

Ahsanullah University of Science and Technology (AUST)
Department of Computer Science and Engineering

LABORATORY MANUAL

Course No: CSE2106
Course Title: Digital Logic Design Lab

For the students of 2nd Year, 1st semester of
B.Sc. in Computer Science and Engineering program

TABLE OF CONTENTS

COURSE INFORMATION

| | |
|--------------------------|---|
| COURSE OBJECTIVES | 1 |
| PREFERRED TOOLS | 1 |
| TEXT/REFERENCE BOOK..... | 1 |

| | |
|---|---|
| ADMINISTRATIVE POLICY OF THE LABORATORY | 2 |
|---|---|

LIST OF SESSIONS

SESSION 1:

| | |
|--|---|
| <i>Introduction to logic gates and different digital ICs</i> | 3 |
|--|---|

SESSION 2:

| | |
|--|----|
| <i>Truth tables and simplification using Boolean algebra</i> | 14 |
|--|----|

SESSION 3:

| | |
|--------------------------------------|----|
| <i>Truth tables and K-maps</i> | 16 |
|--------------------------------------|----|

SESSION 4:

| | |
|---|----|
| <i>Combinational Logic Circuits</i> | 17 |
|---|----|

SESSION 5:

| | |
|---|----|
| <i>Construction of adders, subtractors, magnitude comparators using basic gates</i> | 18 |
|---|----|

SESSION 6:

| | |
|---|----|
| <i>Construction of BCD adder and Adder/Subtractor circuit</i> | 19 |
|---|----|

SESSION 7:

| | |
|---|----|
| <i>Design using Multiplexers and Decoders</i> | 20 |
|---|----|

SESSION 8:

| | |
|--|----|
| <i>Design of Sequential circuits</i> | 21 |
|--|----|

SESSION 9:

| | |
|--|----|
| <i>Design of Synchronous and Asynchronous Counters ICs</i> | 22 |
|--|----|

SESSION 10:

| | |
|--|----|
| <i>Design of Shift Registers</i> | 23 |
|--|----|

SESSION 11:

| | |
|--------------------------|----|
| <i>Make-up Lab</i> | 25 |
|--------------------------|----|

| | |
|----------------------------|----|
| MID TERM EXAMINATION | 19 |
|----------------------------|----|

| | |
|------------------------------|----|
| FINAL TERM EXAMINATION | 24 |
|------------------------------|----|

Course Information

Course objectives:

- Understand theory of operation for most of digital electronic devices.
- Analyzing how can a digital computer perform the complex operations based on simply manipulating bits (zeros and ones).
- Designing digital logic systems.

Preferred tools:

- ✓ DC Power Supply
- ✓ Breadboard
- ✓ Pulse Generator
- ✓ Digital Probe
- ✓ Different Digital Integrated ICs

Text/Reference Book:

- i. *"Digital Design With An Introduction to the Verilog HDL"* By M. Morris Mano, Michael D. Ciletti (5th edition).
- ii. *"Microprocessor Data Handbook"*–BPB Publications

Administrative Policy of the Laboratory

- 1) You are not allowed to eat or drink in the Laboratory. You are expected to conduct yourself professionally, and to keep your working area neat and clean. You are required to return all equipment and parts used in the experiment to their proper places before you leave the lab.
- 2) You are expected to work in a group, which is defined to contain at most three people.
- 3) CSE 2106 is a closed lab, therefore, you are expected to build and test your circuit in the lab within the allotted time. You cannot build the circuit ahead of time, but you are required to complete the design before coming to class.
- 4) Lab reports in the prescribed format are due before the experiment is performed. You need to submit report individually.
- 5) If in a particular lab session you finish your experiment ahead of time you may choose to work on any of the previous experiments you have attempted to build and were unable to get it to work. You will not be allowed to work on an experiment you have not worked on previously.
- 6) There will be a make-up lab session at the end of the semester. During that session you will be allowed to make up at most two experiments you have previously worked on. Note that this is an opportunity being afforded to you so that you may work again on an experiment you previously attempted but were unsuccessful. If you did not work on an experiment during a previous lab session you will not be allowed to make it up during the make-up session.

Session1: Introduction to logic gates and different digital ICs

Objectives

- To get acquainted with the Analog/Digital Training System.
- To get acquainted with different standard integrated circuits (ICs).
- To study the basic logic gates: AND, OR, INVERT, NAND, NOR, and XOR.
- To understand formulation of Boolean function and truth table for logic circuits.

Apparatus

- Analog/Digital Training System
- IC Type 7400 Quadruple 2-input NAND gates
- IC Type 7402 Quadruple 2-input NOR gates
- IC Type 7404 Hex Inverters
- IC Type 7408 Quadruple 2-input AND gates
- IC Type 7432 Quadruple 2-input OR gates
- IC Type 7486 Quadruple 2-input XOR gate

Theory

Chapters 1 & 2 of the textbook (Mano, Ciletti).

Analog/Digital Training System:

The Analog/Digital Training System consists of DC power supply, breadboard, pulse generator and adigital probe.

Useful features include:

1. DC Power Supply:

- Fixed DC Outputs: +5V & -5V
- Variable DC Outputs: +3V to +15V, -3V to -15V

2. Breadboard:

- Terminal strips arranged for easy connection of standard ICs

3. Pulse Generator:

4. Digital Probe

The Breadboard

The breadboard consists of two terminal strips and two bus strips (often broken in the center). Each bus strip has two rows of contacts. Each of the two rows of contacts is node. That is, each contact along a row on a bus strip is connected together (inside the breadboard). Bus strips are used primarily for power supply connections, but are also used for any node requiring a large number of connections. In the example of breadboard shown in figure 1, each terminal strip has 60 rows and 5 columns of contacts on each side of the center gap. Each row of 5 contacts is a node.

You will build your circuits on the terminal strips by inserting the leads of circuit components into the contact receptacles and making connections with 22-26 gauge wire. There are wire cutter/strippers and a spool of wire in the lab. It is a good practice to wire +5V and 0V power supply connections to separate bus strips.

The 5V supply **MUST NOT BE EXCEEDED** since this will damage the ICs (Integrated circuits) used during the experiments. Incorrect connection of power to the ICs could result in them exploding or becoming ery hot.

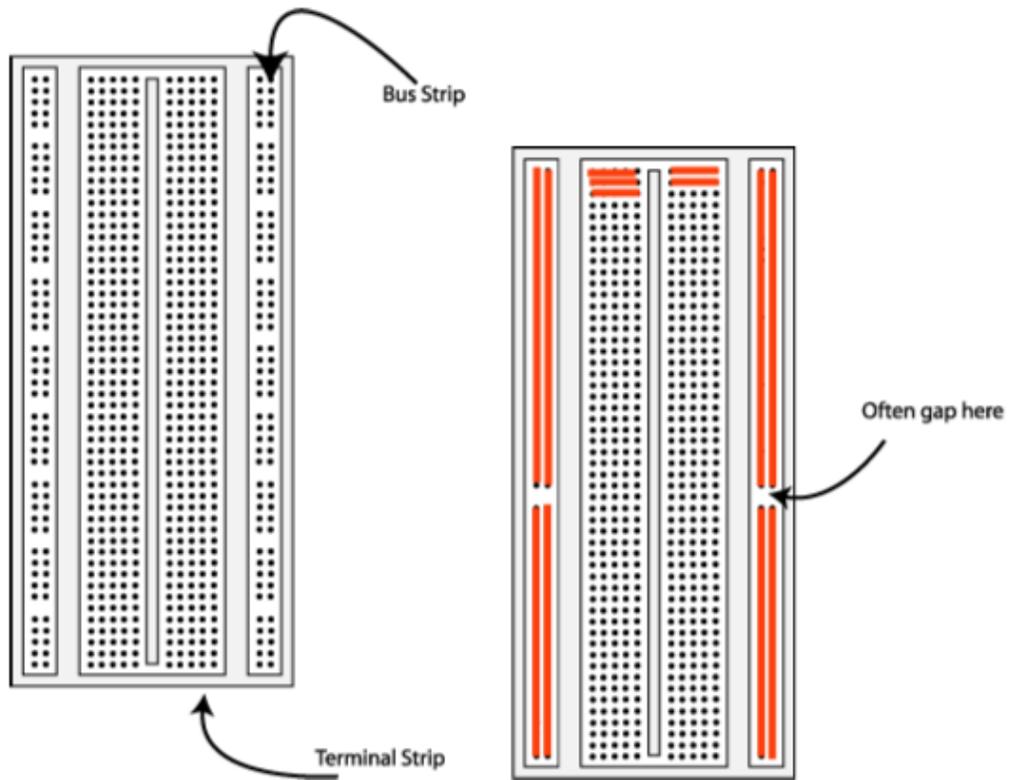


Fig. 1: The breadboard. The shaded lines indicate connected holes

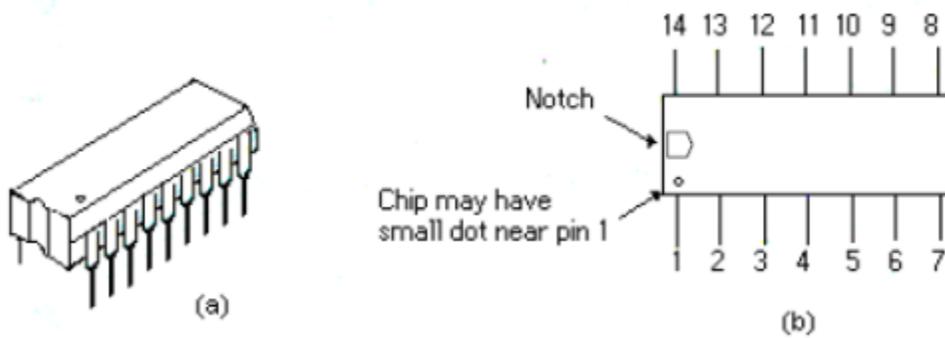


Fig. 2: (a) Dual-In-Line Package (b) Top view showing Pin numbers

Digital Integrated IC's

Digital ICs are a collection of resistors, diodes and transistors fabricated on a single piece of semiconductor material usually silicon and referred to as “chip”. The chip is enclosed in a protective plastic or ceramic package with pins extended out for connecting the IC to other devices. The most common type of package is a dual-in-line package (DIP) as shown in figure 2. The pins are numbered counterclockwise when viewed from the top of the package with respect to an identifying notch or dot at one end of the chip. The DIP below is a 14-pin package. 16, 20, 24, 28, 40 and 64 pin packages are also available. The fabricated resistors, diodes and transistors reside in the chip are called logic gates. Different chip may contain different amount of these logic gates. Digital ICs are often categorized according to their circuit complexity as measured by the number of equivalent logic gates in an IC. There are currently five standard levels of complexity as in Table 1.

Table 1. Standard levels of complexity

| Complexity | Approx. gates per chip | Typical products |
|--------------------------------------|------------------------|--|
| Small scale integration (SSI) | Less than 12 | Logic gates, flip flops |
| Medium scale integration (MSI) | 12 to 99 | Adders, Counters, Multiplexers |
| Large scale integration (LSI) | 100 to 9999 | ROM, RAM, 8 bit Microprocessors |
| Very large scale integration (VLSI) | 10, 000 to 99, 999 | 16 and 32 bit Microprocessors |
| Ultra large scale integration (ULSI) | 100, 000 to more | 64 bit microprocessors, special processors |

Building the Circuit on Breadboard

Throughout these experiments, we will use TTL chips to build circuits. The steps for wiring a circuit should be completed in the order described below:

1. Make sure the power is off before you build anything!
2. Connect the +5V and ground (GND) leads of the power supply to the power and ground bus strips on your breadboard.

3. Plug the chips you will be using into the breadboard. Point all the chips in the same direction with pin 1 at the upper-left corner. (Pin 1 is often identified by a dot or a notch next to it on the chip package).
4. Connect +5V and GND pins of each chip to the power and ground bus strips on the breadboard.
5. Select a connection on your schematic and place a piece of hook-up wire between corresponding pins of the chips on your breadboard. It is better to make the short connections before the longer ones. Mark each connection on your schematic as you go, so as not to try to make the same connection again at a later stage.

If an error is made and is not spotted before you turn the power on. Turn the power off immediately before you begin to rewire the circuit.

At the end of the laboratory session, collect your hook-up wires, chips and all equipment and return them to the lab attendant. Tidy the area that you were working in and leave it in the same condition as it was before you started.

Common Causes of Problems

1. Not connecting the ground and/or power pins for all chips.
2. Not turning on the power supply before checking the operation of the circuit.
3. Leaving out wires.
4. Plugging wires into the wrong holes.
5. Driving a single gate input with the outputs of two or more gates.
6. Modifying the circuit with the power on.

Logic Gates

AND: A multi-input circuit in which the output is 1 only if all inputs are 1. The symbolic representation of the AND gate is shown in Table 2.

OR: A multi-input circuit in which the output is 1 when any input is 1. The symbolic representation of the OR gate is shown in Table 2.

NOT: The output is 0 when the input is 1, and the output is 1 when the input is 0. The symbolic representation of an inverter is shown in Table 2.

NAND: AND followed by NOT. The symbolic representation of the NAND gate is shown in Table 2.

NOR: OR followed by NOT as shown in Table 2.

EX-OR: The output of the Exclusive –OR gate, is 0 when its two inputs are the same and its output is 1 when its two inputs are different.

Truth Table: Representation of the output logic levels of a logic circuit for every possible combination of levels of the inputs. This is best done by means of a systematic tabulation.

Table 2: Digital Logic gates

| GATE | SYMBOL | NOTATION | TRUTH TABLE | | | | | | | | | | | | | | | | | | |
|-------------|---|------------------------|--|-------|--------|--------|-------|---|----------|---|---|---|---|---|---|---|---|---|---|---|---|
| <u>AND</u> |  | $A \cdot B$ | <table border="1"> <thead> <tr> <th colspan="2">INPUT</th> <th>OUTPUT</th> </tr> <tr> <th>A</th> <th>B</th> <th>A AND B</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table> | INPUT | | OUTPUT | A | B | A AND B | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 1 |
| INPUT | | OUTPUT | | | | | | | | | | | | | | | | | | | |
| A | B | A AND B | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 0 | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 1 | | | | | | | | | | | | | | | | | | | |
| <u>OR</u> |  | $A + B$ | <table border="1"> <thead> <tr> <th colspan="2">INPUT</th> <th>OUTPUT</th> </tr> <tr> <th>A</th> <th>B</th> <th>A OR B</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> </tr> </tbody> </table> | INPUT | | OUTPUT | A | B | A OR B | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| INPUT | | OUTPUT | | | | | | | | | | | | | | | | | | | |
| A | B | A OR B | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 1 | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 1 | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 1 | | | | | | | | | | | | | | | | | | | |
| <u>NOT</u> |  | \bar{A} | <table border="1"> <thead> <tr> <th>INPUT</th> <th>OUTPUT</th> </tr> <tr> <th>A</th> <th>NOT A</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> </tr> </tbody> </table> | INPUT | OUTPUT | A | NOT A | 0 | 1 | 1 | 0 | | | | | | | | | | |
| INPUT | OUTPUT | | | | | | | | | | | | | | | | | | | | |
| A | NOT A | | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | | | | | | | | | | | | | | | | | | | | |
| <u>NAND</u> |  | $\overline{A \cdot B}$ | <table border="1"> <thead> <tr> <th colspan="2">INPUT</th> <th>OUTPUT</th> </tr> <tr> <th>A</th> <th>B</th> <th>A NAND B</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table> | INPUT | | OUTPUT | A | B | A NAND B | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| INPUT | | OUTPUT | | | | | | | | | | | | | | | | | | | |
| A | B | A NAND B | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 1 | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 1 | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 0 | | | | | | | | | | | | | | | | | | | |
| <u>NOR</u> |  | $\overline{A + B}$ | <table border="1"> <thead> <tr> <th colspan="2">INPUT</th> <th>OUTPUT</th> </tr> <tr> <th>A</th> <th>B</th> <th>A NOR B</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table> | INPUT | | OUTPUT | A | B | A NOR B | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 |
| INPUT | | OUTPUT | | | | | | | | | | | | | | | | | | | |
| A | B | A NOR B | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 1 | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 0 | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 0 | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 0 | | | | | | | | | | | | | | | | | | | |
| <u>XOR</u> |  | $A \oplus B$ | <table border="1"> <thead> <tr> <th colspan="2">INPUT</th> <th>OUTPUT</th> </tr> <tr> <th>A</th> <th>B</th> <th>A XOR B</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> </tr> </tbody> </table> | INPUT | | OUTPUT | A | B | A XOR B | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |
| INPUT | | OUTPUT | | | | | | | | | | | | | | | | | | | |
| A | B | A XOR B | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | 0 | | | | | | | | | | | | | | | | | | | |
| 0 | 1 | 1 | | | | | | | | | | | | | | | | | | | |
| 1 | 0 | 1 | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | 0 | | | | | | | | | | | | | | | | | | | |

Diagrams and Labeling:

Using the pin distribution for the TTL packages given by the manufacturer, and once that you design the circuit that performs the desired logic function, the next step is to wire up the circuit that implements this function. Because every chip has a different number of gates, a good implementation step is to make a diagram for the circuit and label all inputs, outputs and gates in the way shown in Figure 12. By doing this the wiring and testing process will be done very easily in the lab.

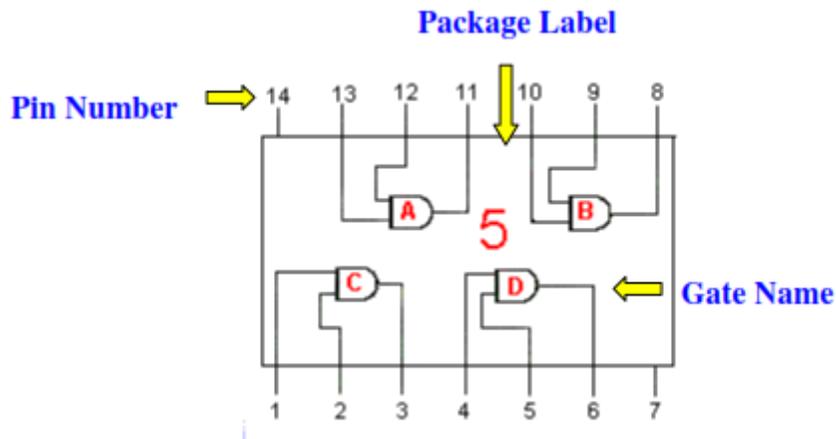


Fig 3: Package, gate and pin labeling

There are many methods that can be used to label a circuit. In this manual we show only one that is easy to understand and implement. In this method, every input and output pin shown in the diagram shows the respective pin number that corresponds to the gate in its package. The gates are labeled using a letter and a number. The letter labels a specific gate inside the package, and the number labels the package corresponding to its order. This is shown in Figure 13.

For Figure 13, the gate labeled *A1* means gate A in chip 1, *B1* means gate B in chip one, and *A2* means gate A in chip 2 and so on and so forth. Using this information it is very easy to wire the circuit on the breadboard since you only need to place the chips on the breadboard following the order that the chip was given on the diagram. Next, you only need to connect a wire between the pins that are given for every chip.

For example, a wire has to be connected between pin 6 of the third chip (*B3*) and pin 9 of the fourth chip (*C4*). Another wire has to be connected between pin 8 of the first chip (*C1*) and pin 5 of chip number 5 (*C5*), and so on and so forth.

Labeling the circuit in this way also makes it easier to find errors on the wiring during the testing process.

Another important characteristic that can be noticed from this diagram is that the flow of interconnections and signals follows a left to right direction. This means that typically in an electric connection diagram, the inputs will be shown on the left and outputs will be shown on the right.

The reason for this is that in this way it is easy to follow the flow of signals in the circuit, and the function implementation at every step that each gate performs. On other hand, it makes the circuit interconnections to appear clearer.

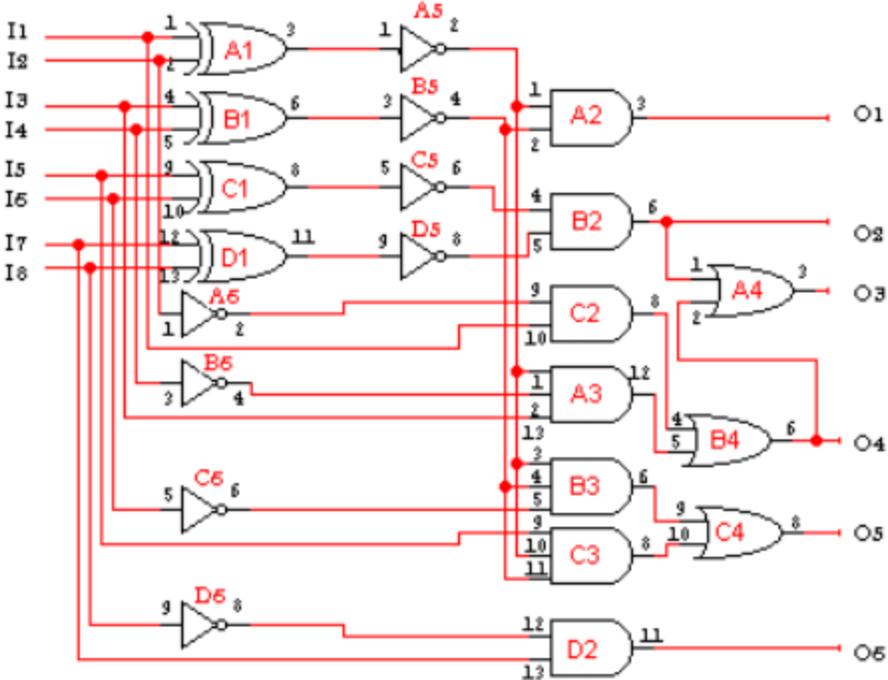
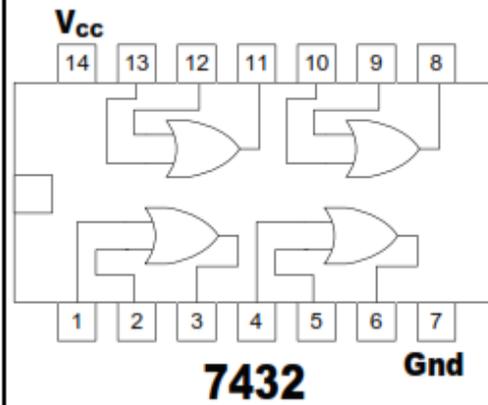


Fig 4: Circuit labels and connections

Task 1. Connect circuits for each of the logic gate as explained in experiment 1 and note your observations in truth table.

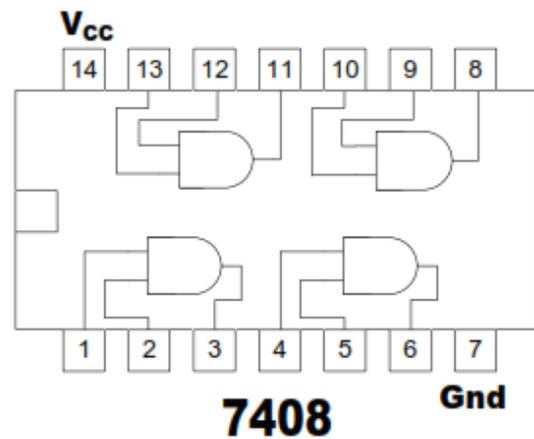
| Inputs | | Outputs |
|--------|-----|---------|
| A | B | Z |
| OFF | OFF | |
| OFF | ON | |
| ON | OFF | |
| ON | ON | |

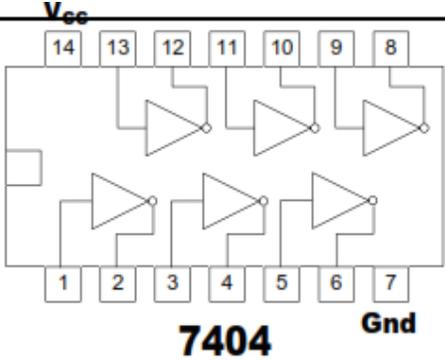
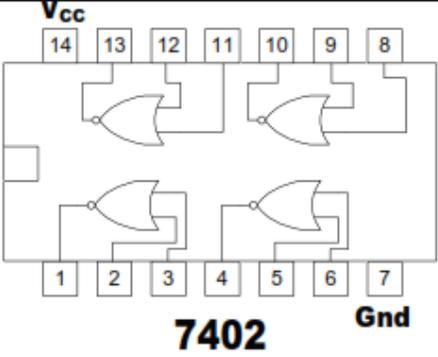
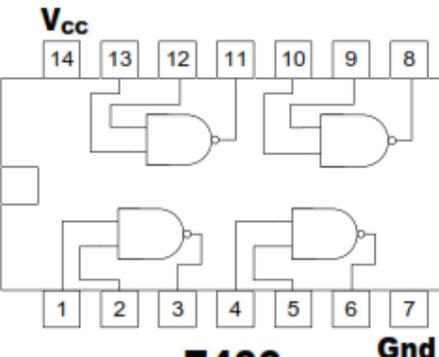
(a) **OR GATE 7432** (Quad 2 input)

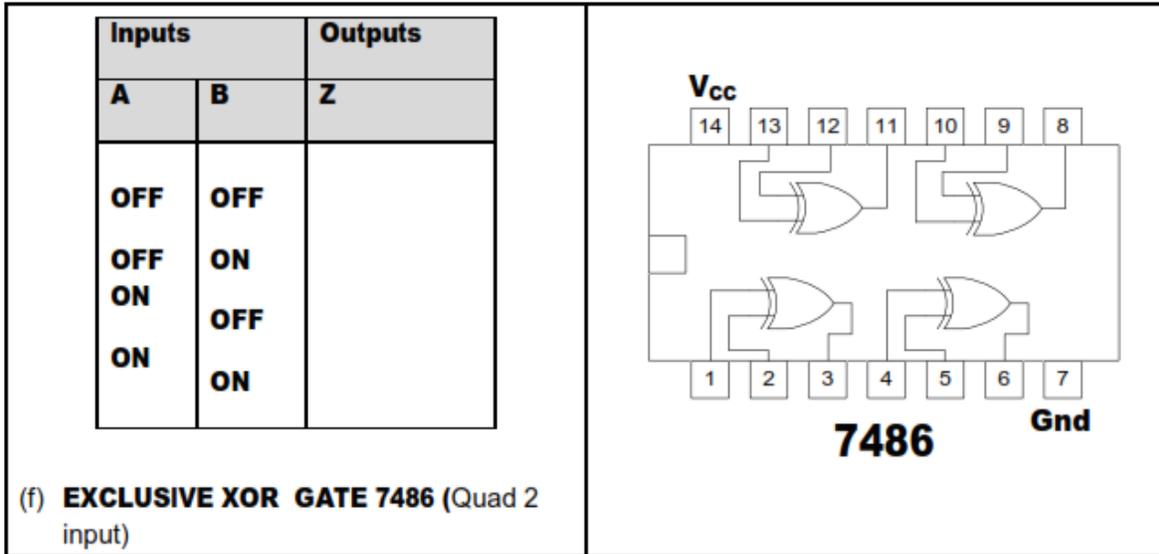


| Inputs | | Outputs |
|--------|-----|---------|
| A | B | Z |
| OFF | OFF | |
| OFF | ON | |
| ON | OFF | |
| ON | ON | |

(b) **AND GATE 7408** (Quad 2 input)

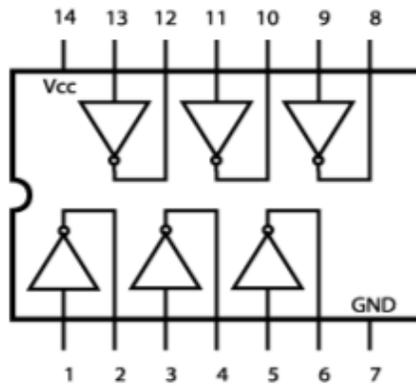
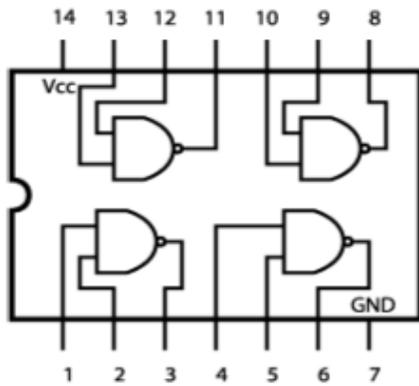


| <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th style="width: 50%;">Inputs</th> <th style="width: 50%;">Output</th> </tr> </thead> <tbody> <tr> <td style="text-align: left;">A</td> <td style="text-align: left;">Z</td> </tr> <tr> <td>OFF</td> <td></td> </tr> <tr> <td>ON</td> <td></td> </tr> </tbody> </table> <p>(c) NOT GATE 7404</p> | Inputs | Output | A | Z | OFF | | ON | |  <p style="text-align: center;">7404</p> | | | | | | | | | | |
|---|------------|---------|----------|----------|------------|---|------------|------------|---|------------|-----------|--|-----------|------------|--|-----------|-----------|--|---|
| Inputs | Output | | | | | | | | | | | | | | | | | | |
| A | Z | | | | | | | | | | | | | | | | | | |
| OFF | | | | | | | | | | | | | | | | | | | |
| ON | | | | | | | | | | | | | | | | | | | |
| <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th colspan="2">Inputs</th> <th>Outputs</th> </tr> <tr> <th style="width: 25%;">A</th> <th style="width: 25%;">B</th> <th style="width: 50%;">Z</th> </tr> </thead> <tbody> <tr> <td>OFF</td> <td>OFF</td> <td></td> </tr> <tr> <td>OFF</td> <td>ON</td> <td></td> </tr> <tr> <td>ON</td> <td>OFF</td> <td></td> </tr> <tr> <td>ON</td> <td>ON</td> <td></td> </tr> </tbody> </table> <p>(d) NOR GATE 7402 (Quad 2 input)</p> | Inputs | | Outputs | A | B | Z | OFF | OFF | | OFF | ON | | ON | OFF | | ON | ON | |  <p style="text-align: center;">7402</p> |
| Inputs | | Outputs | | | | | | | | | | | | | | | | | |
| A | B | Z | | | | | | | | | | | | | | | | | |
| OFF | OFF | | | | | | | | | | | | | | | | | | |
| OFF | ON | | | | | | | | | | | | | | | | | | |
| ON | OFF | | | | | | | | | | | | | | | | | | |
| ON | ON | | | | | | | | | | | | | | | | | | |
| <table border="1" style="width: 100%; border-collapse: collapse; text-align: center;"> <thead> <tr> <th colspan="2">Inputs</th> <th>Outputs</th> </tr> <tr> <th style="width: 25%;">A</th> <th style="width: 25%;">B</th> <th style="width: 50%;">Z</th> </tr> </thead> <tbody> <tr> <td>OFF</td> <td>OFF</td> <td></td> </tr> <tr> <td>OFF</td> <td>ON</td> <td></td> </tr> <tr> <td>ON</td> <td>OFF</td> <td></td> </tr> <tr> <td>ON</td> <td>ON</td> <td></td> </tr> </tbody> </table> <p>(e) NAND GATE 7400 (Quad 2 input)</p> | Inputs | | Outputs | A | B | Z | OFF | OFF | | OFF | ON | | ON | OFF | | ON | ON | |  <p style="text-align: center;">7400</p> |
| Inputs | | Outputs | | | | | | | | | | | | | | | | | |
| A | B | Z | | | | | | | | | | | | | | | | | |
| OFF | OFF | | | | | | | | | | | | | | | | | | |
| OFF | ON | | | | | | | | | | | | | | | | | | |
| ON | OFF | | | | | | | | | | | | | | | | | | |
| ON | ON | | | | | | | | | | | | | | | | | | |



Example Implementation of a Logic Circuit

Task 2. Build a circuit to implement the Boolean function $F = A' \cdot B'$ using TTL IC 74LS00 (AND gate) and TTL IC 7404 (INVERTER).



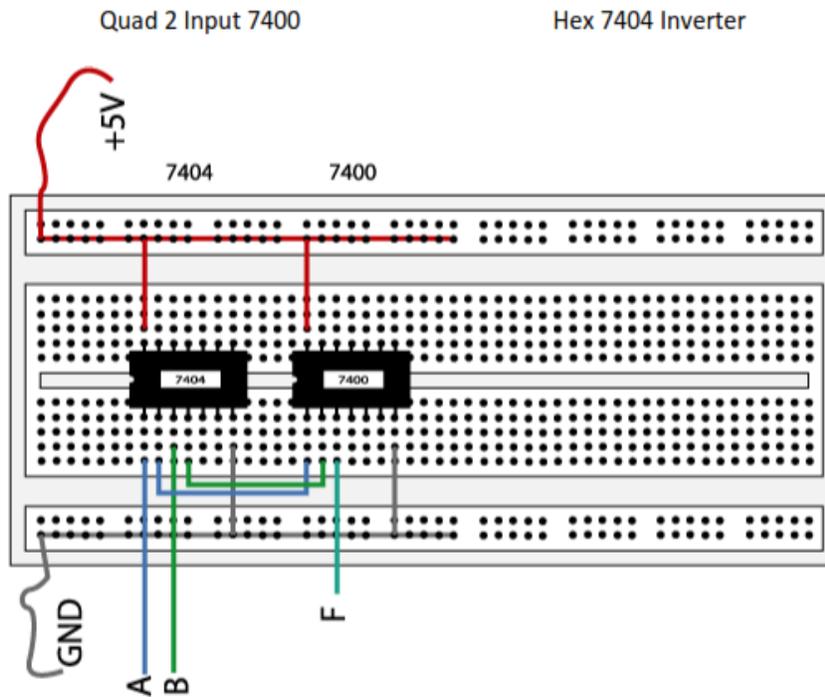


Fig 5: The complete designed and connected circuit

Sometimes the chip manufacturer may denote the first pin by a small indented circle above the first pin of the chip. Place your chips in the same direction, to save confusion at a later stage. Remember that you must connect power to the chips to get them to work.

Experiment 1: Implement the following Boolean functions with basic gates (without simplification).

- i. $F = A'B + B'C$
- ii. $F = (A + B)'C + (B + C)' + AC'$

For the above two problems report should cover:

- Experiment No
- Experiment Name
- Objective
- Truth table
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Session 2: Truth tables and simplification using Boolean algebra

Objectives

- To simplify the given expression and to realize it using Basic gates
- To simplify the Boolean expression and to build the logic circuit.
- Given a Truth table to derive the Boolean expressions and build the logic circuit to realize it.

Apparatus

- Analog/Digital Training System
- IC Type 7400 Quadruple 2-input NAND gates
- IC Type 7402 Quadruple 2-input NOR gates
- IC Type 7404 Hex Inverters
- IC Type 7408 Quadruple 2-input AND gates
- IC Type 7432 Quadruple 2-input OR gates
- IC Type 7486 Quadruple 2-input XOR gate

Theory

Chapters 1 & 2 of the textbook (Mano, Ciletti).

Boolean Algebra

Boolean algebra is a mathematical system with two values (zero and one) and three operators: logical AND, logical OR, and logical NOT. The boolean system is closed with respect to these three operations; that is, these operators always produce a boolean result given boolean operands. Logical AND and OR are commutative; you can switch the operands and get identical results. Logical AND and OR are associative; that is, $A \text{ AND } (B \text{ AND } C)$ is equal to $(A \text{ AND } B) \text{ AND } C$. The same is true for logical OR. Logical AND and OR are also distributive; that is, $A \text{ AND } (B \text{ OR } C)$ is equal to $(A \text{ AND } B) \text{ OR } (A \text{ AND } C)$. Similarly, $A \text{ OR } (B \text{ AND } C)$ is equivalent to $(A \text{ OR } B) \text{ AND } (A \text{ OR } C)$. The value one is the identity element with respect to logical AND, the value zero is the identity element with respect to logical OR. For any boolean value A , there is one other value A' that is not equal to A (the inverse of A). $A \text{ OR } A'$ is one and $A \text{ AND } A'$ is zero. These statements form the basic postulates of the boolean algebra system. We can prove all other theorems and facts about the boolean system using this set of postulates.

Postulates and Theorems of Boolean Algebra

| | | |
|-----------------------------|---------------------------|--------------|
| $X + 0 = X$ | $X.1 = X$ | |
| $X + X' = 1$ | $X.X' = 0$ | |
| $X + X = X$ | $X.X = X$ | |
| $X + 1 = 1$ | $X.0 = 0$ | |
| $(X')' = X$ | | Involution |
| $X + Y = Y + X$ | $XY = YX$ | Commutative |
| $X + (Y + Z) = (X + Y) + Z$ | $X(YZ) = (XY)Z$ | Associative |
| $X(Y + Z) = XY + XZ$ | $X + YZ = (X + Y)(X + Z)$ | Distributive |
| $(X + Y)' = X'Y'$ | $(XY)' = X' + Y'$ | DeMorgan |
| $X + XY = X$ | $X(X + Y) = X$ | Absorption |

Experiment 2: Solve the following problems:

i. Simplify the equation using Boolean algebra and implement it.

$$F(A, B, C, D) = ABC'D + A'BCD' + ABC + AB'C'D' + ABD' + AB'C$$

ii. Derive the output equations for a 4-bit binary to gray code converter and implement it.

iii. Derive the output equations for a 4-bit gray code to binary converter and implement it.

For the above three problems the report should cover:

- Experiment No
- Experiment Name
- Objective
- Truth table
- Required equation in minimized form with necessary steps
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Session 3: Truth tables and K-maps

Objectives

- To simplify the Boolean expression using K-maps and to build the logic circuit.

Apparatus

- Analog/Digital Training System
- IC Type 7400 Quadruple 2-input NAND gates
- IC Type 7402 Quadruple 2-input NOR gates
- IC Type 7404 Hex Inverters
- IC Type 7408 Quadruple 2-input AND gates
- IC Type 7432 Quadruple 2-input OR gates
- IC Type 7486 Quadruple 2-input XOR gate

Theory

Chapter 3 of the textbook (Mano, Ciletti).

Experiment 3: Solve the following problems:

- Simplify the equation using K-map and implement it.

$$f(A,B,C,D)=\Sigma(0,1,8,9,15)$$

- Design a digital circuit that can detect a 4-bit prime number.
- Design an odd parity generator circuit for 4-bit data.

For the above three problems the report should cover:

- Experiment No
- Experiment Name
- Objective
- Truth table
- Required equation in minimized form with necessary steps
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Session 4: Combinational Logic Circuits

Objectives

- Design and implement some simple combinational logic circuits.

Apparatus

- Analog/Digital Training System
- IC Type 7400 Quadruple 2-input NAND gates
- IC Type 7402 Quadruple 2-input NOR gates
- IC Type 7404 Hex Inverters
- IC Type 7408 Quadruple 2-input AND gates
- IC Type 7432 Quadruple 2-input OR gates
- IC Type 7486 Quadruple 2-input XOR gate

Theory

Chapter 4 of the textbook (Mano, Ciletti).

Combinational Logic Circuits

Combinational logic circuits are defined as circuits whose outputs depend only on the present value of their inputs.

Design Procedure

- 1 – From the problem statement find the number of input variables and the number of required output variables. Assign them meaningful names.
- 2 – Create a truth table that defines a relationship between input and output.
- 3 – Obtain a simplified Boolean function for each output.
- 4 – Draw the logic diagram.

Experiment 4: Solve the following problems:

- i. Design a 3-bit square circuit.
- ii. Design a 4-bit two's complement generator circuit.
- iii. Design BCD to excess 3 converter circuit.

For the above three problems the report should cover:

- Experiment No
- Experiment Name
- Objective
- Truth table
- Required equation in minimized form with necessary steps
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Session 5: Construction of adders, subtractors, magnitude comparators using basic gates

Objective

- Design and implement adders, subtractors, magnitude comparators using basic gates and test these circuits.

Apparatus

- Analog/Digital Training System
- IC Type 7400 Quadruple 2-input NAND gates
- IC Type 7402 Quadruple 2-input NOR gates
- IC Type 7404 Hex Inverters
 - IC Type 7408 Quadruple 2-input AND gates
- IC Type 7432 Quadruple 2-input OR gates
 - IC Type 7486 Quadruple 2-input XOR gate

Theory

Chapter 4 of the textbook (Mano, Ciletti).

Experiment 5: Solve the following problems:

- Design one bit full adder circuit using basic gates.
- Design one bit full subtractor circuit using basic gates.
- Design two bit magnitude comparator circuit using basic gates.

For the above three problems the report should cover:

- Experiment No
- Experiment Name
- Objective
- Truth table
- Required equation in minimized form with necessary steps
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Session 6: Construction of BCD adder and Adder/Subtractor circuit

Objective

- Design and implement BCD adders and switch controlled adder subtractor circuit.

Apparatus

- Analog/Digital Training System
- IC Type 74283

Theory

Chapter 4 of the textbook (Mano, Ciletti).

Experiment 6: Solve the following problems:

- Design a BCD adder.
- Design a switch controlled 4 bit adder-subtractor circuit.

For the above three problems the report should cover:

- Experiment No
- Experiment Name
- Objective
- Truth table
- Required equation in minimized form with necessary steps with logical explanations
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Midterm Examination

There will be a 40 minutes practical examination in the 7th week, consisting of a circuit you are expected to build, test, and demonstrate that it works. Every student will work alone on the midterm exam.

Session 7: Design using Multiplexers and Decoders

Objective

- Design and implement simple circuits using multiplexers and decoders.

Apparatus

- Analog/Digital Training System
- IC Type 74153
- IC Type 74138

Theory

Chapter 4 of the textbook (Mano, Ciletti).

Experiment 7: Solve the following problems:

1. Implement the following functions using multiplexers:

i. $F = \sum(0,2,4,6)$ [Using 4X1 Mux]

ii. $F = \sum(1,3,5,7,9,11,13,15)$ [Using 8X1 Mux]

2. Implement the following functions using 3X8 decoders and basic gates:

i. $F = \sum(1,3,5,7)$

ii. $F = \sum(0,2,4,6,8,10,12,14)$

For the above three problems the report should cover:

- Experiment No
- Experiment Name
- Objective
- Implementation table (if any)
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Session 8: Design of Sequential circuits

Objective

- Design and implement simple sequential circuits.

Apparatus

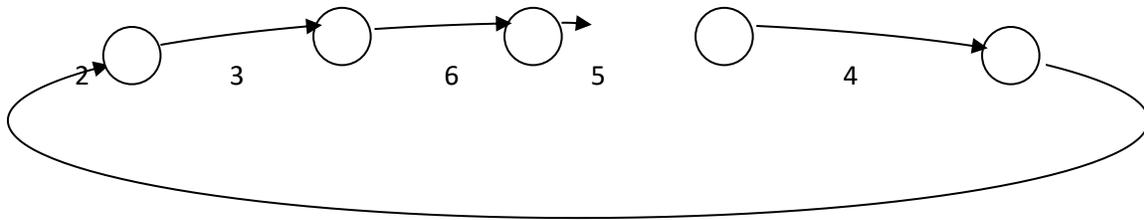
- Analog/Digital Training System
- IC Type 7476

Theory

Chapter 5 and Chapter 6 of the textbook (Mano, Ciletti).

Experiment 8: Solve the following problems:

1. Design a synchronous sequential circuit with the sequence given below using JK FFs:



2. Design a 4 bit odd counter.

For the above three problems the report should cover:

- Experiment No
- Experiment Name
- Objective
- Statetable, Excitation Map, simplified equations
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Session 9: Design of Synchronous and Asynchronous Counters

Objective

- Design and implement simple Synchronous and Asynchronous Counters.

Apparatus

- Analog/Digital Training System
- IC Type 7476

Theory

Chapter 5 and Chapter 6 of the textbook (Mano, Ciletti).

Experiment 9: Solve the following problems:

1. Design a 2 bit switch controlled binary up-down counter using JK FFs.
2. Design a BCD ripple counter.
3. Design MOD-6 Counter.

For the above three problems the report should cover:

- Experiment No
- Experiment Name
- Objective
- Statetable, Excitation Map, simplified equations
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Session 10: Design of Shift Registers

Objective

- Design and implement simple Shift Registers.

Apparatus

- Analog/Digital Training System
- IC Type 7476
- IC Type 74153

Theory

Chapter 6 of the textbook (Mano, Ciletti).

Experiment 10: Solve the following problems:

1. Design a 4 bit Ring Counter
2. Design a 4 bit Johnson Counter.
3. Design a 4 bit universal shift register.

For the above three problems the report should cover:

- Experiment No
- Experiment Name
- Objective
- Working procedure
- Circuit diagram
- Required ICs
- IC pin connections in the diagram
- Discussions

Final Term Examination

There will be a one-hour written examination.

Session 11: Make-up Lab

There will be a make-up lab session at the end of the semester in the 13th week. During that session you will be allowed to make up at most two experiments you have previously worked on. Note that this is an opportunity being afforded to you so that you may work again on an experiment you previously attempted but were unsuccessful. If you did not work on an experiment during a previous lab session you will not be allowed to make it up during the make-up session.