**1. AND Gate**

* **Function:** Outputs HIGH (1) only if all inputs are HIGH (1).
* **Truth Table:**

| **Input A** | **Input B** | **Output** |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

* **Circuit Diagram**: AND gates can be implemented using diodes and a resistor in simple circuits.

A -----|>---+

|

+---[R]--- OUT

B -----|>---+

**Explanation**:

* The diodes allow current to pass through only if both inputs (A and B) are HIGH, generating an output.

**2. OR Gate**

* **Function:** Outputs HIGH (1) if at least one input is HIGH (1).
* **Truth Table:**

| **Input A** | **Input B** | **Output** |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

* **Circuit Diagram**: OR gates use diodes for input and a resistor to pull up the output.

A ----|>|---+

|

+--- OUT

B ----|>|---+

**Explanation**:

* Current flows if at least one of the diodes conducts (i.e., one input is HIGH).

**3. NOT Gate (Inverter)**

* **Function:** Inverts the input. If input is HIGH (1), output is LOW (0), and vice versa.
* **Truth Table:**

| **Input** | **Output** |
| --- | --- |
| 0 | 1 |
| 1 | 0 |

* **Circuit Diagram**: A NOT gate can be implemented using a transistor.

A ----|BJT|--- OUT

**Explanation**:

* When the input is HIGH, the transistor conducts, pulling the output to LOW.

**4. NAND Gate**

* **Function:** Outputs LOW (0) only if all inputs are HIGH (1). It’s the complement of the AND gate.
* **Truth Table:**

| **Input A** | **Input B** | **Output** |
| --- | --- | --- |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

* **Circuit Diagram**: NAND gates combine an AND gate followed by a NOT gate.

A ----|>---+

|

+---[NOT]--- OUT

B ----|>---+

**5. NOR Gate**

* **Function:** Outputs HIGH (1) only if all inputs are LOW (0). It’s the complement of the OR gate.
* **Truth Table:**

| **Input A** | **Input B** | **Output** |
| --- | --- | --- |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |

* **Circuit Diagram**: NOR gates combine an OR gate followed by a NOT gate.

A ----|>|---+

|

+---[NOT]--- OUT

B ----|>|---+

**6. XOR Gate**

* **Function:** Outputs HIGH (1) if inputs are different.
* **Truth Table:**

| **Input A** | **Input B** | **Output** |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

* **Circuit Diagram**: XOR gates use a combination of AND, OR, and NOT gates.

A ----+

+---[AND]---+

B ----| |

[NOT] OUT

**7. XNOR Gate**

* **Function:** Outputs HIGH (1) if inputs are the same. It’s the complement of XOR.
* **Truth Table:**

| **Input A** | **Input B** | **Output** |
| --- | --- | --- |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

* **Circuit Diagram**: XNOR gates combine an XOR gate with a NOT gate.

A ----+

+---[XOR]---[NOT]--- OUT

B ----+