National University of Computer and Emerging Sciences



**Laboratory Manuals**

*for*

**Digital Logic Design**

|  |  |
| --- | --- |
| Course Instructor | Ms. Mamoona Awan |
| Lab Instructor(s) | Zara Asif |
| Section | A2 |

Department of Computer Science

FAST-NU, Lahore, Pakistan

**computer Logic Design Lab Manual-2**

**Arithmetic Operations On Number Systems**

**Introduction To Logic Gates**

**Spring-2017 Lab Instructor: Ms. Zara Asif**

**Objective:**

* Students should be able to perform addition, subtraction and multiplication on decimal, binary, octal and hexadecimal number systems.
* Students should be able to understand how IC’s work and to implement basic gates of AND, OR and NOT on breadboard. Also, to verify their implementation results with Truth Table. Introduction to LogicWork software.

# Statement 1:

1. Convert following decimal numbers to binary, octal and hexadecimal: **23**, **16**, **3.4**, **7.81**, **11.3**
2. Add the following binary numbers: **11+11, 110+100**
3. Using 8-bit two’s complement numbers Add **−8 to +3**, Add **−5 to −2**
4. Using 8-bit One’s and then Two's Complement Numbers Subtract **+5 from +8**, Subtract **−6 from +7**
5. Multiply: **11001012 x 1111012** , **1001012 x 1012**
6. Subtract the following hexadecimal numbers: **(84)16-(2A)16** **,** **(C3)16-(B)16**
7. Add the following hexadecimal numbers: **(4A)16+(3F)16****, (BF)16+(AC)16**
8. Multiply the following hexadecimal numbers: **(1F)16 \* (C)16** **, (2B)16 \* (5A)16**
9. Subtract the following octal numbers: **(537)8 - (162)8**
10. Add the following octal numbers: **(162)8 + (537)8 , (136)8 + (636)8**
11. Multiply the following octal numbers: **(6)8\* (23)8 , (15)8\* (44)8**

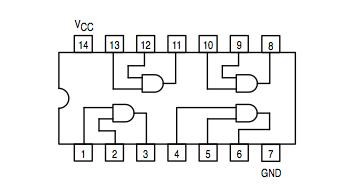
# Logic Gates

**Apparatus:** Logic trainer, Logic probe, 74LS08, 74LS32 and 74LS04 ICs, Jumper wires

**Software:** LogicWork 4

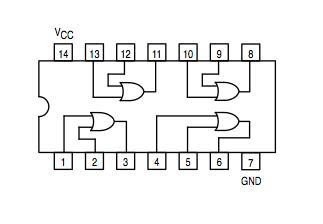
**Theory:** Digital circuits are the electronic circuits that manipulate binary information. Logic gates are the basic building blocks in constructing digital circuits. Logic gate has one output and one or more inputs. Each logic gate performs a specific logical operation. The interconnection between inputs and outputs of gates form a digital circuit. Any digital circuit can implemented using three basic logical operations called AND, OR, and NOT. That is why AND, OR, and NOT gates are referred as basic logic gates. AND and OR logic functions exhibit the phenomenon of dominance. In both cases, there is an input value that will force the output of the gate to a known value regardless of the state of other inputs. This value is known as the dominant value of the gate. The dominant value of an AND gate is zero, while the dominant value of an OR gate is one. In this experiment, we will use **74LS08, 74LS32** and **74LS04** ICs for the implementation of **AND, OR** and **NOT** logical operations.

**AND GATE:** 74LS08 IC contains four 2-input AND gates.

**Function Table/ Truth Table and Connection Diagram:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Inputs** | | **Output** |
| **A** |  | **B** | **Y** |
| L |  | L | L |
| L |  | H | L |
| H |  | L | L |
| H |  | H | H |

H= Logic High, L= Logic Low

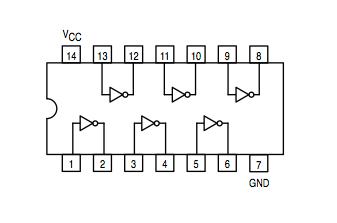
**OR GATE:** 74LS32 IC contains four 2-input OR gates.

**Function Table and Connection Diagram:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Inputs** | | **Output** |
| **A** |  | **B** | **Y** |
| L |  | L | L |
| L |  | H | H |
| H |  | L | H |
| H |  | H | H |

H= Logic High, L= Logic Low

**NOT GATE:** 74LS04 IC contains six NOT gates.

**Function Table and Connection Diagram:**

|  |  |
| --- | --- |
| **Input** | **Output** |
| **A** | **Y** |
| L | H |
| H | L |

H= Logic High, L= Logic Low

**Testing of ICs:**

Before starting implementation of a specific logic circuit, all basic gate ICs should be tested in order to make sure that the ICs are working properly. Using the function table (truth table) for each gate, in a particular IC, apply all input combinations one by one and check its output logic level on LED.

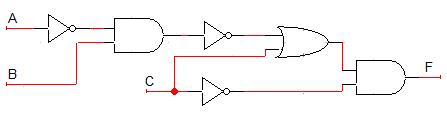
**Troubleshooting:**

After testing all required number and type of ICs we need to implement a digital circuit, we start implementing the circuit on logic trainer. Once we complete the implementation, we need to test the output of the circuit to make sure that whether the circuit is working accurately or not. Using the truth table that represents the functionality of the logic circuit, we apply all input combinations one by one and check its output logic level on LED.

If the output of circuit is incorrect for some input combinations then there must be some fault in hardware implementation of the circuit. The fault can be an open circuit, short circuit, incorrect interconnection between the gates, incorrect connection with switches etc. Therefore, we need to troubleshoot the circuit. To troubleshoot the logic circuit apply the input combination at which the output of the circuit is incorrect, and then trace back the fault by checking each intermediate connection of the circuit using logic probe. The most efficient way to quickly reach the fault location is to exploit the low logic level dominance in AND gate and high logic level dominance in OR gate.

# Statement 2:

Implement the following logic circuit on logic trainer & make a truth table



|  |  |  |  |
| --- | --- | --- | --- |
| A | B | C | F |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Statement 3:

Implement the following function on logic trainer. First make its circuit diagram and truth table. Also implement it in logic works.

**F= A’BC + BC + AB’C’**

**POST LAB:**

* For the Boolean functions implement the circuit on Logic Work 4 and make a truth table.

1. **F1 = XYZ + X’Y + XYZ’**
2. **F2 = [(A+BC’)(A’C’)]’+ C**

* Draw the timing diagram of the following circuit including input **A, B, C** and at points **D, E, F.** Also write its Boolean function.

