**PRACTICAL 2**

**COMPUTER ORGANISATION AND ARCHITECTURE**

|  |  |
| --- | --- |
| **NAME: VARUN KHADAYATE** | **ROLL NO: A016** |
| **PROGRAM: BTECH SY** | **DIVISION: CSBS** |
| **BATCH: 1** | **DATE OF EXPERIMENT: 05/08/2020** |

**AIM**

**Verification and interpretation of truth table for AND, OR, NOT, NAND, NOR, Ex-OR, Ex-NOR gates.**

**THEORY**

1. **AND Gate**

The AND gate is a basic digital logic gate that implements logical conjunction - it behaves according to the truth table to the right. A HIGH output results only if all the inputs to the AND gate are HIGH. If none or not all inputs to the AND gate are HIGH, LOW output results.

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Truth Table | | |
| 2-input AND gate  2-input AND Gate | B | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| Boolean Expression **Q = A.B** | Read as A **AND** B gives Q | | |

1. **OR Gate**

The OR gate is a digital logic gate that implements logical disjunction – it behaves according to the adjacent truth table. A HIGH output results if one or both the inputs to the gate are HIGH. If neither input is high, a LOW output results.

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Truth Table | | |
| 2-input OR gate  2-input OR Gate | B | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |
| Boolean Expression **Q = A+B** | Read as A **OR** B gives Q | | |

1. **NOT Gate**

In digital logic, an inverter or NOT gate is a logic gate which implements logical negation. The truth table is shown on the right.

|  |  |  |
| --- | --- | --- |
| Symbol | Truth Table | |
| logic not gate  Inverter or NOT Gate | A | Q |
| 0 | 1 |
| 1 | 0 |
| Boolean Expression **Q = not A or A** | Read as inverse of **A** gives Q | |

1. NAND Gate

In digital electronics, a NAND gate is a logic gate which produces an output which is false only if all its inputs are true; thus, its output is complement to that of an AND gate. A LOW output results only if all the inputs to the gate are HIGH; if any input is LOW, a HIGH output results.

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Truth Table | | |
| 2-input logic NAND gate  2-input NAND Gate | B | A | Q |
| 0 | 0 | 1 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| Boolean Expression **Q = A.B** | Read as A **AND** B gives **NOT** Q | | |

1. NOR Gate

The NOR gate is a digital logic gate that implements logical NOR - it behaves according to the truth table to the right. A HIGH output results if both the inputs to the gate are LOW; if one or both input is HIGH, a LOW output results. NOR is the result of the negation of the OR operator.

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Truth Table | | |
| 2-input nor gate  2-input NOR Gate | B | A | Q |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 0 |
| Boolean Expression **Q = A+B** | Read as A **OR** B gives **NOT** Q | | |

1. XOR Gate

XOR gate is a digital logic gate that gives a true output when the number of true inputs is odd. An XOR gate implements an exclusive or; that is, a true output results if one, and only one, of the inputs to the gate is true. If both inputs are false or both are true, a false output results.

|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Truth Table | | |
| 2-input exclusive-or gate  2-input Ex-OR Gate | B | A | Q |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |
| Boolean Expression Q = A ⊕ B | A **OR** B but NOT **BOTH** gives Q | | |

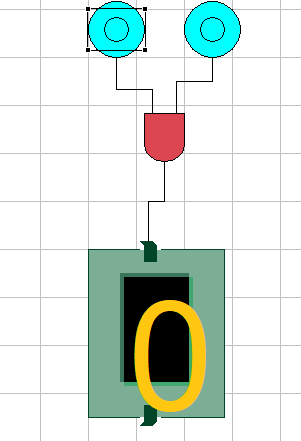
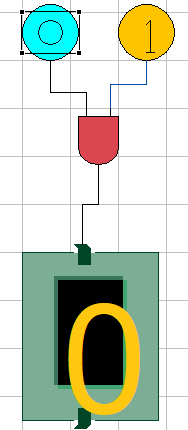
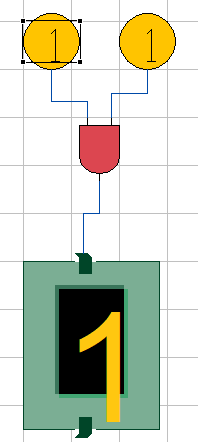
1. XNOR Gate

The XNOR gate is a digital logic gate whose function is the logical complement of the exclusive OR gate. The two-input version implements logical equality, behaving according to the truth table to the right, and hence the gate is sometimes called an "equivalence gate".

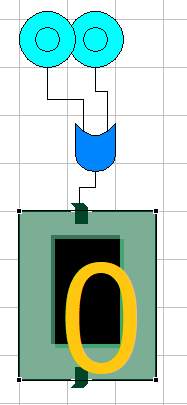
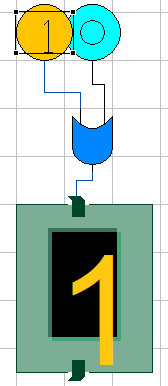
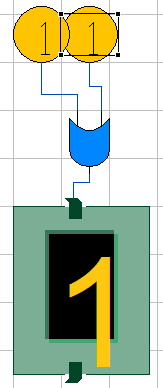
|  |  |  |  |
| --- | --- | --- | --- |
| Symbol | Truth Table | | |
| 2-input exclusive-nor gate  2-input Ex-NOR Gate | B | A | Q |
| 0 | 0 | 1 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| Boolean Expression Q = A ⊕ B | Read if A **AND** B the **SAME** gives Q | | |

**SIMULATION**

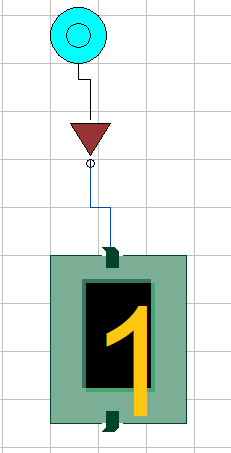
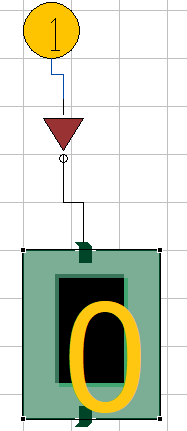
1. **AND GATE**

**  **

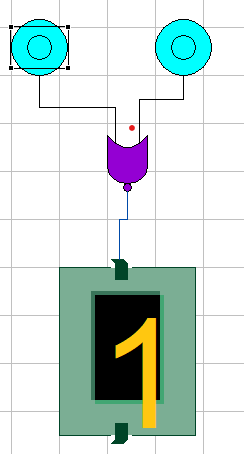
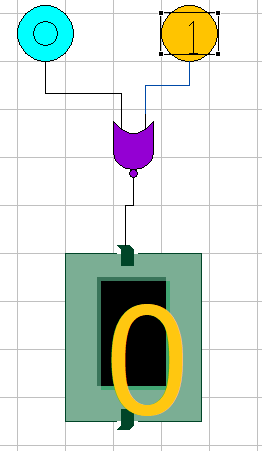
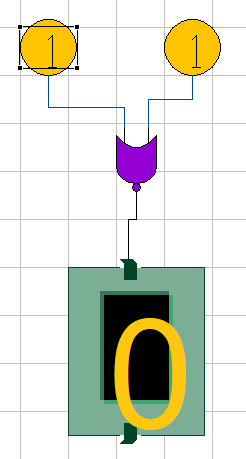
1. **OR GATE**

**  **

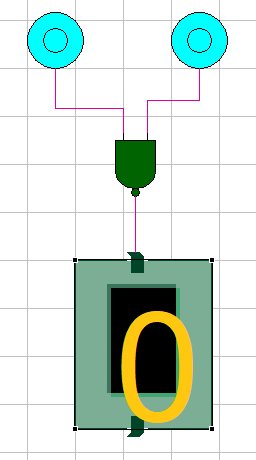
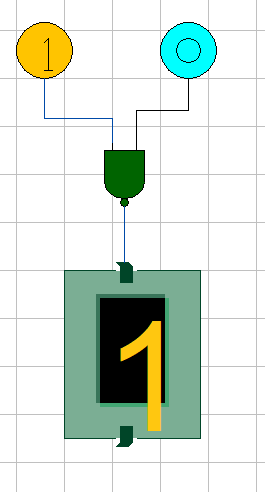
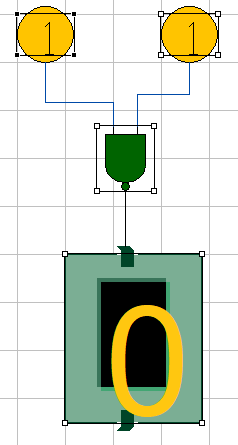
1. **NOT GATE**

****

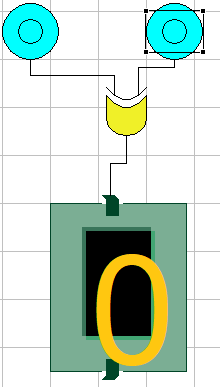
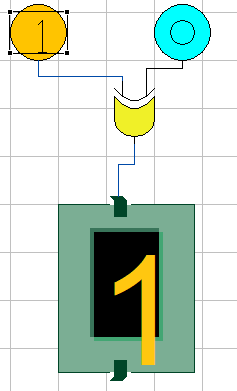
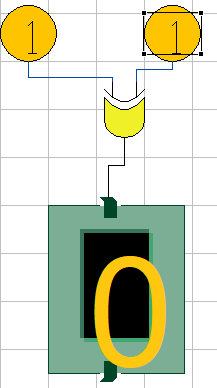
1. **NOR GATE**

**  **

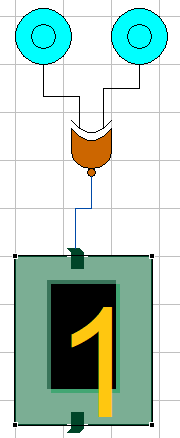
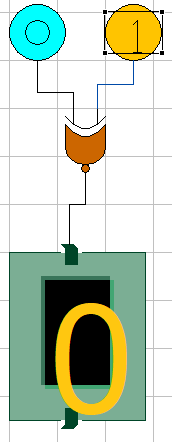
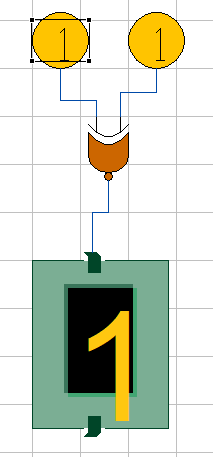
1. **NAND GATE**

**** **** 

1. **XOR GATE**

**  **

1. **XNOR GATE**

**  **

**CONCLUSION**

Hence, we are able to study and verify the study of all the logic gates