

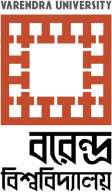
**Lab Manual for**

**CSE 2106 (Digital System Design Lab)**

**Credit: 1, Contact hour: 1.5 Hours Per week**

Department of Computer Science & Engineering

Varendra University Rajshahi, Bangladesh



**Varendra University**

**Department of Computer Science and Engineering**

**CSE 2106**

**Digital Logic Design Lab**

|  |  |
| --- | --- |
| Student ID |  |
| Student Name |  |
| Section |  |
| Name of the Program |  |
| Name of the Department |  |

## Index

[Index 3](#_Toc149606854)

[Varendra University 5](#_Toc149606855)

[COURSE SYLLABUS 5](#_Toc149606856)

[Lab No.: 1 10](#_Toc149606857)

[Logic Gates 10](#_Toc149606858)

[Lab No.: 2 14](#_Toc149606859)

[Expression implementation with basic gates 14](#_Toc149606860)

[Lab No.: 3 16](#_Toc149606861)

[Simplification using Boolean Logic 16](#_Toc149606862)

[Lab No.: 4 18](#_Toc149606863)

[Universality with NAND gate 18](#_Toc149606864)

[ NAND gate as NOT gate: 18](#_Toc149606865)

[ NAND gates as AND gate: 18](#_Toc149606866)

[ NAND gates as OR gate: 18](#_Toc149606867)

[Lab No.: 5 20](#_Toc149606868)

[Universality with NOR gate 20](#_Toc149606869)

[ NOR gate as NOT gate: 20](#_Toc149606870)

[ NOR gates as OR gate: 20](#_Toc149606871)

[ NOR gates as AND gate: 20](#_Toc149606872)

[Lab No.: 6 22](#_Toc149606873)

[Half Adder & Full Adder 22](#_Toc149606874)

[Lab No.: 7 24](#_Toc149606875)

[Half Subtractor & Full Subtractor 24](#_Toc149606876)

[Lab No.: 8 26](#_Toc149606877)

[Encoder Decoder 26](#_Toc149606878)

[Lab No.: 9 29](#_Toc149606879)

[Lab Final Test 29](#_Toc149606880)

**INSTRUCTIONS FOR LABORATORY**

* The experiments are designed to illustrate different expressions based on Boolean algebra.
* Students should come with thorough preparation for the experiment to be
* Students should come in proper dress.
* Students will not be permitted to attend the laboratory unless they bring the practical record fully completed in all respects pertaining to the experiment conducted in the previous
* Work quietly and carefully, and equally share the work with your group.
* Be honest in developing and representing your implementation.
* All presentations of programs and outputs should be neatly and carefully
* If you finish early, spend the remaining time providing support to the team. Come equipped with the necessary documents and other materials related to the lab.
* Handle instruments with care. Report any breakage or faulty equipment to the
* Return all the used instruments   for   conducting   your   experiment   before   leaving the laboratory.

# Varendra University

**Department of Computer Science and Engineering**

## COURSE SYLLABUS

|  |  |  |
| --- | --- | --- |
| 1 | **Faculty** | Faculty of Science & Engineering |
| 2 | **Department** | Department of CSE |
| 3 | **Program** | B.Sc. in Computer Science and Engineering |
| 4 | **Name of Course** | Digital Logic Design Lab |
| 5 | **Course Code** | CSE 2106 |
| 6 | **Year** | Spring, 2025 |
| 7 | **Pre-requisites** | EEE 1132, EEE 1232 |
| 8 | **Status** | Core Course |
| 9 | **Credit Hours** | 1.5 |
| 10 | **Section** |  |
| 11 | **Class Hours** |  |
| 12 | **Class Location** |  |
| 13 | **Name (s) of Academic staff / Instructor(s)** |  |
| 14 | **Contact** |  |
| 15 | **Office** |  |
| 16 | **Counseling Hours** |  |
| 17 | **Text Book** | 1. Digital Logic and Computer Design by Morris Mano 2. Digital Systems: Principles and Application by Ronald J. Tocci |
| 19 | **Equipment & Aids** | 1. Lab Sheet 2. Text Book 3. IC(s) |
| 20 | **Course Rationale** | Understand the principles and methodology of digital logic |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | | design at the gate and switch level, including both combinational and sequential logic elements. Appreciate methods for specifying digital logic, as well as the process by which a high-level specification of a circuit is synthesized into logic design. | | | |
| 21 | **Course Description** | | This course provides a modern introduction to logic design and the basic building blocks used in digital systems, in particular digital computers. It starts with a discussion of combinational logic: logic gates, minimization techniques, arithmetic circuits, and modern logic devices such as field programmable logic gates. The second part of the course deals with sequential circuits: flip-flops, synthesis of sequential circuits, and case studies, including registers. Different representations including truth table, logic gate, timing diagram, switch representation, and state diagram will be discussed. | | | |
| 22 | C**ourse Objectives** | | The course is designed to provide the background of the following topics:   1. Realizing complex logic functions. 2. Different techniques to minimize multiple variables. 3. Design combinational and sequential circuit. | | | |
| 23 | **Learning Outcomes** | | After the successful completion of this course, students will be able,   1. To implement basic and universal gates. 2. To implement different expression with gates. 3. To apply Boolean logic and Karnaugh map. 4. To design different combinational logic circuit. | | | |
| 24 | **Teaching Methods** | | Lecture, Video Demonstration, Problem Solving, Brainstorming, Project Development, Q/A, Project Presentation | | | |
| 25 | **Topic Outline** | |  | | | |
|  | **Class** | **Topics Or Assignments** | | **CLOs** | **Reading Reference** | **Activities** |
|  | **1-2** | Digital Logic Gates | | **I** | Lecture note & textbook | Problem Solving. Question Answer |
|  | **3-4** | Boolean Algebra, Boolean Functions | | **II** | Lecture note and textbook | Problem Solving. Question Answer |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **5-6** | Multilevel NAND circuits | **I** | Lecture note and textbook | Implementation, Problem Solving. Question Answer |
|  | **7-8** | Multilevel NOR circuits | **I** | Lecture note and textbook | Implementation, Problem Solving. Question Answer |
|  | **9-10** | Adder design procedure | **IV** | Lecture note and textbook | Implementation, Problem Solving. Question Answer |
|  | **11-12** | Subtractor design procedure | **IV** | Lecture note and textbook | Implementation, Problem Solving. Question Answer |
|  | **13-14** | Combinational Logic implementation: Encode, Decoder | **IV** | Lecture note and textbook | Implementation, Problem Solving. Question Answer |
|  | **15-16** | Final Lab Examination, Quiz, viva voce |  |  | Problem Solving, Multiple-Choice Question Answer |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 26 | **Assessment Methods** |  | **Assessment type** | | | **Marks** | | |  |
| Attendance | | | (10%) | | |  |
| Lab Quiz | | | (30%) | | |  |
| Continuous assessment | Lap Report (LR) | | (10%) | | |  |
| Regular  assessment | | (10%) | | |  |
| Lab Test (Mid) | | (10%) | | |  |
| Lab Final (Final) | | (20%) | | |  |
| Viva voce | | (10%) | | |  |
| **Total** | | **(100%)** | | |  |
| 27 | **Grading Policy** | **Grading System:** | | | | | | | |
|  | **Numerical Grade** | | **Letter Grade** | | **Grade Point** |  | |
| 80 % and above | | A+ (A Plus) | | 4.00 |
| 75% to less than 80 % | | A (A Regular) | | 3.75 |
| 70 % to less than 75% | | A- (A Minus) | | 3.50 |
| 65% to less than 70% | | B+ (B Plus) | | 3.25 |
| 60% to less than 65% | | B (B Regular) | | 3.00 |
| 55% to less than 60% | | B- (B Minus) | | 2.75 |
| 50% to less than 55% | | C+ (C Plus) | | 2.50 |
| 45% to less than 50% | | C (C Regular) | | 2.25 |
| 40% to less than 45% | | D (D Regular) | | 2.00 |
| Less than 40% | | F (Failure) | | 0.00 |

|  |  |  |
| --- | --- | --- |
| 28 | **Additional Course Policies** | 1. *Lab Reports*   Report on previous Experiment must be submitted before the beginning of new experiment.   1. *Examination*   There will be lab exam in mid and at the end of the semester, that will be closed book.   1. *Unfair means policy*   In case of copying/plagiarism in any of the assessments, the students involved will receive zero marks. Zero Tolerance will be shown in this regard. In case of severe offences, actions will be taken as per university rule. |

|  |  |  |
| --- | --- | --- |
|  |  | 1. *Counseling*   Students are expected to follow the counseling hours posted. In case of emergency/unavoidable situations, students can e-mail me to make an appointment.   1. *Policy for Absence in Class/Exam*   If a student is absent in the class for anything other than medical reasons, he/she will not receive attendance. If a student misses a class for genuine medical reasons, he/she must apply with the supporting documents (prescription/medical report). He/she will then have to follow the instructions given by the instructor for makeup.  In case of absence in the mid/final exam for medical grounds, the student must also get his/her application forwarded by the head of the department before a make-up exam can be taken. It is recommended that the students inform the instructor beforehand through mail/phone call, if they feel that they will miss a class/evaluation due to medical reasons. |
| 29 | **Additional Info** | 1. Academic Calendar Summer 2023:   <https://vu.edu.bd/academics/academic-calendar>   1. Academic Information and Policies: <https://vu.edu.bd/index.php/academics/departments/computer-science-and-engineering/degree/59> 2. Grading and Performance Evaluation: <https://vu.edu.bd/index.php/academics/departments/computer-science-and-engineering/degree/59> |

# Lab No.: 1

## Logic Gates

**AIM:** To study and verify the truth table of logic gates.

**Learning objective:** Identify various ICs and their specification.

**Components required:**

* Logic gates (IC) trainer kit.
* Connecting Wires.
* IC 7400, IC 7402, IC 7404, IC 7408, IC 7432, IC 7486

**Theory:**

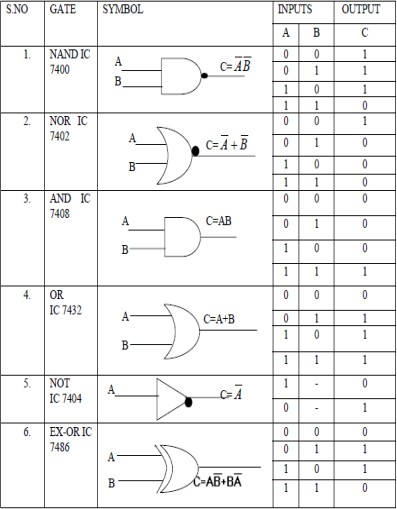
The basic logic gates are the building blocks of more complex logic circuits. These logic gates perform the basic Boolean functions, such as AND, OR, NAND, NOR, Inversion, Exclusive-OR, Exclusive-NOR. Fig. below shows the circuit symbol, Boolean function, and truth. It is seen from the Figures that each gate has one or two binary inputs, A and B, and one binary output, C. The small circle on the output of the circuit symbols designates the logic complement.

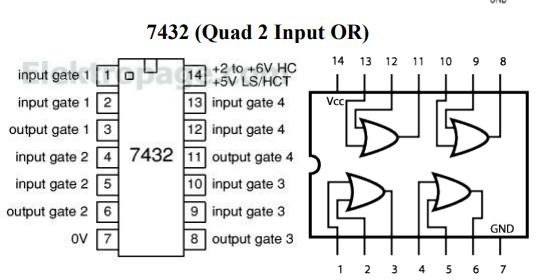
The AND, OR, NAND, and NOR gates can be extended to have more than two inputs. A gate can be extended to have multiple inputs if the binary operation it represents is commutative and associative.

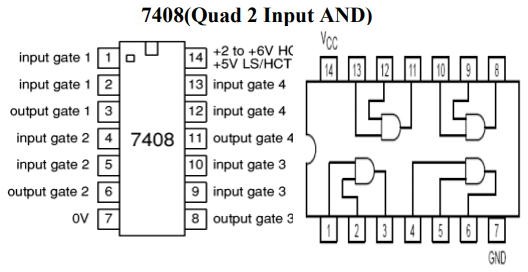
**Procedure:**

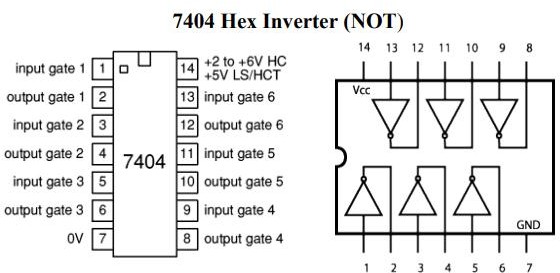
* + Check the components for their working.
  + Insert the appropriate IC into the IC base.
  + Make connections as shown in the circuit diagram.
  + Provide the input data via the input switches and observe the output-on-output LEDs

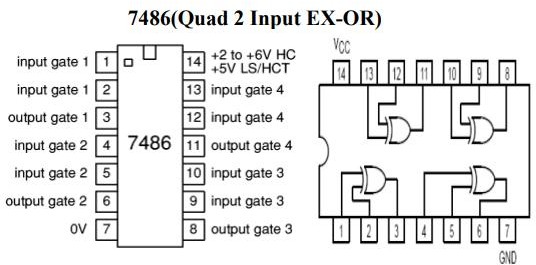
**Logic gates with symbols & input-output:**

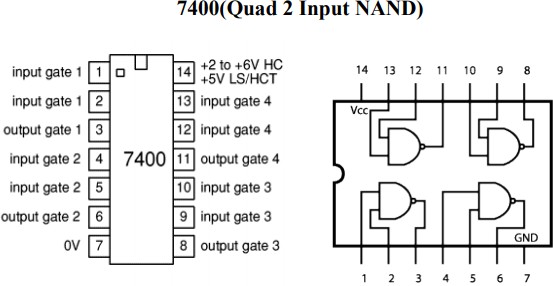


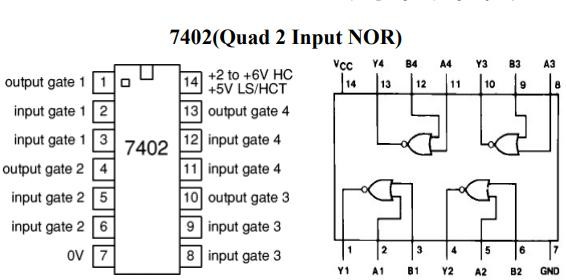












# Lab No.: 2

## Expression implementation with basic gates

**AIM:** To implement the given expression using basic gates.

**Learning objective:** To implement different Boolean expression and to build the logic circuit. Given Truth table to derive the Boolean expressions and build the logic circuit to realize it.

**Components required:**

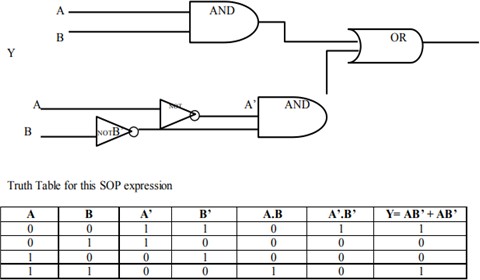
* Logic gates (IC) trainer kit.
* Connecting Wires.
* IC 7404, IC 7408, IC 7432

**Theory:** To form Boolean expression three kinds of gates which called basic gates is needed. Any kinds of Boolean expression can draw using AND, OR & NOT gates. The relationship between the gates can be clearly understood after expressing it through logic circuit.

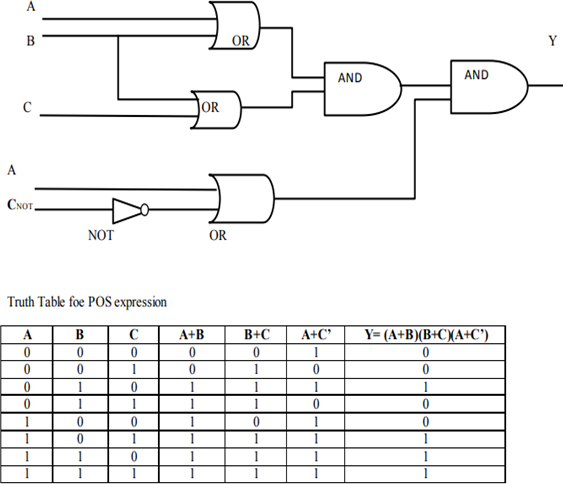
**Procedure:**

* + Check the components for their working.
  + Insert the appropriate IC into the IC base.
  + Make connections as shown in the circuit diagram.
  + Provide the input data via the input switches and observe the output-on-output LEDs

**DIAGRAM AND TRUTH TABLE FOR EXPESSION: AB+A’B’**



**DIAGRAM AND TRUTH TABLE FOR EXPESSION: (A+B) (B+C) (A+C’):**



# Lab No.: 3

## Simplification using Boolean Logic

**AIM:** To simplify the given expression and to implement it using logic gates.

**Learning objective:** 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.

**Components required:**

Logic gates (IC) trainer kit.

Connecting Wires.

IC 7404, IC 7408, IC 7432

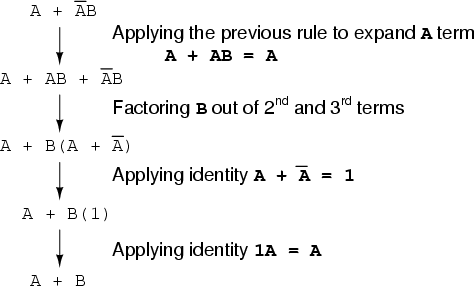
**Theory:** Boolean algebra, a logic algebra, allows the rules used in the algebra of numbers to be applied to logic. It formalizes the rules of logic. Boolean algebra is used to simplify Boolean expressions which represent combinational logic circuits. It reduces the original expression to an equivalent expression that has fewer terms which means that less logic gates are needed to implement the combinational logic circuit.

Canonical Forms (Normal Forms): Any Boolean function can be written in disjunctive normal form (sum of min-terms) or conjunctive normal form (product of max-terms).

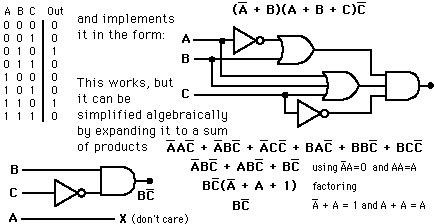
**Procedure:**

* + Check the components for their working.
  + Insert the appropriate IC into the IC base.
  + Make connections as shown in the circuit diagram.
  + Provide the input data via the input switches and observe the output-on-output LEDs

**Sample simplification Example:**



**Simplification example with truth table and logic circuit:**



# Lab No.: 4

## Universality with NAND gate

**AIM:** To construct NOT, AND, OR logic gates using only NAND gate

**Learning objective:** To construct basic gates using universal (NAND) gate.

**Components required:**

* Logic gates (IC) trainer kit.
* Connecting Wires.
* IC 7400

**Theory:** NAND gate is actually a combination of two logic gates: AND gate followed by NOT gate. So, its output is complementing of the output of an AND gate. This gate can have minimum two inputs, output is always one. By using only NAND gates, we can realize all logic functions: AND, OR, NOT, X-OR, X-NOR, NOR. So, this gate is also called universal gate.

**Procedure:**

* + Check the components for their working.
  + Insert the appropriate IC into the IC base.
  + Make connections as shown in the circuit diagram.
  + Provide the input data via the input switches and observe the output-on-output LEDs

### NAND gate as NOT gate:

A NOT produces complement of the input. It can have only one input, tie the inputs of a NAND gate together. Now it will work as a NOT gate. Its output is Y = (A.A)’ = (A)’.

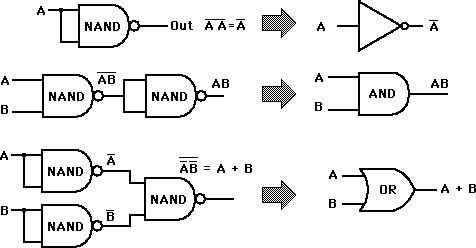
### NAND gates as AND gate:

A NAND produces complement of AND gate. So, if the output of a NAND gate is inverted, overall output will be that of an AND gate. Y = ((A.B)’)’= (A.B).

### NAND gates as OR gate:

From De Morgan’s theorems: (A.B)’ = A’ + B’. Similarly, (A’. B’)’ = A’’ + B’’ = A + B. So, give the inverted inputs to a NAND gate, obtain OR operation at output.

**Sample circuit diagram:**



# Lab No.: 5

## Universality with NOR gate

**AIM:** To construct NOT, AND, OR logic gates using only NOR gate

**Learning objective:** To construct basic gates using universal (NOR) gates.

**Components required:**

Logic gates (IC) trainer kit.

Connecting Wires.

IC 7402

**Theory:** NOR gate is actually a combination of two logic gates: OR gate followed by NOT gate. So, its output is complementing of the output of an OR gate. This gate can have minimum two inputs, output is always one. By using only NOR gates, we can realize all logic functions: AND, OR, NOT, X-OR, X-NOR, NAND. So, this gate is also called universal gate.

**Procedure:**

* + Check the components for their working.
  + Insert the appropriate IC into the IC base.
  + Make connections as shown in the circuit diagram.
  + Provide the input data via the input switches and observe the output-on-output LEDs

### NOR gate as NOT gate:

A NOT produces complement of the input. It can have only one input, tie the inputs of a NOR gate together. Now it will work as a NOT gate. Its output is Y = (A+A)’ = (A)’.

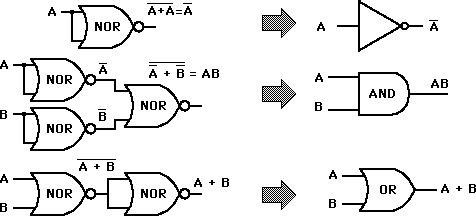
### NOR gates as OR gate:

A NOR produces complement of OR gate. So, if the output of a NOR gate is inverted, overall output will be that of an OR gate. Y = ((A+B)’)’= (A+B).

### NOR gates as AND gate:

From De Morgan’s theorems: (A+B)’ = A’. B’. Similarly, (A’+B’)’ = A’’. B’’ = A. B So, give the inverted inputs to a NOR gate, obtain AND operation at output.

**Sample circuit diagram:**



# Lab No.: 6

## Half Adder & Full Adder

**AIM:** To design and construct half adder and full adder using logic gates and verify the truth table.

**Learning objective:** To construct half adder and full adder using logic gates.

**Components required:**

* Logic gates (IC) trainer kit.
* Connecting Wires.
* IC 7408, IC 7432, IC 7486

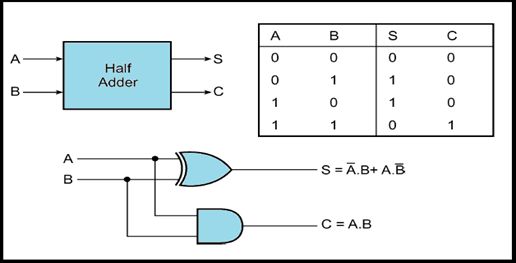
**Theory:** A half adder has two inputs for the two bits to be added and two outputs one from the sum ‘S’ and other from the carry ‘C’ into the higher adder position. Above circuit is called as a carry signal from the addition of the less significant bits sum from the X-OR Gate the carry out from the AND gate.

A full adder is a combinational circuit that forms the arithmetic sum of input; it consists of three inputs and two outputs. A full adder is useful to add three bits at a time but a half adder cannot do so. In full adder sum output will be taken from X-OR Gate, carry output will be taken from OR Gate

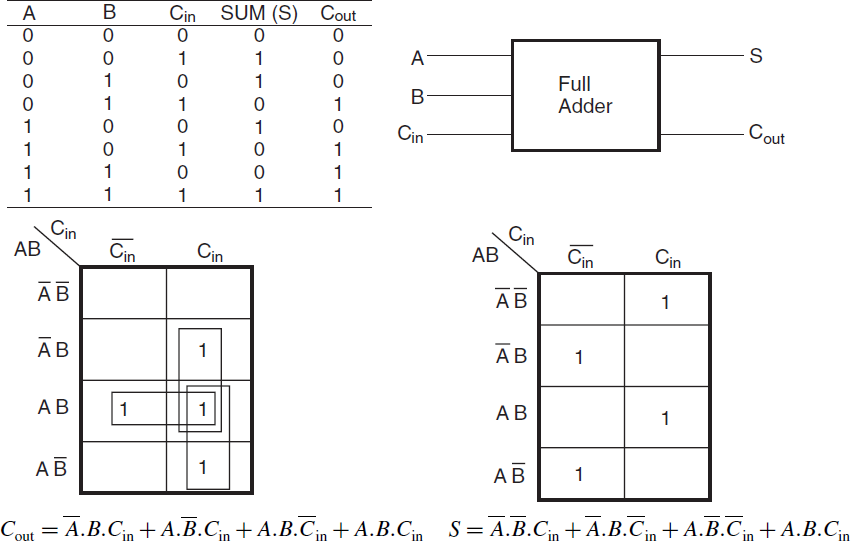
**Procedure:**

* + Check the components for their working.
  + Insert the appropriate IC into the IC base.
  + Make connections as shown in the circuit diagram.
  + Provide the input data via the input switches and observe the output-on-output LEDs.

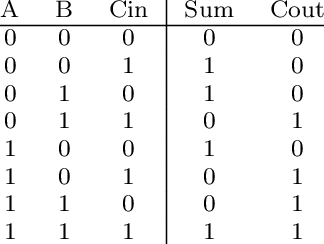
**Sample circuit diagram of half adder:**

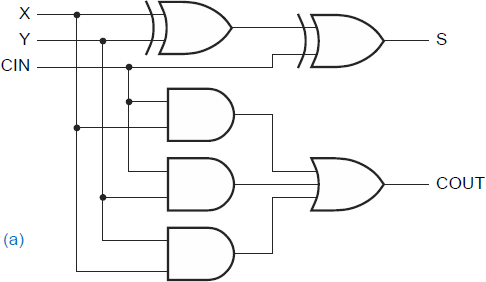


Table, block diagram and simplification procedure of Full Adder circuit:



**Logic circuit diagram of Full Adder:**





# Lab No.: 7

## Half Subtractor & Full Subtractor

**AIM:** To design and construct Half Subtractor & Full Subtractor and verify the truth table**.**

**Learning objective:** To construct half Subtractor & full Subtractor**.**

**Components required:**

* Logic gates (IC) trainer kit.
* Connecting Wires.
* IC 7404, IC 7408, IC 7432, IC 7486

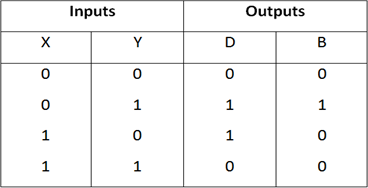
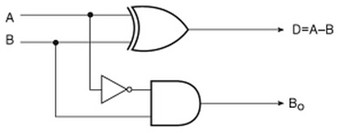
**Theory:** The half subtractor is constructed using X-OR and AND Gate. The half subtractor has two input and two outputs. The outputs are difference and borrow. The difference can be applied using XOR Gate, borrow output can be implemented using an AND Gate and an inverter.

The full subtractor is a combination of X-OR, AND, OR, NOT Gates. In a full subtractor the logic circuit should have three inputs and two outputs. The two half subtractor put together gives a full subtractor. The first half subtractor will be C and A B. The output will be difference output of full subtractor. The expression AB assembles the borrow output of the half subtractor and the second term is the inverted difference output of first X-OR.

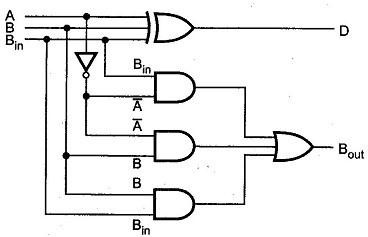
**Procedure:**

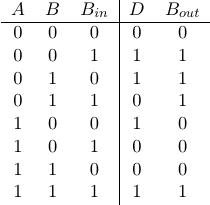
* + Check the components for their working.
  + Insert the appropriate IC into the IC base.
  + Make connections as shown in the circuit diagram.
  + Provide the input data via the input switches and observe the output-on-output LEDs.

**Sample circuit diagram and truth table of half subtractor:**



**Sample circuit diagram and truth table of full subtractor:**





# Lab No.: 8

## Encoder Decoder

**AIM:** Implementation and verification of encoder and decoder.

**Learning objective:** To construct encoder and decoder.

**Components required:**

* Logic gates (IC) trainer kit.
* Connecting Wires.
* IC 7404, IC 7408, IC 7432

**Theory:** Binary decoder, digital circuits such as 1-of-N and seven-segment decoders. Instruction decoder, an electronic circuit that converts computer instructions into CPU control signals.

An encoder is a combinational circuit that converts binary information in the form of a 2Ninput lines into N output lines, which represent N bit code for the input. For simple encoders, it is assumed that only one input line is active at a time.

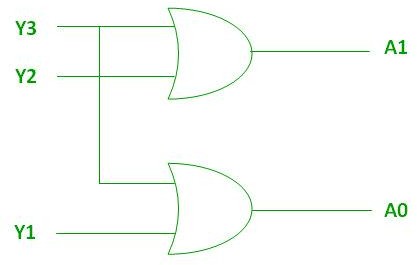
**Procedure:**

* + Check the components for their working.
  + Insert the appropriate IC into the IC base.
  + Make connections as shown in the circuit diagram.
  + Provide the input data via the input switches and observe the output-on-output LEDs.

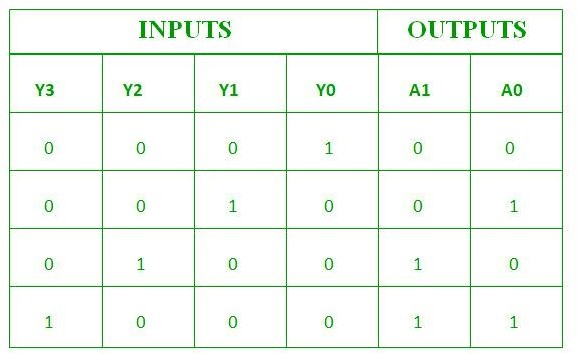
**Sample block diagram of Encoder:**



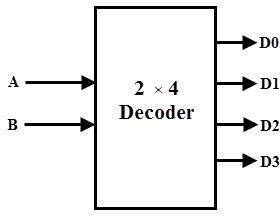
**Sample circuit diagram of Encoder:**



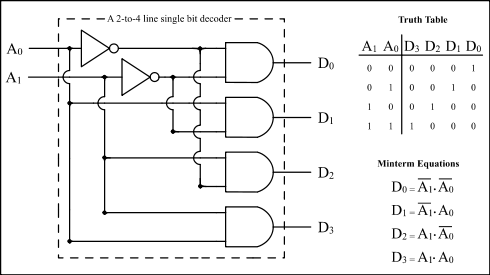
**Sample truth table of Encoder:**



**Sample block diagram of Decoder:**



**Sample truth table and circuit diagram of Decoder:**



# Lab No.: 9

## Lab Final Test

**Task 1:** Quiz

**Task 2:** Implementation of given problem with written answer.

**Task 3:** Viva Voce.

**Task 4:** Lab report evaluation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Assessment and Marks Distribution** | | | | |
|  | **Assessment type** | | **Marks** |  |
| Attendance | | (10%) |
| Lab Quiz | | (30%) |
| Continuous assessment | Lap Report (LR) | (10%) |
| Regular  assessment | (10%) |
| Lab Test (LT) | (10%) |
| Lab Final (LF) | (20%) |
| Viva voce | (10%) |
| **Total** | **(100%)** |

**D.M. Asadujjaman**

Lecturer (Contractual),

Dept. of CSE Varendra University, Rajshahi.

Lecturer

Dept. of CSE Varendra University, Rajshahi.