SRM Institute of Science and Technology

Delhi – Meerut Road, Sikri Kalan, Ghaziabad, Uttar Pradesh – 201204

Department of Computer Applications

Circular – 2024-25

BCA 1st Sem

Digital Logic Design (UCA24101J)

Lab Manual

Lab 1: Verification of Basic Gates and Derived Gates

Title: Verification of Basic Gates and Derived Gates

Aim: To verify the truth tables of basic logic gates (AND, OR, NOT) and derived logic gates (NAND, NOR, XOR, XNOR).

Procedure:

1. Implement the following logic gates using appropriate hardware/software (e.g., Breadboard and ICs, or a logic simulation software):

   AND gate

   OR gate

   NOT gate

   NAND gate

   NOR gate

   XOR gate

   XNOR gate

2. For each gate, apply all possible combinations of input values.
3. Record the output for each input combination.
4. Compare the recorded outputs with the standard truth tables of the respective logic gates.

Source Code:

5. Conceptual Description (Hardware):
   - Use appropriate ICs for each gate (e.g., 7408 for AND, 7432 for OR, 7404 for NOT, 7400 for NAND, 7402 for NOR, 7486 for XOR, 74266 for XNOR).
   - Connect the ICs on a breadboard.
   - Apply input signals using a signal generator or switches.
   - Observe the output using an LED or a logic analyzer.
6. Conceptual Description (Software Simulation):

Use a logic simulation software (e.g., Logisim, Multisim).

Place the required logic gate components on the workspace.

Connect the inputs and outputs.

Simulate the circuit and observe the output for all input combinations.

Input:

7. For each gate, the input will be all possible combinations of 0 and 1.

   For single-input gates (NOT): 0, 1

   For two-input gates (AND, OR, NAND, NOR, XOR, XNOR): 00, 01, 10, 11

Expected Output:

8. The output for each gate should match its standard truth table:

   NOT:

   0 -> 1

   1 -> 0

   AND:

   00 -> 0

   01 -> 0

   10 -> 0

   11 -> 1

   OR:

   00 -> 0

   01 -> 1

   10 -> 1

   11 -> 1

   NAND:

   00 -> 1

   01 -> 1

   10 -> 1

11 -> 0

NOR:

00 -> 1

01 -> 0

10 -> 0

11 -> 0

XOR:

00 -> 0

01 -> 1

10 -> 1

11 -> 0

XNOR:

00 -> 1

01 -> 0

10 -> 0

11 -> 1

General Structure for Remaining Labs

The following labs will follow the same structure as Lab 1:

Lab 2: NAND as Universal Gate NOR as Universal Gate

Title: NAND as Universal Gate, NOR as Universal Gate

Aim: To implement basic gates (AND, OR, NOT) using only NAND gates and only NOR gates.

Procedure: Design and implement circuits for AND, OR, and NOT gates using only NAND gates in one case, and only NOR gates in another case. Verify the truth tables.

Source Code: (Conceptual Description of circuit implementation using NANDs and then using NORs)

Input: 0, 1 and combinations

Expected Output: Truth tables of AND, OR, NOT

Lab 3: Laws of Boolean Expressions

Title: Laws of Boolean Expressions

Aim: To verify De Morgan's Laws and other Boolean algebra laws.

Procedure: Implement both sides of the Boolean equations and verify that their truth tables are identical.

Source Code: (Conceptual Description of implementing both sides of the equation)

Input: 0, 1 and combinations

Expected Output: Truth tables

Lab 4: Verifications of Distributive Law

Title: Verification of Distributive Law

Aim: To verify the distributive law of Boolean algebra: A(B+C) = AB + AC.

Procedure: Implement both sides of the equation and verify their truth tables.

Source Code: (Conceptual Description)

Input: 0, 1 combinations

Expected Output: Identical truth tables for both sides.

Lab 5: Simplifying Boolean Expressions using theorems

Title: Simplifying Boolean Expressions using theorems

Aim: To simplify given Boolean expressions using Boolean algebra theorems and verify the simplified expression.

Procedure: Simplify a given expression. Implement both the original and simplified expressions. Verify that their truth tables are identical.

Source Code: (Conceptual Description)

Input: 0, 1 and combinations

Expected Output: Truth tables

Lab 6: Implementation of Binary Addition and Subtraction

Title: Implementation of Binary Addition and Subtraction

Aim: To implement binary addition and subtraction circuits.

Procedure: Design and implement a circuit that can perform both addition and subtraction. Use adders/subtractors and control logic.

Source Code: (Conceptual Description)

Input: Two binary numbers and a control signal (for addition or subtraction).

Expected Output: The sum or difference of the two binary numbers.

Lab 7: Half Adder and Full Adder

Title: Half Adder and Full Adder

Aim: To implement and verify the functionality of half adder and full adder circuits.

Procedure: Implement half adder and full adder circuits. Verify their truth tables.

Source Code: (Conceptual Description)

Input:

Half Adder: Two single-bit inputs.

Full Adder: Two single-bit inputs and a carry-in.

Expected Output:

Half Adder: Sum and Carry.

Full Adder: Sum and Carry-out.

Lab 8: Half Subtractor and Full Subtractor

Title: Half Subtractor and Full Subtractor

Aim: To implement and verify the functionality of half subtractor and full subtractor circuits.

Procedure: Implement half subtractor and full subtractor circuits. Verify their truth tables.

Source Code: (Conceptual Description)

Input:

Half Subtractor: Two single-bit inputs.

Full Subtractor: Two single-bit inputs and a borrow-in.

Expected Output:

Half Subtractor: Difference and Borrow.

Full Subtractor: Difference and Borrow-out.

Lab 9: Implementation of Multiplexer

Title: Implementation of Multiplexer

Aim: To implement a multiplexer (MUX) circuit.

Procedure: Implement a 4-to-1 multiplexer (or another size as appropriate). Verify its truth table.

Source Code: (Conceptual Description)

Input: Data inputs and select inputs.

Expected Output: The selected data input at the output.

Lab 10: Implementation of DeMultiplexer

Title: Implementation of DeMultiplexer

Aim: To implement a demultiplexer (DEMUX) circuit.

Procedure: Implement a 1-to-4 demultiplexer (or another size as appropriate). Verify its truth table.

Source Code: (Conceptual Description)

Input: A data input and select inputs.

Expected Output: The data input routed to the selected output.

Lab 11: Implementation of Shift Registers and Serial Transfer

Title: Implementation of Shift Registers and Serial Transfer

Aim: To implement a shift register and demonstrate serial data transfer.

Procedure: Implement a 4-bit shift register (e.g., using D flip-flops). Demonstrate serial loading and serial reading of data.

Source Code: (Conceptual Description)

Input: Serial data input, clock signal.

Expected Output: Serial data output.

Lab 12: Four Bit Binary Shift Counters

Title: Four Bit Binary Shift Counters

Aim: To implement a 4-bit binary shift counter.

Procedure: Implement a 4-bit shift counter. Verify its counting sequence.

Source Code: (Conceptual)

Input: Clock signal

Output: 4-bit binary count

Lab 13: Ring Counters

Title: Ring Counters

Aim: To implement a ring counter.

Procedure: Implement a ring counter (e.g., using D flip-flops). Observe the sequence of states.

Source Code: (Conceptual Description)

Input: Clock signal, Initial preset

Expected Output: The circulating pattern of the active bit.

Lab 14: Implementation of DOWN Counter

Title: Implementation of DOWN Counter

Aim: To implement a synchronous or asynchronous DOWN counter.

Procedure: Design and implement a DOWN counter. Verify its counting sequence.

Source Code: (Conceptual Description)

Input: Clock signal

Expected Output: The counter counts down with each clock pulse.

Lab 15: Implementation of DOWN Counter

Title: Implementation of DOWN Counter

Aim: To implement a synchronous or asynchronous DOWN counter.

Procedure: Design and implement a DOWN counter. Verify its counting sequence.

Source Code: (Conceptual Description)

Input: Clock signal

Expected Output: The counter counts down with each clock pulse.