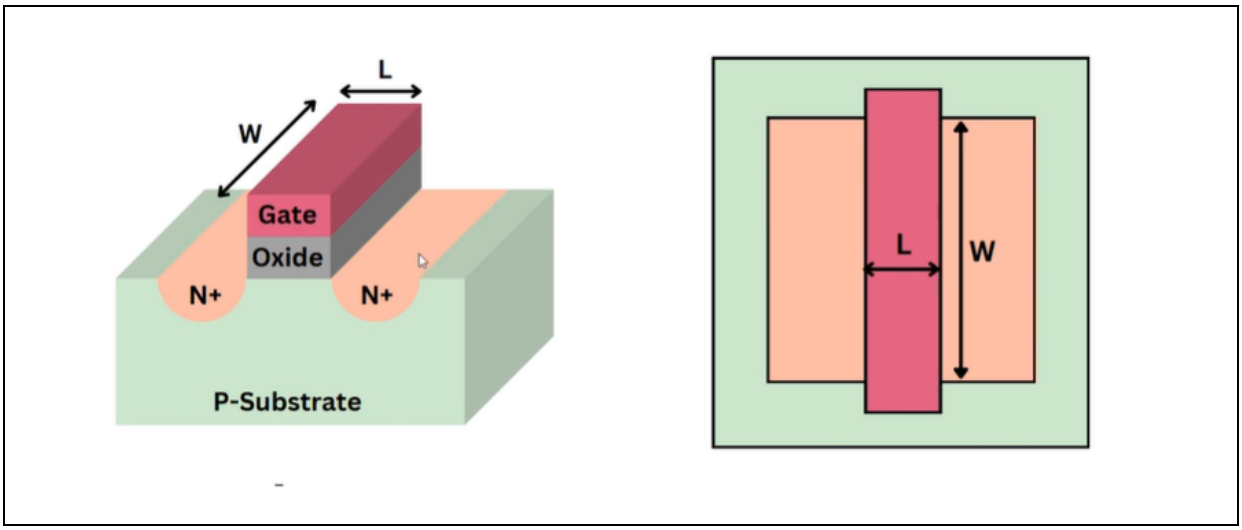
As discussed earlier through analogies, think of it like a water pipeline: Applying the right logic signal allows the water (or current) to flow by forming a passage — in this case, the channel.

To summarize:

- When you apply logic 0 or logic 1 to the gate, depending on whether it's a PMOS or NMOS, a channel either forms or doesn't.
- This channel formation is what determines whether or not current flows between the Source and Drain terminals

This is why we say that NMOS and PMOS transistors behave like switches, and their gate terminal acts as the control for turning the switch ON or OFF.

Effect of width and length variation



Let's now focus on a critical aspect of transistor design — how the Width and Length of a transistor influence its performance.

So far, we've viewed the transistor channel in a front-facing cross-sectional diagram — essentially as if sliced vertically from the front. However, in reality, when seen in three dimensions, the channel actually forms directly beneath the gate, stretching across the entire width of the transistor. This is the path through which electrons flow from Source to Drain during operation.

Here is a key insight: the wider the gate (i.e., the greater the Width), the wider the channel becomes. A wider channel offers more room for electrons to move through, resulting in lower electrical resistance and higher current flow. Therefore, when a circuit requires higher current capacity, designers use transistors with wider gates.

Now let's examine the effect of Length. The Length of the gate defines how long the channel is — essentially, the distance electrons must travel from Source to Drain.

When this length increases, electrons encounter greater resistance, which results in slower performance. On the other hand, shorter gate lengths reduce resistance, enabling electrons to travel faster and thereby making the transistor operate more quickly.

This leads us to a logical question — if we want a very fast transistor, why not simply reduce the gate length as much as possible?

At first glance, this seems like a good idea. But the answer is: No, we cannot shorten the gate length indefinitely. There are physical and technological limitations that prevent this.

Let's explore why. A transistor consists of Source and Drain terminals, which are both N+ doped regions. These regions slightly extend under the gate area to ensure proper channel formation. But if the gate length becomes too short, there is a risk that the Source and Drain regions might touch. If this happens, current will flow directly from Source to Drain without any control from the gate.

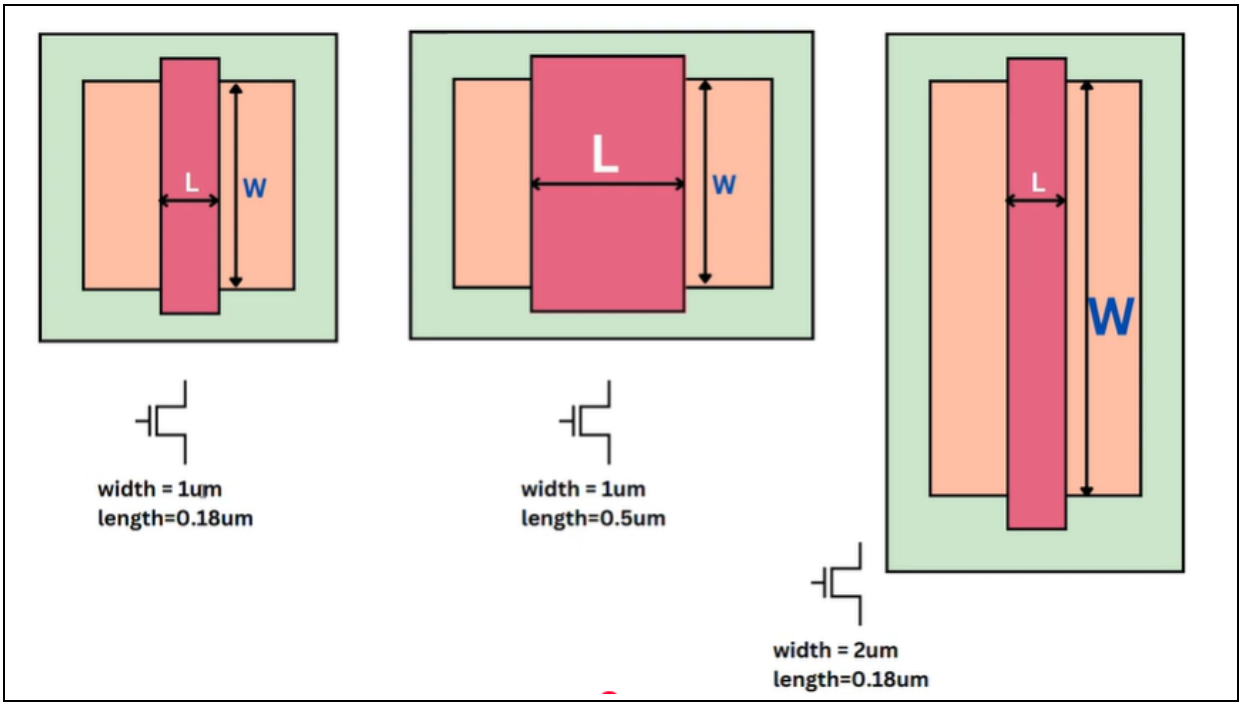
This condition leads to a catastrophic failure — the transistor loses its ability to act as a switch, which is its fundamental purpose.

To avoid such scenarios, fixed design constraints are imposed on gate dimensions. These limitations are documented in a formal set of design specifications called the **Design Rule Deck (DRC)** or the **Physical Design Kit (PDK).** These rules define minimum allowable dimensions, spacing, overlaps, and other physical parameters for transistors and other layout elements.

Therefore, whenever a Layout Design is created, it must strictly adhere to the guidelines provided in the DRC or PDK. This ensures that the transistor can be fabricated successfully and will operate reliably.

This brings us to the next important question — how can we verify that our layout complies with these design rules?

The answer lies in a process known as **Verification**. During verification, the layout is checked against all defined design rules to ensure there are no violations or errors. This step is critical to ensuring that the final chip will function as intended. The details of the Verification process will be discussed in upcoming sections.



To deepen our understanding, let's explore a few practical examples that illustrate how transistor dimensions — specifically Width and Length — impact performance.

In the first example, we consider a transistor with a width of 1 micrometer and a length of 0.18 micrometer. This represents a standard, well-balanced configuration, commonly used in many circuits.

In the second example, the width remains the same at 1 micrometer, but the length is increased to 0.5 micrometer.

Now think carefully — which of these two transistors will operate slower?

The answer is clear: the second transistor will be slower. Why? Because its channel length is longer, which means electrons must travel a greater distance from Source to Drain. This extended path introduces higher resistance, resulting in slower current flow, and ultimately, a slower transistor.

Now let's examine the third example. In this case, the length remains the same as the first example (0.18 micrometer), but the width is doubled to 2 micrometers. Compared to the first transistor, this design provides a wider path for current. As a result, resistance decreases significantly, allowing more current to flow in less time. This translates to a faster-operating transistor.

These three cases clearly demonstrate a fundamental principle:

- Increasing Length → Increases Resistance → Slower Transistor
- Increasing Width → Decreases Resistance → Faster Transistor

Now, a natural question follows — why do we vary transistor width at all?

The answer lies in the requirements of the circuit being designed.

You do not use a single, uniform type of transistor throughout a chip. Depending on where a transistor is placed and what role it plays, its performance characteristics may need to be adjusted. Sometimes, fast switching is necessary — such as in critical timing paths — where the transistor must quickly respond to changes in input. In such cases, the transistor is made wider to increase current flow and reduce switching delay.

In other parts of the circuit, however, slower switching is desirable — to prevent noise, avoid abrupt transitions, or ensure signal integrity. In those cases, the transistor is designed narrower, which introduces controlled resistance and smoothens the signal transition.

This design flexibility plays a vital role in managing timing delays and signal transitions, as discussed in earlier chapters. By carefully adjusting Width and Length, designers achieve the desired speed and behavior of each transistor within the larger system.

Technology Node and Gate Length

Now let's focus on another important concept.

You may have noticed that in earlier examples, we referenced a gate length of 0.18 micrometer. This value is not arbitrary — it represents the minimum gate length that can be reliably manufactured using a specific technology. That is why this particular process is known as 0.18 micrometer technology, or more commonly, 180 nanometer (180 nm) technology.

0.18 micrometer = 180 nanometer (nm)

In the world of chip design, we typically refer to a technology node by its minimum feature size — in this case, the minimum gate length. So, a process that can produce a 180 nm gate is called 180 nm technology.

As semiconductor fabrication technology has advanced, we have been able to shrink gate lengths even further, enabling faster, more power-efficient chips. The progression of technology nodes has followed a roadmap such as:

180 nm

90 nm

65 nm

45 nm

32 nm

28 nm

... and today, we've reached 2 nm and 3 nm technologies.

Each new node represents a significant leap in precision, performance, and complexity. However, for educational and academic purposes, 180 nm technology remains widely used.

It is considered ideal for teaching because it provides a clear, understandable view of the fundamental concepts involved in semiconductor design, without the overwhelming complexity and cost associated with advanced nodes. This makes it a powerful tool for students to build a strong foundation in chip design, layout, and fabrication principles.

Making a Silicon Chip

Now let's explore how the intricate 3D structures we've studied are physically constructed on silicon. As mentioned earlier, the entire process of chip fabrication takes place on a silicon wafer — a thin, flat substrate that forms the foundation of all integrated circuits.

But before we dive into how the chip is fabricated, it's important to understand what exactly a silicon wafer is, and how it is made.

Silicon Wafer

The journey begins with the production of a high-purity silicon ingot. An ingot is essentially a large, solid cylindrical block of pure silicon. It resembles a long wooden log or metallic pipe, but instead of wood or metal, it is composed of crystalline silicon. This purity is of paramount importance because even the slightest impurity at the atomic level can affect the electrical behavior of the final chip.

To achieve the required purity, a sophisticated process called the **Czochralski process** is often used. In this process, a seed crystal of silicon is slowly dipped into

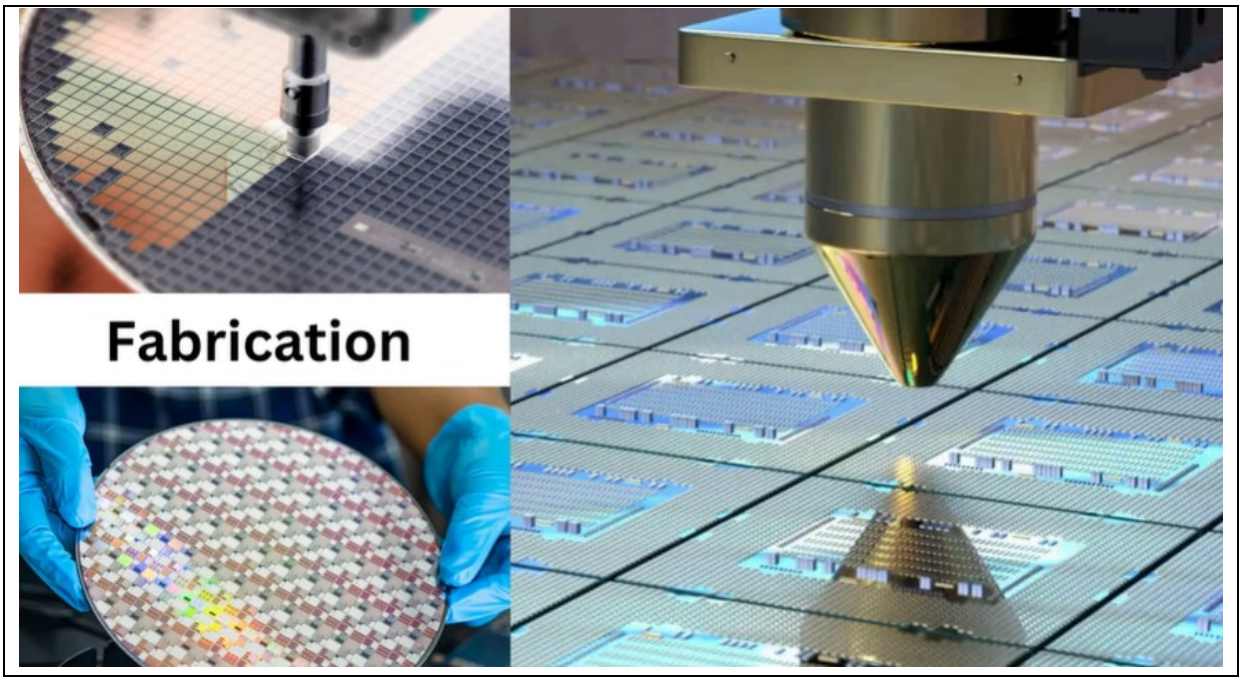
molten silicon and rotated while being pulled upward, allowing a large single crystal (monocrystalline silicon) to grow.

Once the silicon ingot is formed and cooled, it is sliced into ultra-thin, perfectly circular discs using high-precision diamond saws. These thin discs are what we refer to as silicon wafers.

Each wafer is meticulously polished to create an incredibly smooth and flat surface, which is essential for the subsequent photolithography and deposition steps in chip fabrication. The resulting wafers are typically hundreds of millimeters in diameter (commonly 200 mm or 300 mm) but only a fraction of a millimeter thick.

These highly uniform, flat wafers serve as the base upon which millions — or even billions — of microscopic transistors and interconnects are built layer by layer, ultimately forming a fully functional silicon chip.

Fabrication



Once the silicon wafer is prepared, the next critical phase is fabrication — the process by which transistors and other essential circuit elements are constructed directly onto the wafer's surface.

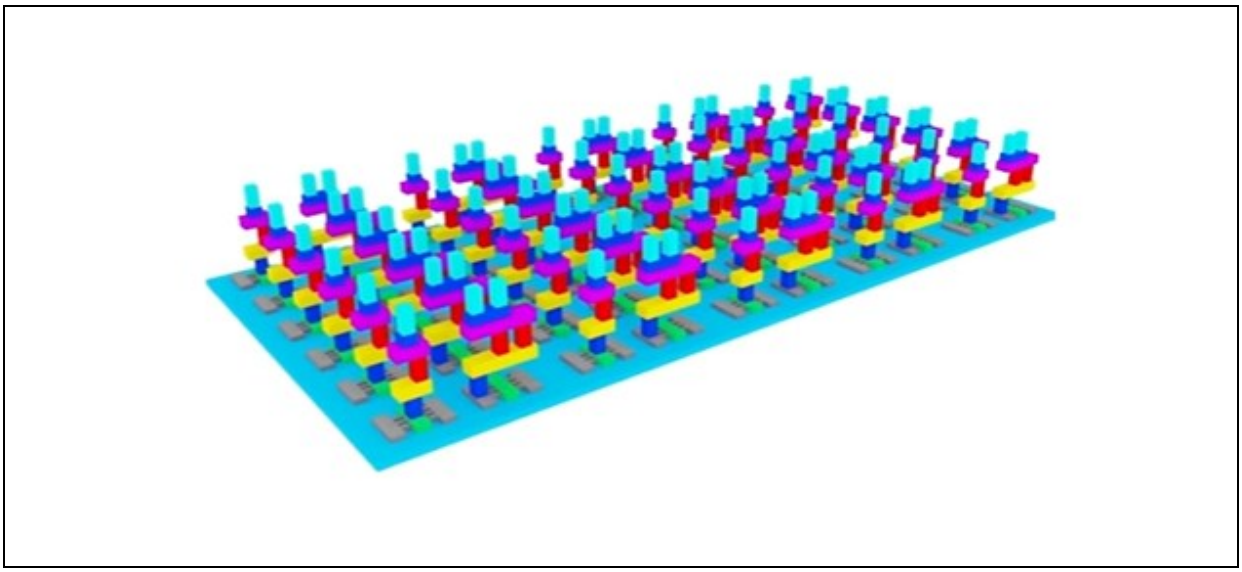
At the outset, the wafer appears as a smooth, flat disc, typically dark grey or black in color. It may look plain and unremarkable to the naked eye. However, during fabrication, an extraordinary transformation takes place. Using a series of meticulously controlled steps, millions to billions of transistors are created on this flat surface, each occupying an incredibly small footprint.

A large portion of these transistors are NMOS (n-type metal-oxide-semiconductor) and PMOS (p-type metal-oxide-semiconductor) types. Together, they are combined

to form CMOS (Complementary Metal-Oxide-Semiconductor) circuits — the building blocks of virtually all modern digital electronics.

After fabrication is complete, the once-flat wafer now resembles something akin to a miniature cityscape when viewed under a microscope. Each tiny square or rectangular area represents a microchip, densely packed with intricate pathways, transistors, and logic gates — all layered with nanometer precision

3D Structure



When we magnify the surface of a silicon wafer, we begin to see incredibly small 3D structures formed on it — structures so tiny that they are invisible to the naked eye. Observing them requires the use of high-powered instruments such as optical or electron microscopes.

In educational materials or design software, these structures are often illustrated using color-coded layers to help learners distinguish between different components and understand their specific roles. However, in reality, these layers are not colorful. Instead, they typically appear in monochromatic tones, often grayish, metallic, or dull in appearance, due to the materials used in their construction (such as metal, silicon, and oxide).

If you examine a real 3D view of the fabricated wafer, it often resembles a miniature multi-story building. Much like an architectural structure with various levels, the silicon wafer comprises multiple stacked layers, each serving a vital function in the chip's operation.

- Some layers are responsible for creating the transistors, the fundamental switching elements of digital circuits.
- Others serve as interconnects, forming conductive pathways that link different components and allow signals to travel across the chip.

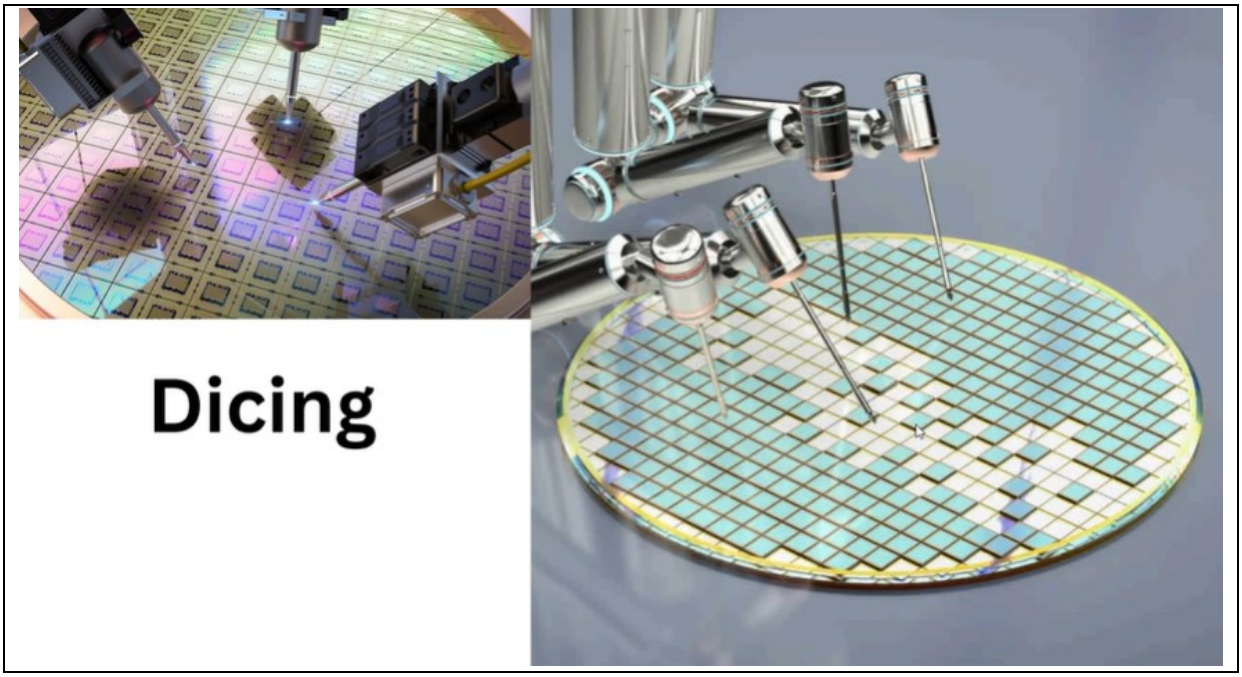
- Additional layers act as insulators, ensuring electrical isolation between circuits and preventing unintended current flow or short circuits between overlapping layers.

Now that the entire chip is ready, let's carefully understand what happens next.

If you closely observe the whole silicon wafer, you'll see many small rectangular or square-shaped patterns on it. In fact, each square is a complete chip in itself — that is, a full electronic system.

These chips may appear small in size, but they contain millions — or even billions — of transistors inside them. These transistors together form a computer's processor, a mobile phone's memory, or become part of some other electronic device.

Dicing



The next step in the chip fabrication process is known as Dicing.

Dicing involves cutting the entire silicon wafer into small, square-shaped units, where each square represents an individual chip. This process is carried out using high-precision equipment, as even the slightest error or defect during cutting can render a chip non-functional.

However, the process doesn't end with dicing. The most critical stage follows — Testing and Packaging.

Testing and Packaging

Testing is a critical stage where each individual chip is meticulously examined to ensure it functions as intended.

Given that even a single defective chip embedded within a mobile phone, computer, or any other electronic system can lead to a complete malfunction, rigorous testing is essential. Specialized testing machines are used to detect electrical or functional defects. Only those chips that successfully pass all test parameters proceed to the next stage.

Following testing is the process of Packaging.

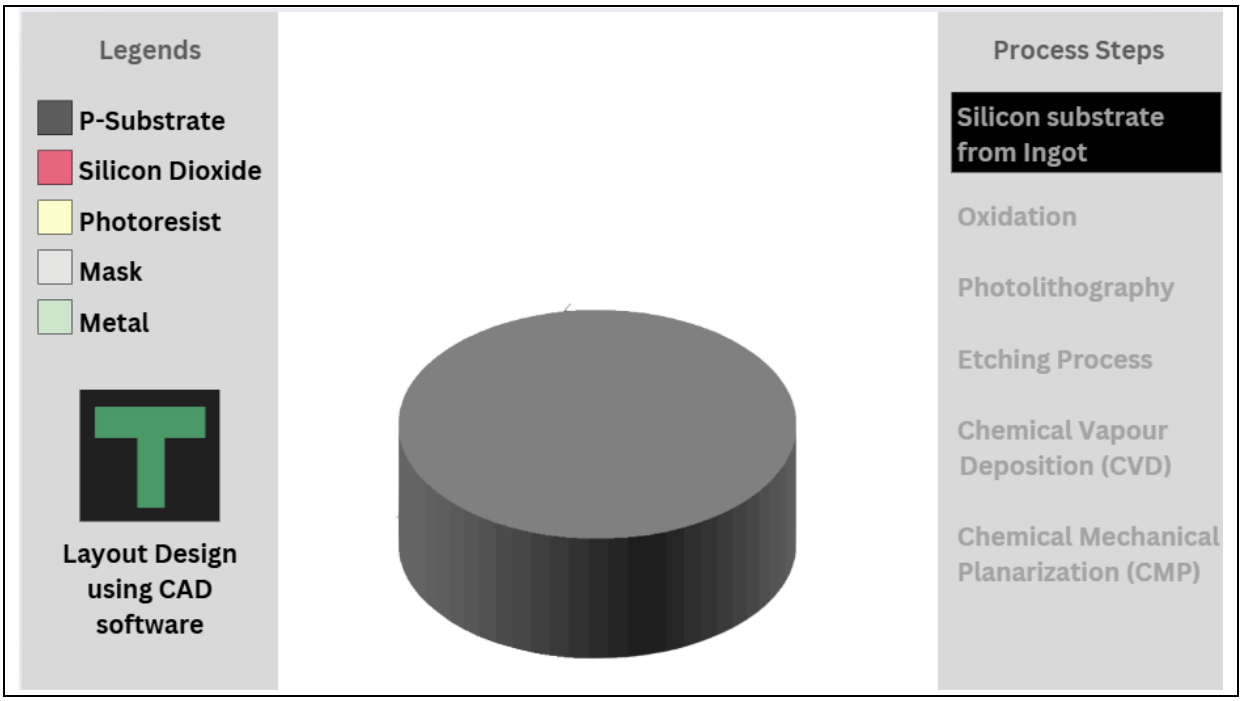
The final chip you see in the market — typically encased in a hard, black, plastic-like shell — is the result of this packaging stage. Packaging serves two vital purposes: protection of the delicate silicon die from physical and environmental damage, and providing electrical contacts that allow the chip to be easily integrated onto a circuit board or motherboard.

Inside this external casing lies the actual tiny silicon chip, fabricated earlier on the wafer. While the outer package may appear large and robust, the real working element is an extremely small and precisely engineered structure.

This entire journey — from designing the chip, fabricating it on a wafer, dicing it into individual units, testing each one for quality, and finally packaging it — culminates in a product that is assembled into larger systems. These systems form the backbone of today's electronics — found in everything from laptops and smartphones to satellites, aircraft, and industrial machines around the world.

Fabrication Process

Silicon substrate

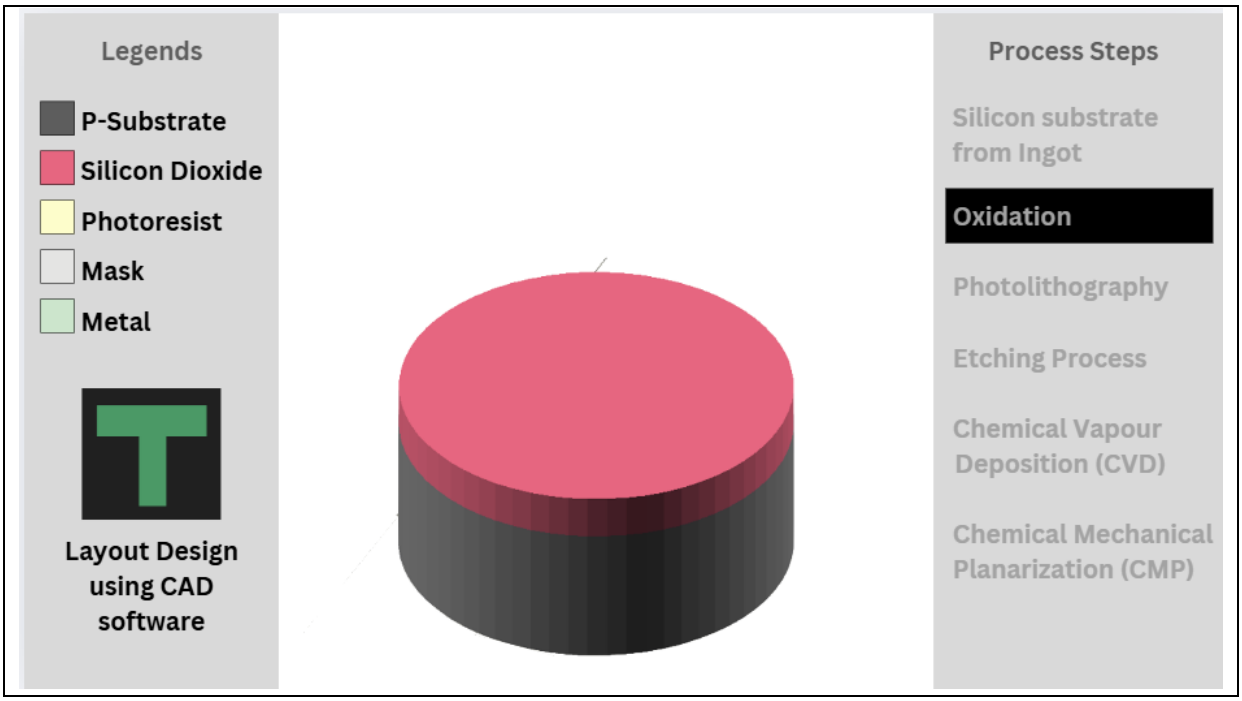


Let us now closely examine how the fabrication process is carried out on a silicon wafer.

To begin any integrated circuit design, a silicon wafer is essential. This wafer is a thin, precisely cut circular disc, sliced from a highly pure silicon ingot — a large cylindrical block of monocrystalline silicon.

The dimensions of the wafer can vary based on the manufacturing scale and design requirements. To simplify our understanding, we'll use a T-shaped layout in our explanation. It is important to note that this T-shape is purely illustrative — real circuit layouts are often far more intricate, with patterns that can be rectangular, plus-shaped (+), or highly complex geometries, depending on the circuit's functionality.

Oxidation



We begin the fabrication process with a silicon wafer, which serves as the foundational substrate for building integrated circuits.

The first step in this process is known as Oxidation.

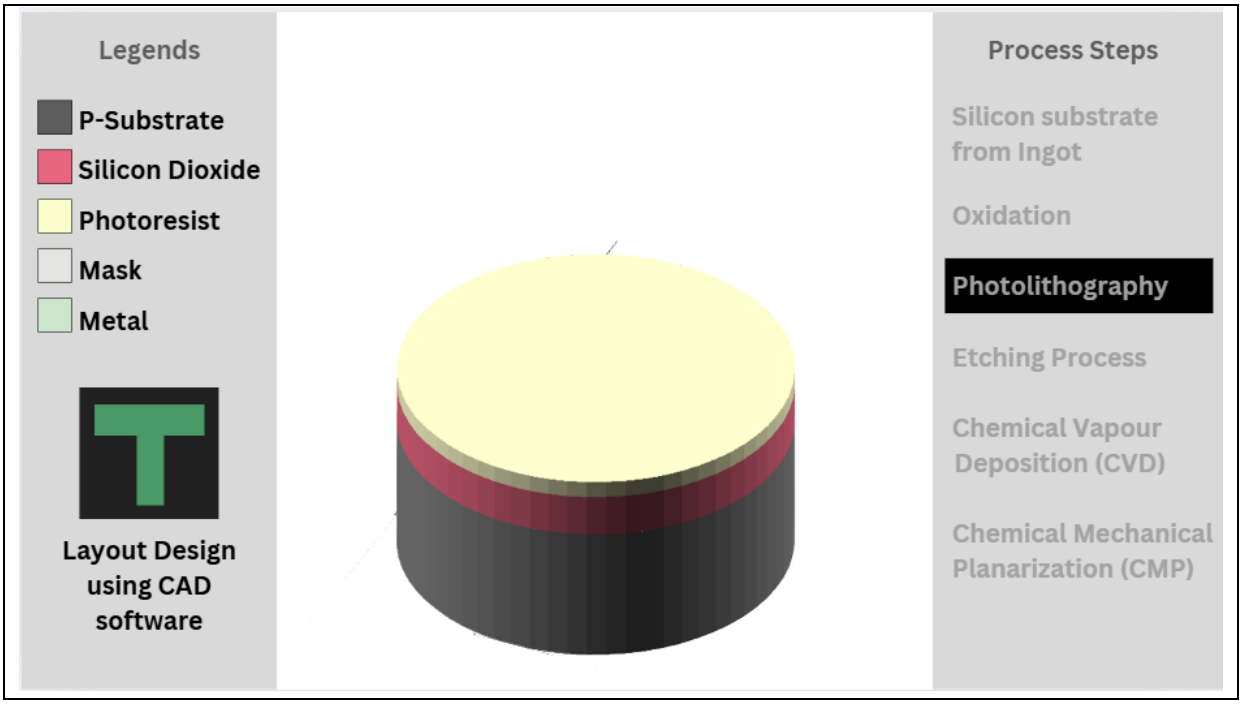
During oxidation, a thin layer of silicon dioxide (SiO_2) is thermally grown on the surface of the wafer. This oxide layer is essential, as it acts as a base for defining and isolating various regions during subsequent steps in transistor fabrication.

The silicon dioxide layer is later patterned, etched, or modified to create precise geometries required for transistor operation. It also functions as an electrical insulator and plays a key role in the formation of the gate dielectric.

In many transistor cross-sectional diagrams, the region beneath the gate represents this critical silicon dioxide layer. Its uniformity and quality directly impact the performance and reliability of the transistors built on top of it

Photolithography

Photoresist



After the formation of the silicon dioxide layer, the next step involves applying a photoresist layer on top of the wafer.

Photoresist is a light-sensitive chemical material specifically designed to react when exposed to ultraviolet (UV) light. Its behavior is straightforward:

- Regions exposed to light become soluble and can be removed during the development process.
- Regions not exposed to light remain intact, acting as a protective mask for the underlying layers.

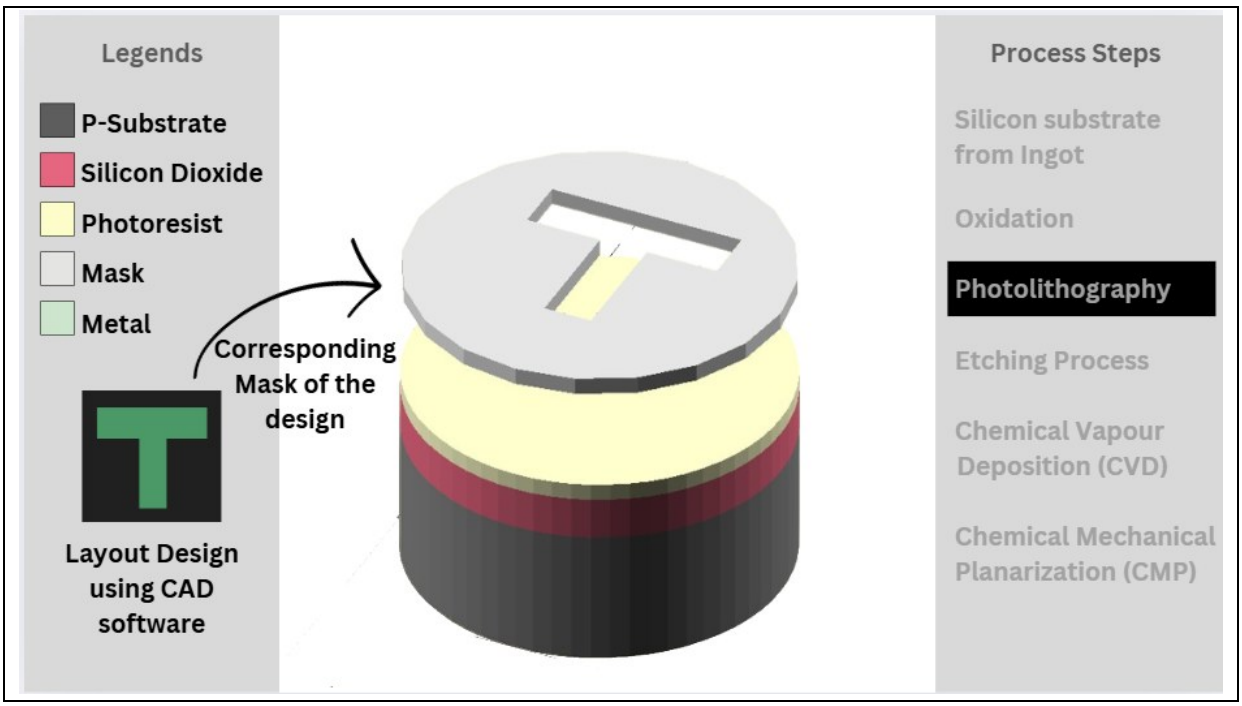
At this point, a natural question arises: How are such incredibly small and intricate patterns created?

The answer lies in a process called Photolithography — a core technology in semiconductor manufacturing that enables the creation of millions to billions of transistors within an extremely small area.

As previously discussed, even a 1GB memory chip can contain approximately 1.08 billion transistors. Achieving this level of precision and density is made possible through advanced photolithographic techniques.

This entire fabrication process is deeply rooted in chemical engineering. Every layer on the wafer — whether it's silicon dioxide, photoresist, or doped regions — is formed and manipulated using highly controlled chemical processes, allowing for the precise creation of complex circuit structures at the nanometer scale.

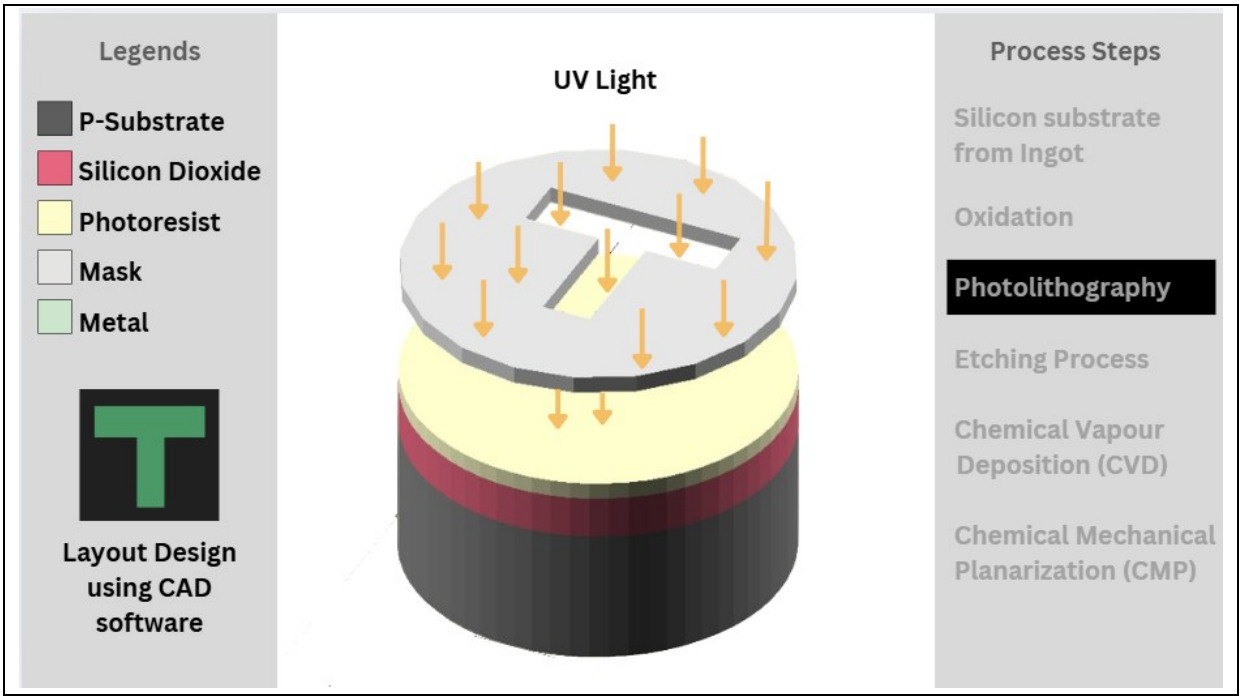
Mask



Let us now consider the creation of a T-shaped design on a silicon wafer.

The process begins by designing the T-shape layout using computer-aided design (CAD) tools. Once the layout is finalized, a photomask—commonly referred to as a mask—is fabricated based on the design.

You can think of a mask as a high-precision stencil. Similar to cutting a T-shape into a sheet of paper, the mask contains transparent sections where light is allowed to pass through, and opaque sections where light is blocked. These transparent areas correspond exactly to the intended layout pattern.

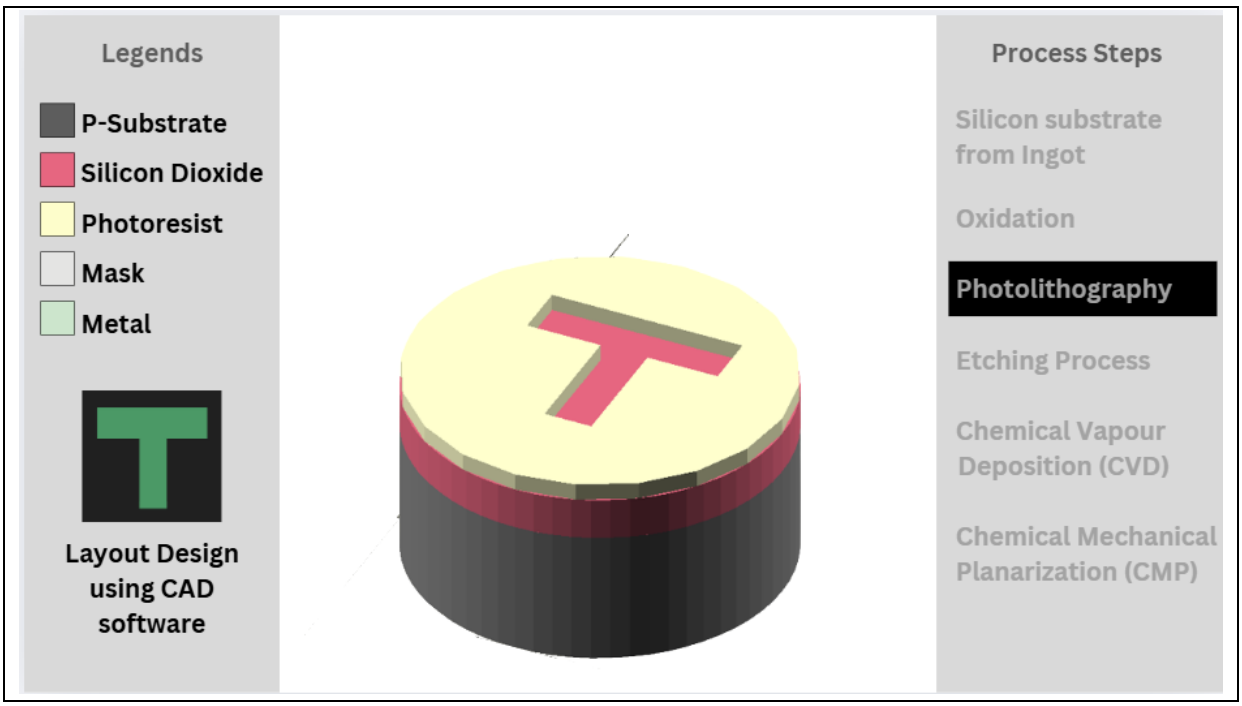


With the mask prepared, the next step is known as the Exposure Process. During this step, intense ultraviolet (UV) light is projected onto the wafer through the mask.

Because the mask blocks light in all areas except the defined T-shape, only the T-shaped region of the photoresist layer on the wafer gets exposed to UV light. The remaining regions are protected from exposure.

It's important to understand a key nuance here: light, when passing through corners or fine edges of the mask, does not create perfectly sharp boundaries. This is due to diffraction effects — a phenomenon where light bends slightly around the edges. As a result, even a geometrically sharp T-shape may appear with slightly rounded corners on the wafer.

This limitation is well known in semiconductor manufacturing and is managed through advanced techniques in more advanced fabrication processes.

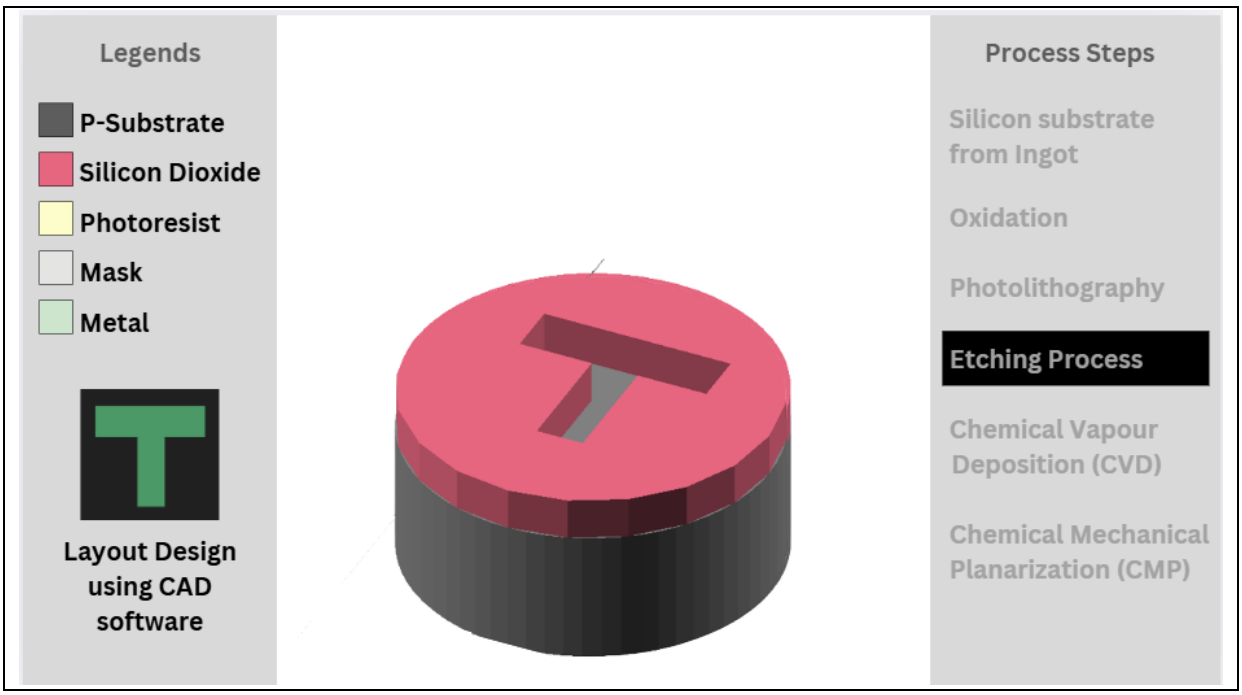


Following exposure, the wafer undergoes a development process:

- Exposed areas of the photoresist become soluble and are dissolved away using a developer solution.
- Unexposed areas remain intact, effectively forming a mask for the next processing steps.

What we now have is a T-shaped opening (or trench) in the photoresist layer. This opening exposes the underlying silicon dioxide layer or silicon substrate, depending on the process stage.

This region can now be further processed — for instance, the silicon dioxide beneath can be etched away, or the area can be filled with metal to form electrical connections.

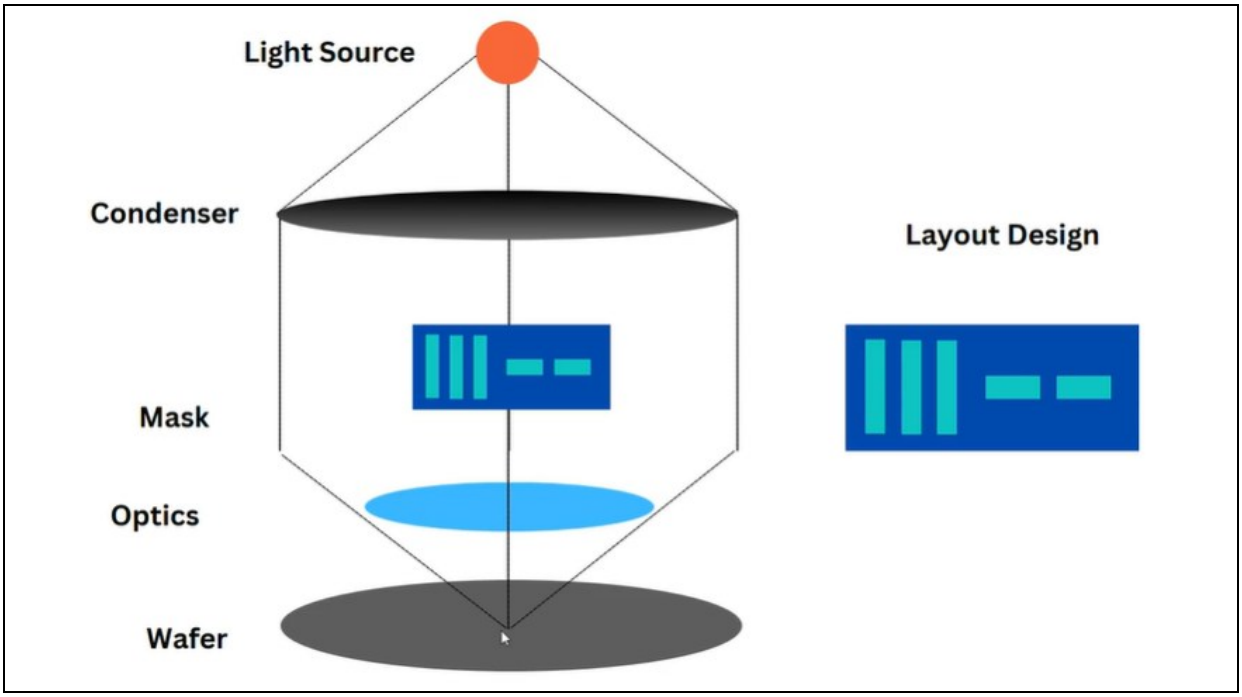


This entire process is known as Photolithography — a cornerstone of modern semiconductor fabrication.

After photolithography, the exposed areas of the wafer may undergo additional steps such as etching, doping, metal deposition, or insulation, depending on the specific circuit structure being built.

By using light with extremely short wavelengths, and focusing it with high-precision optical lenses, engineers are able to define incredibly fine features — allowing millions to billions of transistors to be constructed on a single chip, all within a remarkably compact area.

Patterning



Creating a mask at extremely small dimensions is a significant challenge. In practice, photomasks are intentionally designed and fabricated at a much larger scale. This larger size makes it easier for engineers to make fine adjustments and maintain precision during the mask-making process. Producing a mask directly at nanoscale resolutions would not only be prohibitively expensive but also highly impractical with current manufacturing techniques.

This brings us to an important question:

If the mask is large, how are such small, highly precise patterns formed on the wafer?

The answer lies in the optical projection system used in the photolithography process.

Once ultraviolet (UV) light passes through the large-scale mask, it enters an optical component known as a condenser. The condenser's role is to collimate, align, and focus the incoming light rays, ensuring that the illumination is both uniform and controlled.

After passing through the condenser, the light is directed into a high-precision lens system. This is where the actual optical reduction takes place.

- The lens system reduces the size of the mask pattern
- This reduced image is then projected onto the surface of the wafer coated with photoresist.
- The level of reduction determines how finely the pattern is etched onto the wafer, allowing us to create micro- and nanoscale features despite starting with a much larger mask.

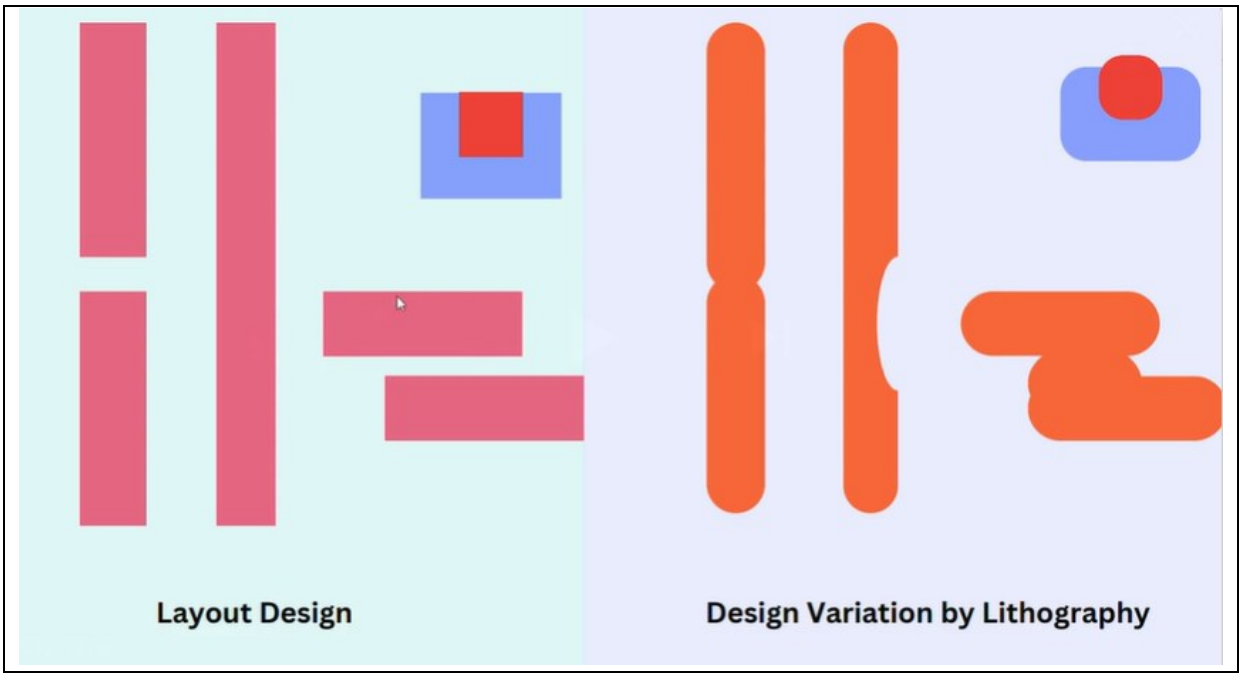
This method enables the formation of incredibly small structures while retaining the accuracy and ease of working with larger mask layouts.

The full sequence is as follows:

Mask ➔ Condenser ➔ Lens ➔ Focused Light ➔ Wafer ➔ Pattern Formation ➔ Chemical Reaction in Photoresist ➔ Final Design on Wafer

This entire process is the foundation of modern semiconductor fabrication. It is through this optical scaling technique that we are able to manufacture millions to billions of transistors, all within the confines of a tiny silicon chip, enabling the remarkable performance of today's electronic devices.

Design Variations



This diagram beautifully illustrates why there is a difference between the Layout Design and the actual Fabricated Design, and why Design Rules (DRC Rules) are essential. Let's understand this in a structured and professional manner.

When we design a chip, the first step is to prepare the Layout Design. This layout typically consists of 2D rectangles, lines, and various shapes — as shown on the left side of the diagram. You can see rectangles in different colors, such as red and blue. On the computer, this layout appears perfectly clean and sharp, and we expect it will be replicated exactly the same way on the silicon wafer.

But in reality, this is not what happens.

As soon as the layout passes through the Photolithography Process, it gets distorted at several points due to the effects of the mask, light, lens, and chemical reactions.

You can observe this on the right side of the diagram, where the Design Variation by Lithography is illustrated.

Corner Rounding Effect

When light falls on the wafer through the mask, the rectangles that appear sharp and clean in the layout design do not come out the same after fabrication.

This is due to the wave nature of light. As light passes through the cut-out regions of the mask, slight diffraction occurs at the edges.

As a result, the sharp 90° corners designed in the layout become rounded corners in the final fabricated structure.

👉 In the diagram:

Notice the two long rectangles on the left — they are designed with straight, sharp edges. But on the right, those same corners appear rounded.

This is the Sharp → Round effect.

Short Circuit

Sometimes, in the layout, two rectangles are placed very close to one another to optimize space. However, during photolithography, due to light diffraction and photoresist sensitivity, these shapes may unintentionally merge.

👉 In the diagram:

On the left, there is a visible gap between the two rectangles. But on the right, the same shapes appear connected.

This can result in a short circuit, potentially making the circuit non-functional.

Width variations – opens and shorts

During photolithography, the light exposure effect varies between horizontal and vertical shapes, because light intensity and focus are not uniform in all directions.

- Horizontal shapes may become slightly wider, due to greater light spread.
- Vertical shapes may become narrower, due to less light at the edges or uneven removal of photoresist.

👉 In the diagram:

On the left is a line with uniform thickness.

On the right, the horizontal line is thicker, and the vertical line is thinner.

These width variations directly affect the resistance and capacitance of the circuit, impacting overall performance.

Design overlaps

When shapes are placed edge-to-edge in the layout (i.e., one touches another exactly at its boundary), the photolithography process may cause the shapes to overlap or misalign.

👉 In the diagram:

Look at the red and blue rectangles at the top.

In the layout, they are clearly separated.

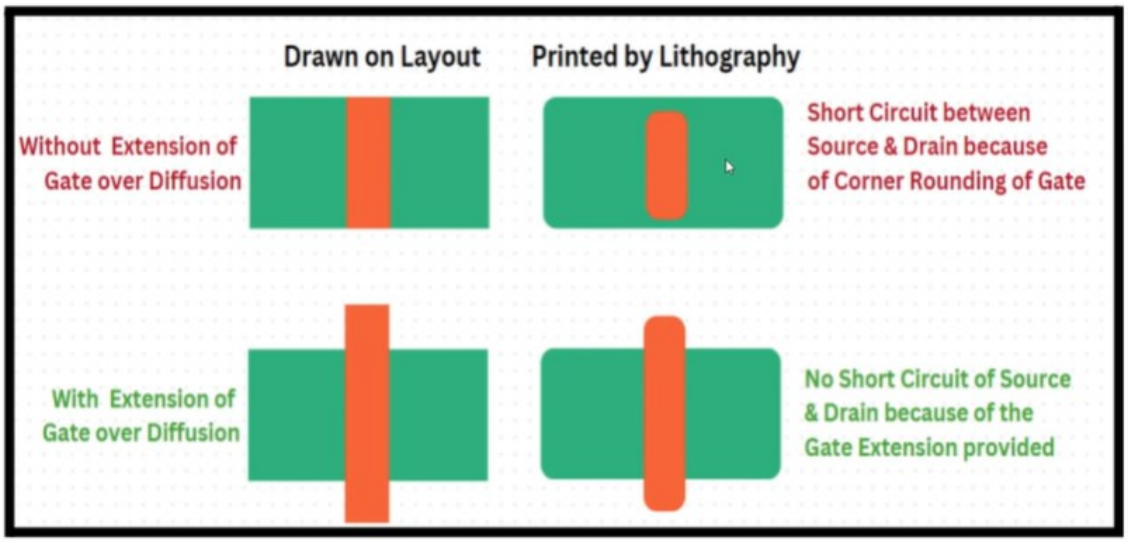
But in the fabricated version, the edges do not remain distinct and appear misaligned.

These overlaps can result in functional errors, causing logic gates or memory cells to behave incorrectly.

It is impossible to manually verify all such conditions, because a single technology node can involve thousands of DRC rules. Hence, specialized software tools, known as DRC Tools (Design Rule Check Tools), are used. These tools analyze the layout on a computer and automatically highlight rule violations.

By using tools like the Icurate Tool, you can learn these rules practically and gain hands-on experience in correcting Design Rule Violations. Dedicated chapters are available to guide you through this process, enabling you to perform it on your own.

Importance of Gate cap



Let us now explore another important example related to transistor fabrication.

A typical transistor consists of three fundamental components — the Gate, Source, and Drain. It is essential to maintain precise spacing and structure among these elements to ensure correct transistor operation.

You may have observed that in many designs, the Gate is intentionally extended slightly forward. But why is this done?

The reason ties back to the principles discussed earlier — during photolithography, as light passes through the mask, it undergoes diffraction, causing the corners of shapes to become rounded.

If the Gate is designed exactly edge-to-edge between the Source and Drain, then due to light spreading and rounding effects, its printed shape may become distorted. This distortion can lead to the Source and Drain unintentionally connecting, resulting in a short circuit.

This poses a critical issue, because the Gate plays the key role in switching the transistor ON and OFF. If the Source and Drain are already connected due to fabrication distortion, the transistor will never fully turn OFF, even when the Gate is inactive. This leads to constant and unintended current flow, which can severely impact the performance of the chip, and in many cases, cause permanent damage to the circuit.

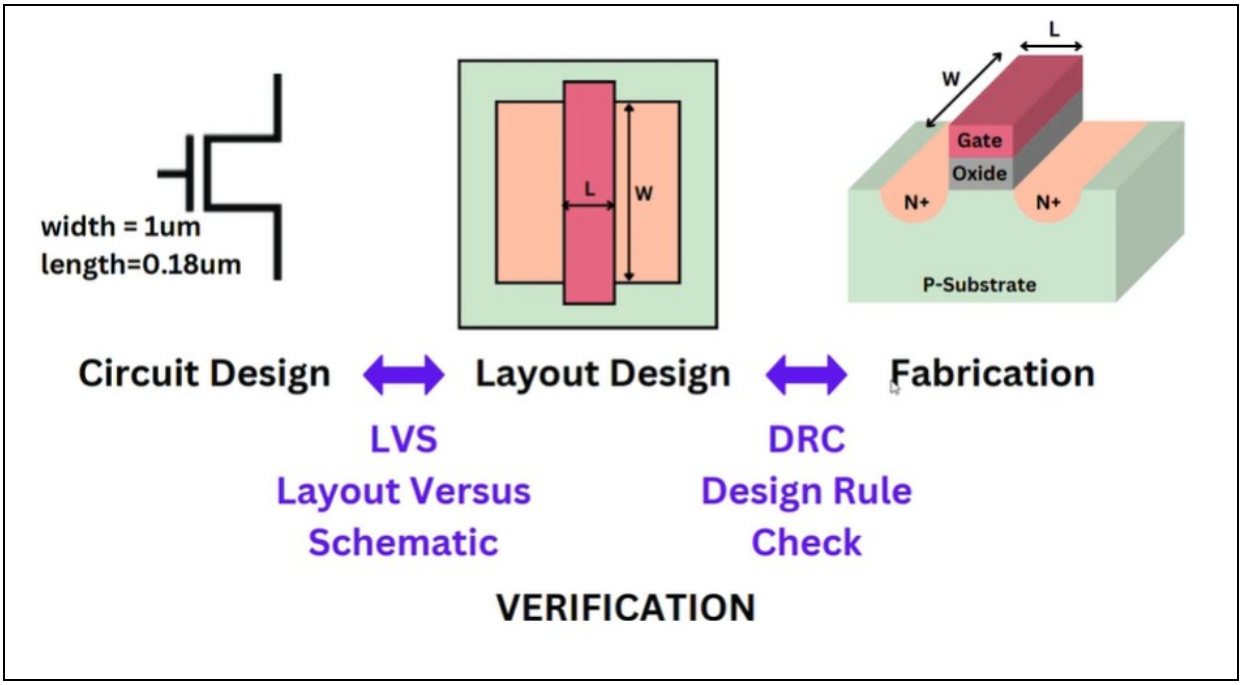
To prevent such failures, the Gate is designed to extend slightly forward. With this extension, the shape that is ultimately formed on the silicon after photolithography ensures proper separation between the Source and Drain. This design adjustment ensures that the transistor functions correctly and reliably.

👉 This concept is illustrated clearly in the diagram:

On the side where the Gate is extended, there is a visible and accurate separation between the Source and Drain, preventing any short circuit.

On the other side, where the Gate was aligned edge-to-edge, the Source and Drain become connected, rendering the transistor non-functional.

Design Verification



Let us now consolidate everything we've learned to clearly understand the complete process of chip design and fabrication. The development of any chip primarily involves two key stages: Circuit Design and Layout Design.

Circuit Design refers to the symbolic or schematic representation of the circuit. In this phase, we use standard symbols to depict components such as NMOS and PMOS transistors, resistors, and capacitors. However, simply drawing these symbols is not sufficient. Each component must also have its properties and parameters defined—for instance, specifying the width and length of a transistor to ensure it performs correctly during simulation and in real-world implementation.

Moving to the Layout Design stage, we transition from symbolic representation to a physical one. While the schematic shows how components are connected logically, layout design defines where and how each component will be constructed on the silicon wafer. This is achieved by drawing precise 2D geometric shapes—typically rectangles or polygons—that indicate the actual dimensions, positioning, and spacing of transistors and interconnects. This layout becomes the blueprint for fabrication, as it is transferred directly onto the silicon wafer during the manufacturing process.

Layout Versus Schematic (LVS)

An important question arises at this stage: Is the layout I have created an exact match to the original circuit I designed? Or have errors been introduced in the process?

Such discrepancies are indeed possible. For instance, while the schematic might contain 1,000 transistors, it is entirely feasible that only 998 were implemented during layout, or that 1,200 were unintentionally included. If a layout with such inconsistencies is sent directly for fabrication without validation, it can lead to significant functional failures and wasted resources.

To identify and prevent these types of errors, the industry relies on a verification process known as Layout Versus Schematic (LVS). The purpose of LVS is to ensure that the layout precisely matches the original schematic design. Any mismatches—whether in connectivity, component count, or structure—are flagged by the LVS tool.

A successful LVS check confirms that the layout implementation is consistent with the intended circuit design, providing confidence that the chip will function as expected once fabricated.

Design Rule Check (DRC)

The second and more critical verification step in chip design is known as the Design Rule Check (DRC). The primary objective of DRC is to ensure that the physical layout of the chip adheres strictly to the manufacturing constraints defined by the target semiconductor technology.

These design rules are essential because they guarantee that the chip can be reliably fabricated without functional defects. A failure to meet even a single rule can render the design non-manufacturable, which is why passing the DRC is an absolute prerequisite before proceeding to fabrication.

This raises an important question: Where do these rules originate? The answer lies in the specific semiconductor technology being used. Each technology node—such as 180 nanometers or 90 nanometers—comes with its own comprehensive set of design rules. For example, in 180 nm technology, the minimum allowable transistor gate length is 180 nm. In contrast, for 90 nm technology, this minimum reduces to 90 nm.

As process nodes continue to shrink and technology evolves, these design rules become increasingly precise and restrictive. This progression enables the creation of smaller, faster, and more power-efficient chips, driving innovation across the semiconductor industry.

In Design Rule Check (DRC), there are many different types of rules that must be carefully followed during the design process.

Together, these rules ensure that the layout we are creating is suitable for fabrication and free from any technical errors.

Some basic types of DRC rules can be practiced using the Icurate Tool:

- Spacing Rule:
  When we place two different layers in a design, there must be a minimum distance between them.
  If they are placed too close to each other, they might merge during fabrication, which can lead to circuit errors.
  Therefore, a minimum spacing is defined that must be maintained between any two layers.

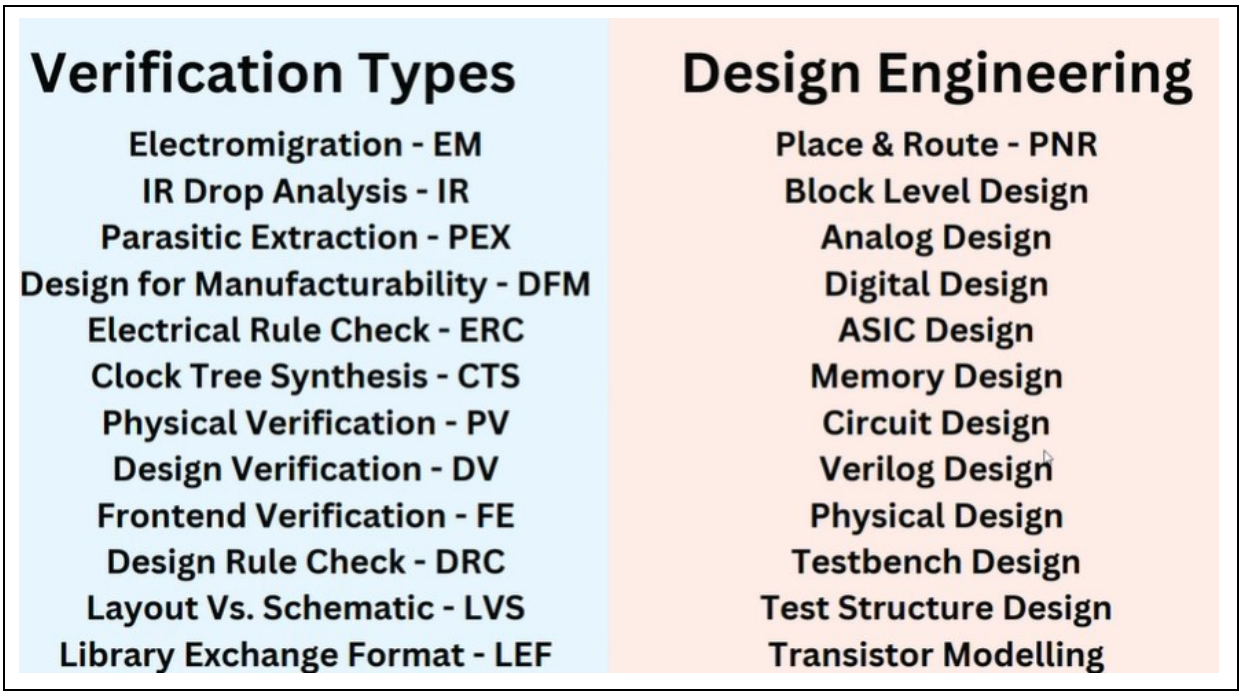
- Minimum Area Rule:
  If a shape is made too small, it might disappear during fabrication or may not form properly.
  Hence, each shape must meet a minimum area requirement to ensure it is formed correctly.

- Enclosure Rule:
  When one layer is meant to cover another layer, it must enclose it by a specified margin on all sides.
  This ensures that the resulting shape after fabrication is complete and functional.

Apart from these basic rules, as design complexity increases, advanced DRC rules also come into play. These include rules for Latch-Up, Antenna effects, Density, and many more, which are covered in advanced training.

It is not practical to check all these rules manually, which is why DRC Verification Tools are used. You can begin practicing the basic rules using the Icurate Tool, and gradually learn the advanced DRC rules to become proficient in professional chip designing.

Verification types and associated job roles



So far, we've learned about two important types of verification: LVS (Layout Versus Schematic) and DRC (Design Rule Check). However, these two verifications alone are not sufficient to fully complete a chip design. In reality, the chip design process involves multiple types of verifications to ensure the chip works correctly and is free from any errors.

For example, processes like Electromagnetic Analysis, IR Drop Analysis, and Parasitic Extraction are also essential. In addition, several other checks are performed to ensure that the chip design is not only accurate, but also functions properly and remains reliable over time.

This task is so vast and complex that a single engineer cannot handle everything alone. Each type of verification has its own specialists.

For example:

- For DRC Verification, there are DRC Experts
- For LVS, there are LVS Experts
- Engineers who build the Clock Tree are called CTS Engineers
- For DFM (Design for Manufacturability) checks, there are separate DFM Experts

This is why the chip design industry offers a wide range of job roles.

Some of the key job profiles are:

- Verilog Design Engineer — engineers who write code for digital circuits
- Physical Design Engineer — engineers who convert the circuit into a physical chip layout
- Test Bench Design Engineer — engineers who design simulations to test the circuit
- Test Structure Design Engineer — engineers who create test circuits for fabrication
- Transistor Modeling Engineer — engineers who model the behavior of transistors
- Analog Design Engineer — engineers who design analog circuits
- Digital Design Engineer — engineers who design digital circuits

This entire industry is so vast that each job role requires dedicated specialists.

These roles are highly specialized and require advanced-level training. If you want to build a career in this field, you can prepare for these roles and find great opportunities in the semiconductor industry.