| Experiment No. 3 |
| --- |
| Design a digital circuit using various logic gates using Cedar Logic Simulator |
| Date of Performance: |
| Date of Correction: |

**Aim:** To design and simulate a digital logic circuit using various basic and derived logic gates with the help of Cedar Logic Simulator.

**Objective:** To understand the working of logic gates and apply them in designing and simulating digital circuits using Cedar Logic Simulator.

**Theory:**

**Introduction to Logic Gates:**

Logic gates are the **fundamental building blocks** of digital electronics. These gates perform **Boolean algebra operations** on one or more binary inputs to produce a binary output. Logic gates are implemented using transistors in hardware but can also be simulated virtually using logic simulation software such as **Cedar Logic Simulator**.

There are two categories of gates:

1. **Basic Gates**: AND, OR, NOT
2. **Derived (Universal) Gates**: NAND, NOR, XOR, XNOR

These gates are used to build more complex components like **adders, multiplexers, decoders, flip-flops**, and ultimately **processors and memory units**.

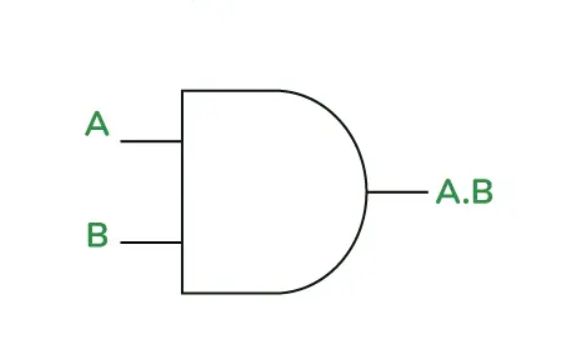
Logic gates are the fundamental building blocks of digital electronics. They perform Boolean operations on binary inputs to produce a binary output. Below are the most commonly used gates:

1. **AND Gate**

The AND gate outputs 1 only if all inputs are 1.

Logic Expression: Y = A · B

| **A** | **B** | **Y (A · B)** |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

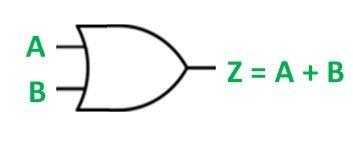


1. **OR Gate**

The OR gate outputs 1 if any input is 1.

Logic Expression: Y = A + B

| **A** | **B** | **Y (A + B)** |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

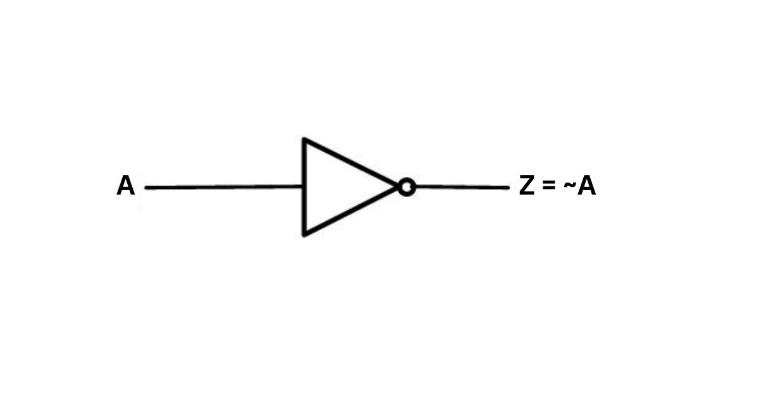


1. **NOT Gate**

The NOT gate inverts the input (0 becomes 1, and 1 becomes 0).

Logic Expression: Y = ¬A

| **A** | **Y (¬A)** |
| --- | --- |
| 0 | 1 |
| 1 | 0 |

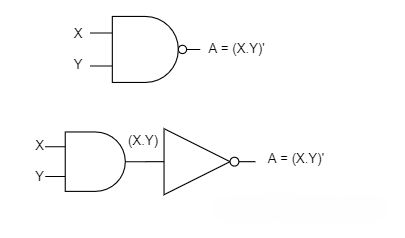


1. **NAND Gate**

The NAND gate is the inverse of the AND gate.

Logic Expression: Y = ¬ (A · B)

| **A** | **B** | **Y (¬ (A · B))** |
| --- | --- | --- |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

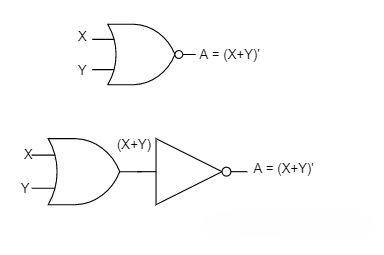


1. **NOR Gate**

The NOR gate is the inverse of the OR gate.

Logic Expression: Y = ¬ (A + B)

| **A** | **B** | **Y (¬ (A + B))** |
| --- | --- | --- |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

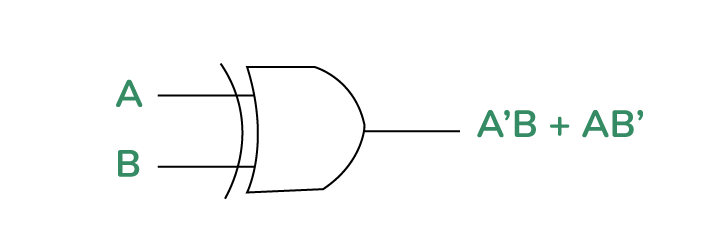


1. **XOR Gate**

The XOR gate outputs 1 only if the inputs are different.

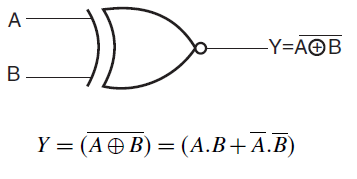
Logic Expression: Y = A ⊕ B

| **A** | **B** | **Y (A ⊕ B)** |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |



1. **XNOR Gate**  
   The XNOR (Exclusive-NOR) gate is the complement of the XOR gate. It outputs 1 only if the inputs are the same.  
   Logic Expression: Y = ¬ (A ⊕ B) or Y = A ⊙ B

| **A** | **B** | **Y (¬ (A ⊕ B))** |
| --- | --- | --- |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |



**Why Are We Using Cedar Logic Simulator?**

Cedar Logic Simulator is used in this practical to help students visually understand and design digital circuits using logic gates. It provides a simple drag-and-drop interface with real-time simulation, making it ideal for beginners. By eliminating hardware constraints, students can focus on learning logic design concepts effectively. It also saves lab time, supports experimentation, and bridges the gap between theoretical learning and practical application.

**What is Cedar Logic Simulator?**

Cedar Logic Simulator is a free, open-source, **digital logic design and simulation tool**. It allows users to design logic circuits graphically by placing logic gates, connecting them with wires, and observing real-time behavior of the circuit. It’s ideal for beginners learning digital logic design.

**Installation Steps for Cedar Logic Simulator:**

1. **Download Cedar Logic**:
   * Visit the official GitHub page:
   * Or search: “Cedar Logic Simulator GitHub” in your browser.
2. **Download the Installer**:
   * Download the .zip file or Windows installer (CedarLogicSetup.exe).
3. **Install the Software**:
   * Extract the .zip file or run the .exe file.
   * Follow the on-screen instructions to install it.
4. **Launch Cedar Logic**:
   * Open the simulator.
   * Use the toolbar to add gates, switches, and output LEDs.
5. **Begin Simulation**:
   * Build your circuit by dragging gates onto the canvas.
   * Connect them using wires.
   * Add inputs (switches) and outputs (LEDs).
   * Toggle switches to see real-time output.

**Practical Applications of Logic Gates:**

* Designing **Arithmetic Circuits**: Adders, subtractors.
* Building **Decision-Making Units**: Comparators, encoders, decoders.
* Implementing **Storage and Memory**: Flip-flops, registers.
* Creating **Control Logic** in CPUs and embedded systems.
* Used in **communication systems, robotics, AI processors**, and **IoT devices**.

**Solution:**

**Conclusion:** We designed and simulated digital circuits using various logic gates, which helped us visualize and understand how logical operations are performed in hardware.