

DIGITAL ELECTRONICS LABORATORY MANUAL

(ECE 2002L)



For

4thSemester B.Tech

Prepared by:

Ms. Srilakshmi K H

Verified by :

Dr. Rajlv Rajan Singh

Department of Electronics and Communication Engineering

Presidency University

School of Engineering

**Itagalpura, Rajanukunte, Yelahanka, Bangalore-
560064**

MARCH 2022

LIST OF EXPERIMENTS

	NAME	PAGE NO
1	Experiment NO 1: Verify the Logic Gates truth table Level 1: Verify basic logic gates on Digital Logic Trainer kit. Level 2: Construct basic logic gates using universal gates and verify using Digital Logic Trainer kit.	
2	Experiment No. 2: Verify the Boolean Function and Rules Level 1: Verify basic Boolean laws on Digital Logic Trainer kit. Level 2: Construct a circuit to verify De Morgan's Theorem on Digital Logic Trainer kit.	
3	Experiment No. 3: Level 1: Construct and verify the HA/FA logic circuits by using logical gates. Level 2: Construct and verify the HA logic circuits by using Universal logic gates.	
4	Experiment No. 4: Level 1: Construct and verify the Half Subtractor /Full Subtractor logic circuits by using logical gates. Level 2: Construct and verify the Half Subtractor logic circuits by using Universal logic gates.	
5	Experiment No. 5: Construct and verify the combinational logic circuit for given specifications. Level 1: Specifications given in the form of Truth table. Implement using basic gates. Level 2: Specification should be extracted from the given scenario. Implement using universal gates only.	
6	Experiment No. 6: Study of SR and D Flip flops Level 1: Verify the operation of SR and D Flip-Flops on Digital Logic Trainer kit Level 2: Construct and verify a SR Flip Flop using D Flip Flops.	
7	Experiment No. 7: Study of JK Flip-flop and Toggle Flip-Flop. Level 1: Verify the operation of JK Flip-flop and Toggle Flip-Flop on Digital Logic Trainer kit Level 2: Construct and verify a T Flip-Flop using JK Flip-Flop. Experiment No. 8: Construct and verify the sequential logic circuit for given specifications Level 1: Specifications given in the form of Truth table.	

	Level 2: Specification should be extracted from the given scenario.	
9	Experiment No. 9: Write the HDL coding for basic combinational logic circuits Level 1: Gate level Modeling Level 2: Behavioral Modeling	
1 0	Experiment No. 10: Write the HDL coding for basic sequential logic circuit Level 1: Gate level Modeling Level 2: Behavioral Modeling	
	SAMPLE VIVA	



PRESIDENCY UNIVERSITY

(Established under the Presidency University Act, 2013 of the Karnataka Act 41 of 2013)

ACA-2[2022-22] COURSE HAND OUT

SCHOOL: SOE	DEPT.: ECE	DATE OF ISSUE: 11-03-2022
NAME OF THE PROGRAM	: B. Tech. Electronics and Communication Engineering	
P.R.C. APPROVAL REF	:	
SEMESTER/YEAR	: 4th Sem/ 2 nd Year	
COURSE TITLE & CODE	: Digital Electronics & ECE 2002	
COURSE CREDIT STRUCTURE	: 3-2-4	
CONTACT HOURS	: 45 (T) + 30 (P) = 75	
COURSE INSTRUCTOR	: Mrs Srilakshmi K H	
PROGRAM OUTCOMES	:	

Graduates of the B.E Program in Electronics and Communication will have the following abilities:

PO1. Engineering Knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2. Problem Analysis: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3. Design/development of Solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4. Conduct Investigations of Complex Problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5. Modern Tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations

PO6. The Engineer and Society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7. Environment and Sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8. Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9. Individual and Team Work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10. Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11. Project Management and Finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12. Life-long Learning: Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

COURSE PREREQUISITES:

Before attempting this course, the student should have prior knowledge of Basic concepts of number representation, Boolean Algebra

COURSE DESCRIPTION:

The purpose of this course is to enable the students to appreciate the fundamentals of digital logic circuits and Boolean algebra focusing on both combinational and sequential logic circuits. The focus of the course will be to discuss the minimization techniques for making canonical and low-cost implementations. Additionally, this course will create a foundation for future courses of the specialization like MPI, MCA, and Embedded Systems etc., where memories and timer / counter circuits are frequently used. In this course we emphasize on analysis and design of digital electronic circuits using logic gates. The course begins by reviewing the important concepts of number systems and Boolean algebra.

The course also enhances the Design, Implementation and Programming abilities through laboratory assignments. The associated laboratory provides an opportunity to verify the theoretic knowledge.

COURSE OUTCOMES:

On successful completion of the course the students shall be able to

- Discuss the concepts of number systems, Boolean algebra, and logic gates.
- Apply minimization techniques to simplify Boolean expressions.
- Demonstrate the Combinational circuits for a given logic.
- Illustrate the Sequential and programmable logic circuits.
- Implement various combinational logic circuits using gates.
- Verify the performance of various sequential logic circuits using gates and memory elements.

MAPPING OF C.O. WITH P.O.

[H-HIGH, M- MODERATE, L-LOW]

PO CO	1	2	3	4	5	10	12
1	H	H				M	H
2	H	H	M			M	H
3	H	M	H		M	L	L
4	H	M	M	L		M	M

COURSE CONTENT (SYLLABUS):

Module: 1: Fundamentals of Number systems- Boolean algebra and digital logic: Review of Number systems, Number base conversions, complements of numbers, Binary Codes, Boolean theorems and Boolean algebra, Boolean functions- canonical and standard forms, Digital logic gates. **[06 hours] [Knowledge]**

Module: 2: Boolean function simplification: Introduction, two, three, four variable K-Maps, utilizing Don't care conditions. Quine McClusky Method for simplification. Universal Gates (NAND & NOR) Implementations. **[11 hours] [Application]**

Module 3: Combinational Logic circuits: Introduction to Combinational circuits, Analysis, Design procedure, Binary Adder and Subtractor, Magnitude comparator, Multiplexers-Demultiplexers, Decoders, Encoders and Priority Encoders, HDL Models of combinational circuits. **[09 hours] [Application]**

Module 4: Sequential and Programmable logic circuits: Introduction to sequential circuits, Storage elements: latches and flip flops, Characteristic tables and equations, excitation table, Analysis of clocked sequential circuits, Mealy & Moore Models of finite state machines - Registers & Counters - HDL Models of Sequential circuits- ROMs, PLDs & PLAs. Implementation of Digital circuits. **[14 hours] [Application]**

SKILL SETS TO BE DEVELOPED:

- An attitude of enquiry.
- Confidence and ability to tackle new problems.
- Ability to interpret events and results.
- Observe and measure physical phenomena.
- Write reports.
- Select suitable equipment, instrument, materials & software

COURSE CONTENT & TASK SCHEDULE FOR LABORATORY COMPONENT:

Sl. No.	Session Number and Date	Task No	Task	Level 01	Level 2	Number of Lab Sessions required to complete the task	Skills to be developed	Course Outcome to be developed
01	1 23/3/2022	1	Program Integration	--	--	1	--	--
02	1 30/3/2022	2	Introduction to Digital Electronics lab	Familiarization with Lab Equipment and lab components	--	1	Basic idea on kits	1
03	2 6/4/2022	3	Verify the Logic Gates truth table	Verify basic logic gates on Digital Logic Trainer kit.	Construct basic logic gates using universal gates and verify using Digital Logic Trainer kit	1	Practical knowledge on basic gates	1
04	3 13/4/2022	4	Verify the Boolean Function and Rules	Verify basic Boolean laws on Digital Logic Trainer kit.	Construct a circuit to verify De Morgan's Theorem on Digital Logic Trainer kit.	1	Practical knowledge on Boolean laws	2
05	4 20/4/2022	5	Construct and verify the HA/FA logic circuits	By using basic logic and XOR gates and Trainer Kit	By using Universal logic gates and Trainer Kit	1	Practical knowledge to handle HA and FA	2
06	5 27/4/2022	6	Construct and verify the HS/FS logic circuits	By using basic logic and XOR gates and Trainer Kit	By using Universal logic gates and Trainer Kit	1	Practical knowledge to handle HS and FS	2
07	6 4/5/2022	7	Practice of previous experiments	--	--	1	--	--

08	7 11/5/2022	8	Construct and verify the combinational logic circuit for given specifications.	Specifications given in the form of Truth table. Implement using basic gates.	Specification should be extracted from the given scenario. Implement using universal gates only.	1	Practical knowledge to construct and handle CLC	2
09	8 18/5/2022	9	Study of Flip flops	Verify the operation of SR and D Flip-Flops on Digital Logic Trainer kit.	Construct and verify a SR Flip Flop using D Flip Flops.	1	Practical knowledge on FF	3
10	9 25/5/2022	10	Study of JK Flip-flop and Toggle Flip-Flop.	Specifications given in the form of Truth table.	Specification should be extracted from the given scenario.	1	Practical knowledge on JKFF	3
11	10 1/6/2022	11	Construct and verify the sequential logic circuit for given specifications	Specifications given in the form of Truth table.	Specification should be extracted from the given scenario.	1	Practical knowledge to construct SLC	3
12	11 8/6/2022	12	Write the HDL coding for basic sequential logic circuit	Gate level Modelling	Behavioural Modelling	1	Hands on training to write HDL code	4
13	12 15/6/22	13	--	Practice Lab	--	1	--	--
14	13 22/06/22	14	--	Practice Lab	--	1	--	--

DELIVERY PROCEDURE (PEDAGOGY): Lectures will be conducted with aid of Microsoft meeting, or physical or etc. Assignments based on course contents will be given to the students at the end of each unit/topic and will be evaluated at regular interval.

SELF-LEARNING TOPICS: Logic family -TTL, ECL, CMOS, Asynchronous Counter.

REFERENCE MATERIALS:

Text Book(s)

T1: Mano, M. Morris and Ciletti Michael D., "Digital Design", Pearson Education.

Reference Book(s)

R1: Jain, R. P., "Modern Digital Electronics", McGraw Hill Education (India).

R2: Roth, Charles H., Jr and Kinney Larry L., "Fundamentals of logic Design", Cengage Learning.

GUIDELINES TO STUDENTS: Refer to NPTEL Digital Electronics lectures available online.

COURSE SCHEDULE FOR THEORY COMPONENT: (This is a macro level planning. Mention the unit wise expected starting and ending dates along with the tests/assignments/quiz and any other activities) [allot about 75% for delivery, about 10 to 12% for Evaluation Discussion, about 10 to 15% on integrating the learning Modules within the course and to the program]

SL No	Activity	Starting date	Concluding Date	No of Hours
-------	----------	---------------	-----------------	-------------

1	Overview of Course	24-03-2022	24-03-2022	1
2	Module 1			6
3	Module 2			2
4	Test 1	18-04-2022	21-04-2022	2
5	Module 2			9
6	Module 3			9
8	Module 4			7
9	Test 2	23-5-2022	26-5-2022	2
10	Module 4			8
11	Revision			1

COURSE SCHEDULE FOR LABORATORY COMPONENT:

Sl. No.	Activity	Starting Date	Concluding Date	Total Number of Periods
01	Overview of the course	23/8/2022	23/8/2022	1
02	Laboratory Familiarization	23/8/2022	23/8/2022	1
03	Demonstration of first set of Experiments/Skills	30/8/2022	27/9/2022	5
	Conduct of first set of experiments			
04	Demonstration of first set of Experiments/Skills	4/10/2022	22/11/2022	6
	Conduct of second set of experiments			
05	Summary of the Laboratory tasks	29/11/2022	06/12/2022	2
06	End Term Evaluation	13/12/2022	13/12/2022	1

SCHEDULE OF INSTRUCTION FOR THE THEORY COMPONENT:

SL No	Date	Title of the Lesson	Topics to be covered	CO	Delivery Mode	Reference
1.	23/03/2022		Program integration		Lecture	T1
2.	25/03/2022		Course Overview		Lecture	T1
3.	28/03/2022	Fundamentals of Number systems- Boolean algebra and digital logic	Review of Number systems, Number base conversions	1	Lecture	T1
4.	30/03/2022		complements of numbers	1	Lecture	T1
5.	01/04/2022		Binary Codes	1	Lecture	T1
6.	04/04/2022		Boolean theorems and Boolean algebra	1	Lecture	T1
7.	06/04/2022		Boolean functions- canonical and standard forms	1	Lecture	T1
8.	08/04/2022		Digital logic gates.	1	Lecture	T1
9.	09/04/2022		Introduction, two, three variable K-Maps	1	Lecture	T1
10.	09/04/2022		Three variable K-Maps	1	Lecture	T1
11.	11/04/2022		Revision for Test 1	1	Lecture	
12.	13/04/2022		Four variable K-Maps	1	Lecture	T1

13.	15/04/2022	Boolean function simplification	utilizing Don't care conditions	2	Lecture	T1
14.	22/04/2022		utilizing Don't care conditions	2	Lecture	T1
15.	25/04/2022		utilizing Don't care conditions	2		T1
16.	27/04/2022		Quine McClusky Method for simplification	2		T1
17.	29/04/2022		Quine McClusky Method for simplification	2	Lecture	T1
18.	30/04/2022		Quine McClusky Method for simplification	2	Lecture	T1
19.	30/04/2022		NAND & NOR implementations	2	Lecture	T1
20.	02/05/2022		NAND & NOR implementations Module 2 concluded	2	Lecture	T1
21.	04/05/2022	Combinational Logic circuits	Introduction to Combinational circuits, Analysis	2	Lecture	T1
22.	06/06/2022		Combinational circuits Design procedure	2	Lecture	T1
23.	09/05/2022		Binary Adder and Subtractor	2	Lecture	T1
24.	11/05/2022		Magnitude comparator	2	Lecture	T1
25.	13/05/2022		Multiplexers-Demultiplexers	2	Lecture	T1
26.	14/05/2022		Decoders, Encoders and Priority Encoders	2	Lecture	T1
27.	14/05/2022		Decoders, Encoders and Priority Encoders	2	Lecture	T1
28.	16/05/2022		HDL Models of combinational circuits	2	Lecture	T1
29.	18/05/2022		HDL Models of combinational circuits	2	Lecture	T1
30.	20/05/2022	Sequential and Programmable logic circuits	Introduction to sequential circuits	2	Lecture	T1
31.	27/05/2022		Storage elements: latches and flip flops	2	Lecture	T1
32.	28/05/2022		Storage elements: latches and flip flops	2	Lecture	T1
33.	28/05/2022		Characteristic tables and equations, excitation table	2		T1
34.	30/05/2022		Analysis of clocked sequential circuits	2		T1
35.	01/06/2022		Mealy & Moore Models of finite state machines	2	Lecture	T1
36.	03/06/2022		Revision for Test 2	2	Lecture	T1
37.	06/06/2022		Mealy & Moore Models of finite state machines	2	Lecture	T1
38.	08/06/2022		Registers	2	Lecture	T1
39.	10/06/2022		Counters	2	Lecture	T1
40.	11/06/2022		HDL Models of Sequential circuits	2	Lecture	T1

41.	11/06/2022		ROMs, PLDs	2	Lecture	T1
42.	13/06/2022		PLDs & PLAs	2	Lecture	T1
43.	15/06/2022		Implementation of Digital circuits	2	Lecture	T1
44.	17/06/2022		Implementation of Digital circuits	2	Lecture	T1
45.	20/06/2022		Revision	2	Lecture	T1

ASSESSMENT SCHEDULE FOR THEORY COMPONENT:

Sl.no	Assessment type	Contents	Course outcome Number	Duration In Hours	Marks	Weightage	Venue, & TIME	DATE
1.	Test-1	Module-1,2 (two topics)	1, 3	1 hr.	30	15%	18-4-2022 to 21-4-2022	
2.	Test-2	Module-2 module-3	2	1 hr.	30	15%	23-5-2022 to 26-5-2022	
3.	End Term Final Examination - Theory	Module-1,2,3,4,	1,2,3	3 hr.	60	30%	27-6-2022 to 9-7-2022	
	Total				120	60%		

ASSESSMENT DETAILS FOR LABORATORY COMPONENT:

Sl. No.	Assessment type [Include here assessment method for self-learning component also]	List of Tasks	Course outcome Number	Duration In Hours	marks	weightage	Venue, & TIME	DATE
1	Laboratory Work, Practical exercises, conducted in every session including Laboratory records	Lab records, execution, viva of each lab session	--	--	40	20%	As per schedule	
2	End Term Final --Lab	Experiment 1-10	1,2,3	2 hr.	40	20%	11-7-2022	
	Total				80	40%		

ASSESSMENT MATRIX FOR DAILY TASK EVALUATION FOR LABORATORY COMPONENT:

Sl. No.	Task No.	Marks for activity 01 [Mention the activity]	Marks for activity 02 [Mention the activity]	Marks for activity 03 [Mention the activity]	Total Marks
1	1	4M [Record writing]	3M [Viva]	3M [Execution]	10M

COURSE CLEARANCE & EVALUATION CRITERIA:

A minimum of 75% attendance is required to attend the end term exam. Make-up policy will be only as per academic regulation. Minimum 30% of the marks is must in the overall course.

MAKEUP POLICY:

If the student misses an evaluation component, he/she may be granted a make-up. In case of an absence that is foreseen, make-up request should be personally made to the Instructor-in-Charge, well ahead of the scheduled evaluation

component. Reasons for unanticipated absence that qualify a student to apply for make-up include medical emergencies or personal exigencies. In such an event, the student should contact the Instructor-in-Charge as soon as practically possible.

CONTACT TIMINGS IN THE CHAMBER FOR ANY DISCUSSIONS:

Prior appointment needed that has to be taken by sending request over email, at least 24 hours early.

SAMPLE THOUGHT PROVOKING QUESTIONS:

SL NO	QUESTION	MARKS	CO NO.	BLOOM'S LEVEL
1.	Raj is an engineering student. In his mini project, he has to design a circuit which has three inputs A, B, C and Two outputs. The circuit performing the summations of all input and produce sum and carry output. But Raj has only 3 into 8 line Decoder IC. Give the truth table and circuit diagram for his project with available Decoder:	10	CO1	Comprehension
2.	A student wants to design a digital logic switching function which is described by the following Boolean Function in SoP, $F(A,B,C,D)=\sum(1,3,4,11,12,13,14,15)$. But he has provided with only 8x1 MUX. Guide the student to design the switching function using MUX only	10	CO3	Comprehension
3.	A engineering student during his mini project, he has to design a circuit which has three inputs A, B, C and Two outputs. The circuit performing the operations of Full Subtractor. But Rahim has only two number of Half Subtractor integrated circuit and OR gate. Guide him to design the Full Subtractor by using Half Subtractor.	10	CO 1	Comprehension
4.	A state machine which is represented by 4bit. Initially Binary combinations are used to refer the states. For effective power consumption needs minimum changes when it moves from one state to another state. Gray code is also known as minimum change code. Design a circuit to convert 4bit Binary to Gray code conversion.	10	CO 4	Comprehension

Target set for course Outcome attainment:

Sl. No	CO No.	Course Outcomes	Target set for attainment in percentage
1.	CO1	Discuss various number system.	50%
2.	CO2	Demonstrate simplification of boolean function.	50%
3.	CO3	Practice several combinational logic circuits	50%
4.	CO4	Sketch sequential and programmable logic circuits	50%

Signature of the course Instructor: Mrs Srilakshmi K H

This course has been duly verified Approved by the D.A.C.

Signature of the Chairperson D.A.C.

Course Completion Remarks & Self-Assessment. [This has to be filled after the completion of the course]

[Please mention about the course coverage details w.r.t. the schedule prepared and implemented. Any specific suggestions to incorporate in the course content. Any Innovative practices followed and its experience. Any specific suggestions from the students about the content, Delivery, Evaluation etc.]

Sl.no.	Activity As listed in the course Schedule	Scheduled Completion Date	Actual Completion Date	Remarks

Any specific suggestion/Observations on content/coverage/pedagogical methods used etc.

Course Outcome Attainment:

Sl.no	C.O. No.	Course Outcomes	Target set for attainment percentage	Actual Attainment In Percentage	C.O.	Remarks on attainment & Measures to enhance the attainment
01	CO1	Discuss various number system.	50%			
02	CO2	Demonstrate simplification of boolean function.	50%			
03	CO3	Practice several combinational logic circuits	50%			
04	CO4	Sketch sequential and programmable logic circuits	50%			

Name and signature of the Faculty member:

BLOOM'S TAXONOMY

Learning Outcomes Verbs at Each Bloom Taxonomy Level to be used for writing the course Outcomes.

Cognitive Level	Illustrative Verbs	Definitions
Knowledge	arrange, define, describe, duplicate, identify, label, list, match, memorize, name, order, outline, recognize, relate, recall, repeat, reproduce, select, state	remembering previously learned information
Comprehension	classify, convert, defend, discuss, distinguish, estimate, explain, express, extend, generalize, give example(s), identify, indicate, infer, locate, paraphrase, predict, recognize, rewrite, report, restate, review, select, summarize, translate	grasping the meaning of information
Application	apply, change, choose, compute, demonstrate, discover, dramatize, employ, illustrate, interpret, manipulate, modify, operate, practice, predict, prepare, produce, relate schedule, show, sketch, solve, use write	applying knowledge to actual situations
Analysis	analyze, appraise, breakdown, calculate, categorize, classify, compare, contrast, criticize, derive, diagram, differentiate, discriminate, distinguish, examine, experiment, identify, illustrate, infer, interpret, model, outline, point out, question, relate, select, separate, subdivide, test	breaking down objects or ideas into simpler parts and seeing how the parts relate and are organized
Synthesis	arrange, assemble, categorize, collect, combine, comply, compose, construct, create, design,	rearranging component ideas into a new whole

	develop, devise, explain, formulate, generate, plan, prepare, propose, rearrange, reconstruct, relate, reorganize, revise, rewrite, set up, summarize, synthesize, tell, write	
Evaluation	appraise, argue, assess, attach, choose, compare, conclude, contrast, defend, describe, discriminate, estimate, evaluate, explain, judge, justify, interpret, relate, predict, rate, select, summarize, support, value	making judgments based on internal evidence or external criteria

GENERAL INSTRUCTIONS

Lab is used for academic purposes. Therefore, a quiet atmosphere is required and is to be maintained.

Maintain silence and complete your exercises on time.

- Eatables are strictly prohibited inside the lab.
- Use of mobile phones inside the lab is strictly prohibited.
- Ask lab instructors if you are not sure about what to do with the exercises.
- Always maintain awareness of the surrounding activities and walk in aisles to the extent possible.
- Maintain clean and orderly laboratory manual/data sheet.
- After the class, before you leave the laboratory, log out from your account and arrange the chairs in proper place.
- Before coming to the laboratory, complete the evaluation of the previous week's exercises with the faculty signature on the contents page.
- Prior permission is needed to bring Laptop inside the Lab
- Unauthorized copying and or installing of unauthorized software are strictly prohibited.
- Tampering with the hardware or software settings is strictly prohibited.
- Visitors are required to take prior permission for visiting the lab.
- Do not leave your personal belongings inside the Lab

SAFETY PRECAUTIONS

Safety first

- Self-Protection
- Others Protection
- Equipment Protection

Plug and cables

- Access to plug and network ports must be kept free. Cables should be kept neat and routed so as to prevent entanglement. Double adapters should not be used and the use of plug boards should be avoided whenever possible.

Accident Reporting

- All accidents or near misses must be reported to the administrator by those concerned or involved immediately.

Monitoring

- Safety issues are subject to continual review and monitoring. Additionally safety inspection of all parts of the Lab is conducted annually to review, highlight and access potential risks.

Responsibility

- While it is the Department's responsibility to ensure, as far as possible, a safe working environment, safe working practices and adequate training, it is the responsibility of all staff, students and visitors to care for their own safety and the safety of others. This includes, but is not limited to:-
 - i. Maintaining safe working practices.
 - ii. Identifying possible hazards and bringing these promptly to the attention of those responsible.
 - iii. Undertaking any necessary safety precautions
 - iv. Being familiar with appropriate emergency procedures including knowledge of :
 - Appropriate escape routes;
 - Location of fire extinguishers;

TOOLS REQUIRED

- TMI Systems –Digital IC trainer kit
- ICs (7400,7402,7410,7432, 7486, 7476, 74138, 74151, 7476, 7495 etc.,)
- Patch Cards

INTRODUCTION

Digital concepts applied to electronics give rise to field of Digital electronics. Digital circuits form the backbone of modern day gadgets like cell phone, digital cameras, GPS displays, etc. since all these devices use information which is digital in nature. Digital systems find application in modern day traffic systems, control systems, weather forecasting systems, and internet, etc.

One of the reasons for widespread application of digital systems is use of Digital computers in applications which provides users with flexibility as any change can be incorporated with the change in system software thus reducing cost which also is an additional advantage. Discrete Information used by digital systems is represented in form of signals which can be classified as Discrete or Continuous signals and systems can be classified as Analog and digital systems.

Discrete information is a finite set like outcome of throwing dice with the output possibilities as 1, 2,3,4,5 or 6. The discrete signals used in electronics systems have two discrete values either 0 or 1. In other words it can be said that outcome can be mapped to two outcomes either 0 or 1.

Each binary digit 0 and 1 is called as a bit and group of bits that represent the information in digital form are called binary codes. Most of the information is in analog form and to represent it in digital form it has to be converted and the device used for this known as Analog to Digital converter (ADC) and at the receiver side the same information has to be converted to analog form using Digital to Analog Converter (DAC).

ADVANTAGES OF DIGITAL SYSTEMS:

Below are the reasons for moving towards Digital systems

1. Ease of programmability

The digital systems can be used for different applications by simply changing the program without additional changes in hardware.

2. Reduction in cost of hardware

The cost of hardware gets reduced by use of digital components and this has been possible due to advances in IC technology. With ICs the number of components that can be placed in a given area of Silicon are increased which helps in cost reduction.

3. High speed

Digital processing of data ensures high speed of operation which is possible due to advances in Digital Signal Processing.

4. High Reliability

Digital systems are highly reliable one of the reasons for that is use of error correction codes.

5.Design is easy

The design of digital systems which require use of Boolean algebra and other digital techniques is easier compared to analog designing.

5. **Result can be reproduced easily** Since the output of digital systems unlike analog systems is independent of temperature, noise, humidity and other characteristics of components the reproducibility of results is higher in digital systems than in analog systems.

Experiment-1

Aim: Verify the Logic Gates truth table

Level 1: Verify basic logic gates on Digital Logic Trainer kit.

Level 2: Construct basic logic gates using universal gates and verify using Digital Logic Trainer kit.

LOGIC GATES

Digital systems are said to be constructed by using logic gates. The basic gates are the AND, OR, NOT gates. The basic operations are described below with the aid of tables in the following, called truth tables.

(i) AND GATE



Fig.1: AND Gate

2 Input AND gate		
A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

Table 1: Truth Table of AND Gate

The AND gate is an electronic circuit that gives a **high** output (1) only if **all** its inputs are high. A dot (.)

is used to show the AND operation i.e. A.B. Bear in mind that this dot is sometimes omitted i.e. AB

(ii) OR GATE



Fig.2: OR Gate

2 Input OR gate		
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

Table 2: Truth Table of OR Gate

The OR gate is an electronic circuit that gives a high output (1) if **one or more** of its inputs are high. A

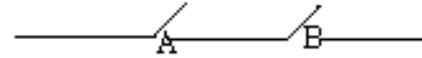
plus (+) is used to show the OR operati

Truth table for AND,

A	B	Z
0	0	0
0	1	0
1	0	0
1	1	1

$$Z = A \cdot B$$

Switch realisation of AND:



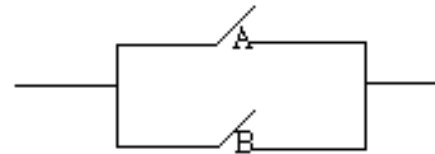
Continuity occurs only when both A AND B are closed.

Truth table for OR,

A	B	Z
0	0	0
0	1	1
1	0	1
1	1	1

$$Z = A + B$$

Switch realisation of OR:

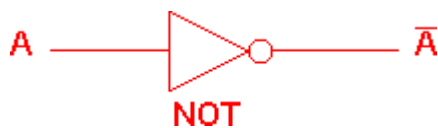


Continuity if A or B or both are closed.

Table 3: Truth Table of AND Gate and OR Gate

Fig.3: AND Gate and OR Gate switch connection

(iii) NOT GATE



NOT gate	
A	\bar{A}
0	1
1	0

Fig.4: NOT Gate

Table 4: Truth Table of NOT Gate

The NOT gate is an electronic circuit that produces an inverted version of the input at its output. It is also known as an *inverter*. If the input variable is A, the inverted output is known as NOT A. This is also shown as A', or A with a bar over the top, as shown at the outputs.

Another useful gate used in the digital logic circuits is EXOR gate.

(iv) EXOR GATE



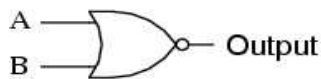
2 Input EXOR gate		
A	B	$A \oplus B$
0	0	0
0	1	1
1	0	1
1	1	0

Fig.5: EXOR Gate

Table 5: Truth Table of EXOR Gate

The '**Exclusive-OR**' gate is a circuit which will give a high output if **either, but not both**, of its two inputs are high. An encircled plus sign (\oplus) is used to show the EOR operation.

(v) NOR GATE



A	B	Output
0	0	1
0	1	0
1	0	0
1	1	0

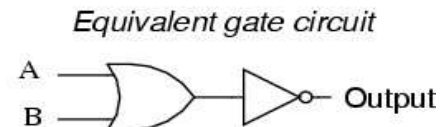
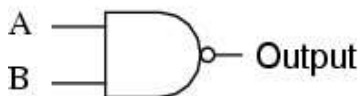


Fig.6: NOR Gate

Table 6: Truth Table of NOR Gate

In NOR gate, the output is true if BOTH input A and input B are NOT true, giving the Boolean Expression of $\text{Out} = \text{NOT} (A \text{ OR } B)$.

(vi) NAND GATE



A	B	Output
0	0	1
0	1	1
1	0	1
1	1	0

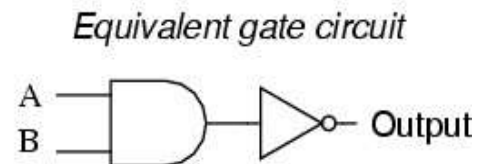


Fig.7: NAND Gate

Table 7: Truth Table of NAND Gate

In NAND gate, the output Q is true if BOTH input A and input B are NOT true, giving the Boolean Expression of $Q = \text{NOT} (A \text{ AND } B)$.

(a) TRANSFORMATION OF NOR GATE INTO BASIC LOGIC GATES:

1. Construction of NOT Gate using NOR Gate:
2. Construction of OR Gate using NOR Gate:
3. Construction of AND Gate using NOR Gate:

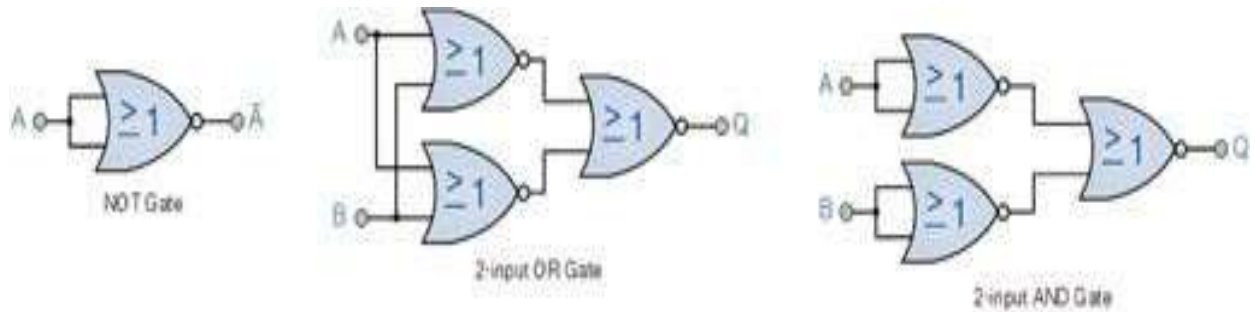


Fig.8: Transformation of NOR Gate into Basic Logic Gates

(b) TRANSFORMATION OF NAND GATE INTO BASIC LOGIC GATES:

1. Construction of NOT Gate using NAND Gate:
2. Construction of OR Gate using NAND Gate:
3. Construction of AND Gate using NAND Gate:

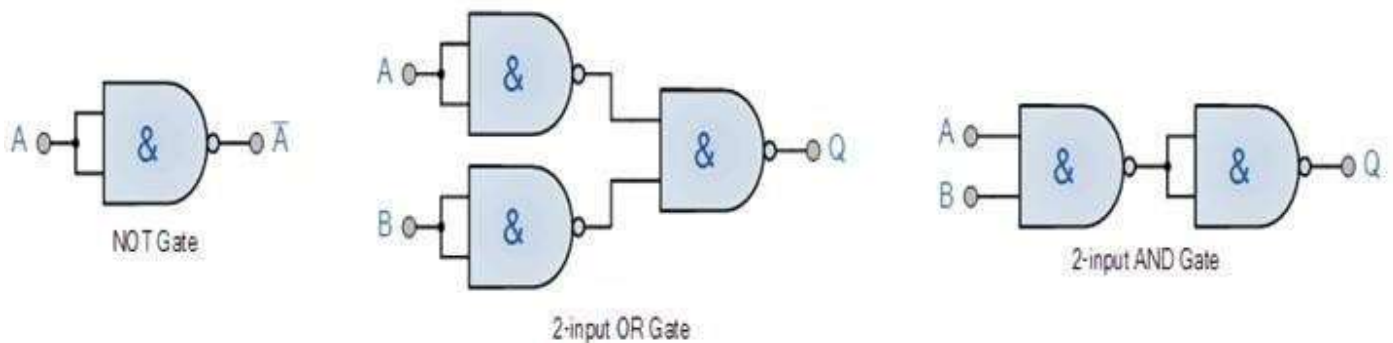
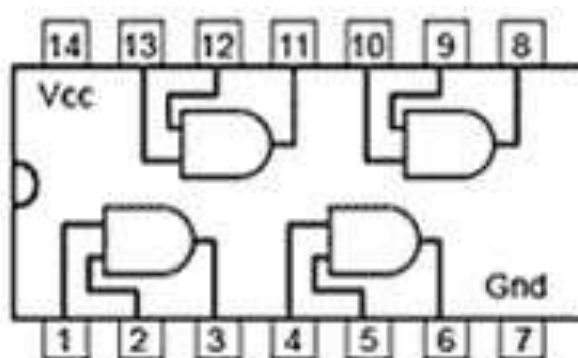


Fig.9: Transformation of NAND Gate into Basic Logic Gates

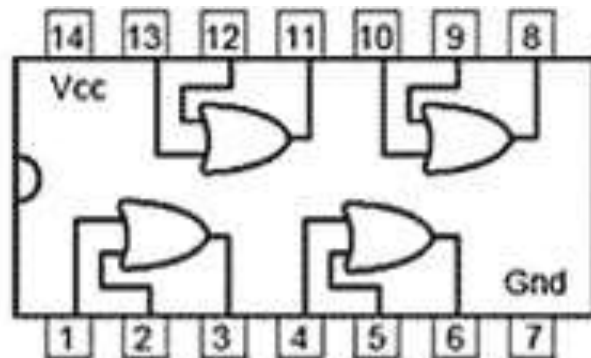
(c) PROCEDURE FOR CONDUCTION OF EXPERIMENT:

1. Design the logic diagrams as per the problem statement.
2. Check the ICs for proper working.
3. Connect the gates as per the pin numbers required.
4. Observe the output by changing the input as per the requirement.
5. Note down the results and verify the truth table.

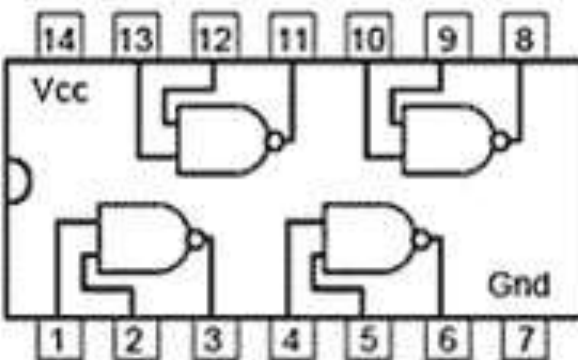
(d) Pin Diagrams of Basic Gates ICs:



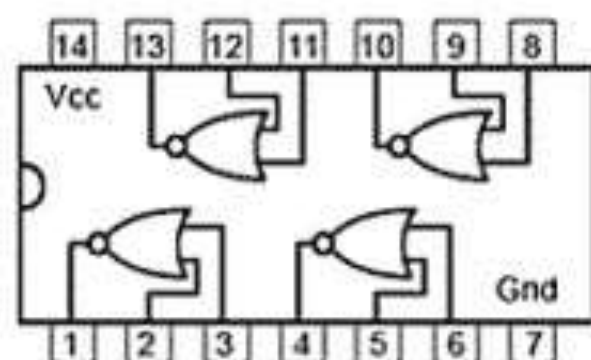
7408 Quad 2 input
AND Gates



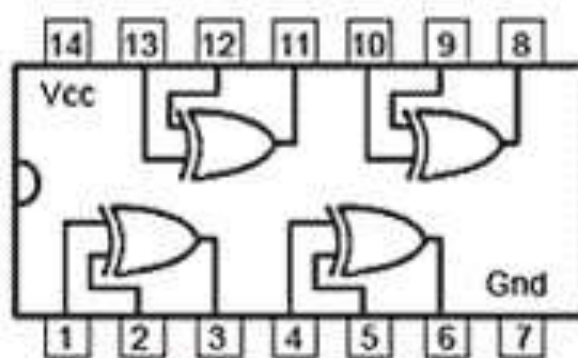
7432 Quad 2 input
OR Gates



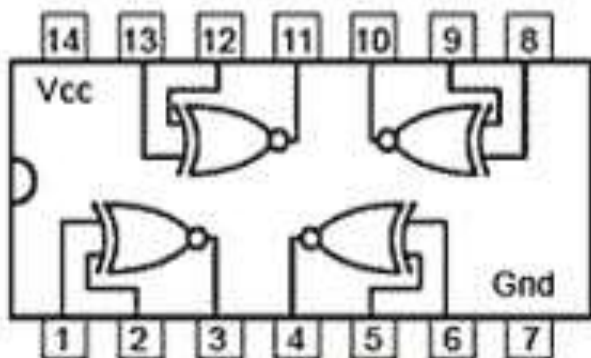
7400 Quad 2 input
NAND Gates



7402 Quad 2 input
NOR Gates



7486 Quad 2 input
XOR Gates



747266 Quad 2 input
XNOR Gates

Experiment No. 2:

AIM: Verify the Boolean Function and Rules

Level 1: Verify basic Boolean laws on Digital Logic Trainer kit.

Level 2: Construct a circuit to verify De Morgan's Theorem on Digital Logic Trainer kit.

Theory: A set of rules or Laws of Boolean Algebra expressions have been invented to help reduce the number of logic gates needed to perform a particular logic operation resulting in a list of functions or theorems known commonly as the Laws of Boolean Algebra. Boolean Algebra is the mathematics we use to analyse digital gates and circuits. We can use these "Laws of Boolean" to both reduce and simplify a complex Boolean expression in an attempt to reduce the number of logic gates required. Boolean Algebra is therefore a system of mathematics based on logic that has its own set of rules or laws which are used to define and reduce Boolean expressions. The variables used in Boolean Algebra only have one of two possible values, a logic "0" and a logic "1" but an expression can have an infinite number of variables all labelled individually to represent inputs to the expression, For example, variables A, B, C etc, giving us a logical expression of $A + B = C$, but each variable can ONLY be a 0 or a 1. Examples of these individual laws of Boolean, rules and theorems for Boolean Algebra are given in the following table.

LEVEL-1: Verify Boolean laws on trainer kit using truth table with the help of Basic Gates

► Associative law (Basic gate)

► $(A.B).C=A.(B.C)$

► $A+(B+C)=(A+B)+C$

► Distributive law (Basic gate)

► $A.(B+C)=(A.B)+(A.C)$

► Associative law (Basic gate)

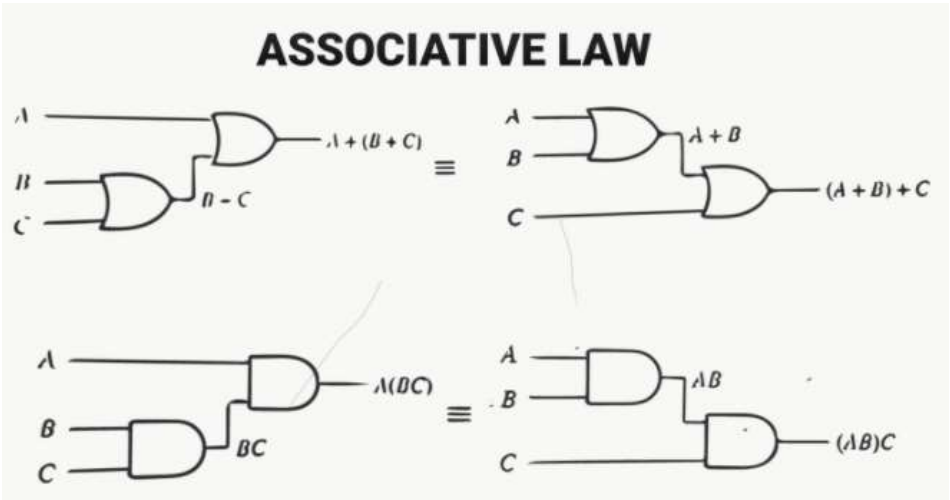
TRUTH TABLE: $(A.B).C=A.(B.C)$

A	B	C	AB	(AB)C	BC	A(BC)
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

► $A+(B.C)=(A+B).C$

A	B	C	B.C	A+(B.C)	(A+B).C	(A+B).C
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

LOGICAL CIRCUIT:



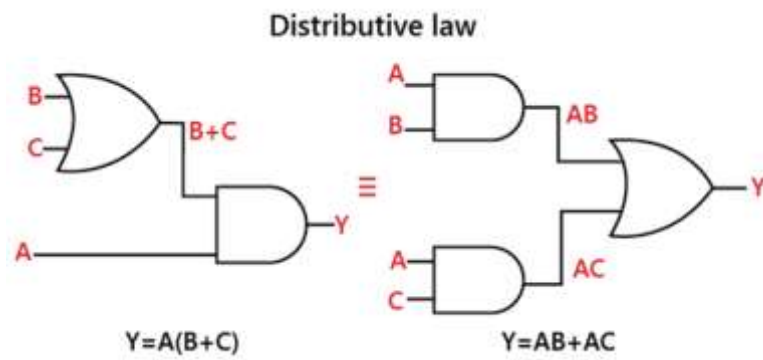
► **Distributive law (Basic gate)**

► $A.(B+C)=(A.B)+(A.C)$

TRUTH TABLE:

A	B	C	A(B+C)	(A.B)	(A.C)	(A.B)+(A.C)
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

LOGICAL CIRCUIT:



LEVEL-2

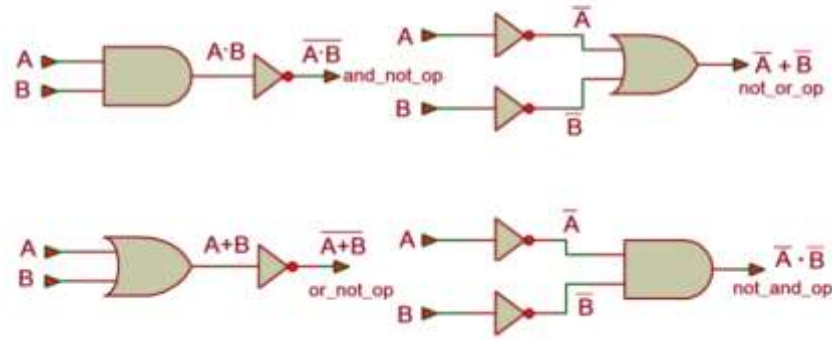
Verify De Morgan's on trainer kit using truth table with the help of logical gates

- $(A.B)' = A' + B'$
- $(A+B+C)' = A' . B' . C'$

Truth Table to prove De Morgan's Theorem:-

A	B	\bar{A}	\bar{B}	$A+B$	$A \cdot B$	$\overline{A+B}$	$\bar{A} \cdot \bar{B}$	$\overline{A \cdot B}$	$\bar{A} + \bar{B}$
0	0	1	1	0	0	1	1	1	1
0	1	1	0	1	0	0	0	1	1
1	0	0	1	1	0	0	0	1	1
1	1	0	0	1	1	0	0	0	0

Digital Circuit



RESULT:

Experiment No. 3:

Aim:

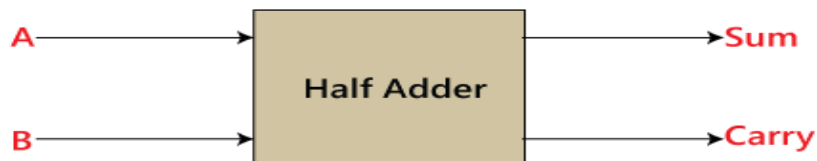
Level 1: Construct and verify the HA/FA logic circuits by using logical gates.

Level 2: Construct and verify the HA logic circuits by using Universal logic gates.

Theory:

A half adder is a **type of adder, an electronic circuit that performs the addition of numbers**. The half adder is able to add two single binary digits and provide the output plus a carry value. It has two inputs, called A and B, and two outputs S (sum) and C (carry).

Block diagram



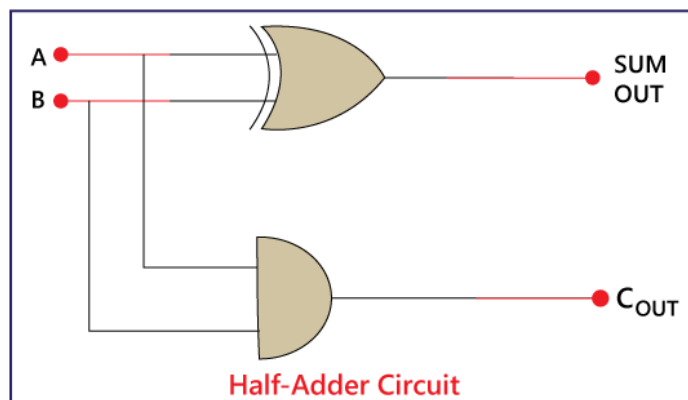
Truth Table

Inputs		Outputs	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

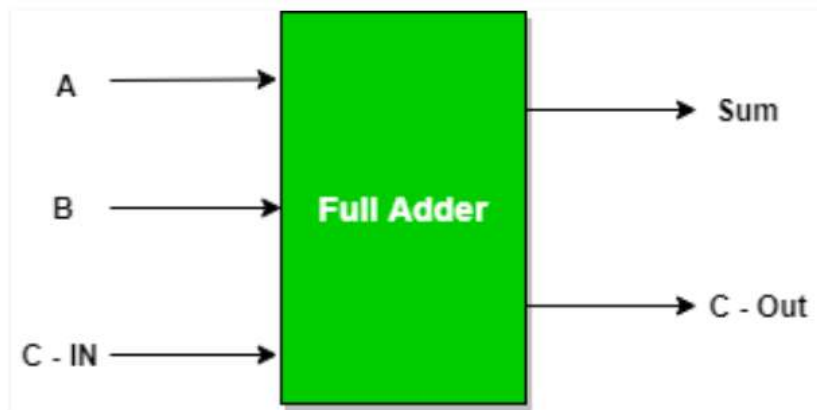
Logical expressions:

- $\text{Sum} = A'B + AB' = A \text{ XOR } B$
- $\text{Carry} = AB$

Circuit implementation:



FULL ADDER CIRCUIT: Full Adder is the adder which adds three inputs and produces two outputs. The first two inputs are A and B and the third input is an input carry as C-IN. The output carry is designated as C-OUT and the normal output is designated as S which is SUM.



Full Adder Truth Table:

Inputs			Outputs	
A	B	C – IN	Sum	C - Out
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Logical Expression for SUM:

$$\begin{aligned}
 &= A' B' C\text{-IN} + A' B C\text{-IN}' + A B' C\text{-IN}' + A B C\text{-IN} \\
 &= C\text{-IN} (A' B' + A B) + C\text{-IN}' (A' B + A B') \\
 &= C\text{-IN} \text{ XOR } (A \text{ XOR } B) \\
 &= (1,2,4,7)
 \end{aligned}$$

Logical Expression for C-OUT:

$$= A' B C\text{-IN} + A B' C\text{-IN} + A B C\text{-IN}' + A B C\text{-IN}$$

$$= A B + B C\text{-IN} + A C\text{-IN}$$

$$= (3,5,6,7)$$

Another form in which C-OUT can be implemented:

$$= A B + A C\text{-IN} + B C\text{-IN} (A + A')$$

$$= A B C\text{-IN} + A B + A C\text{-IN} + A' B C\text{-IN}$$

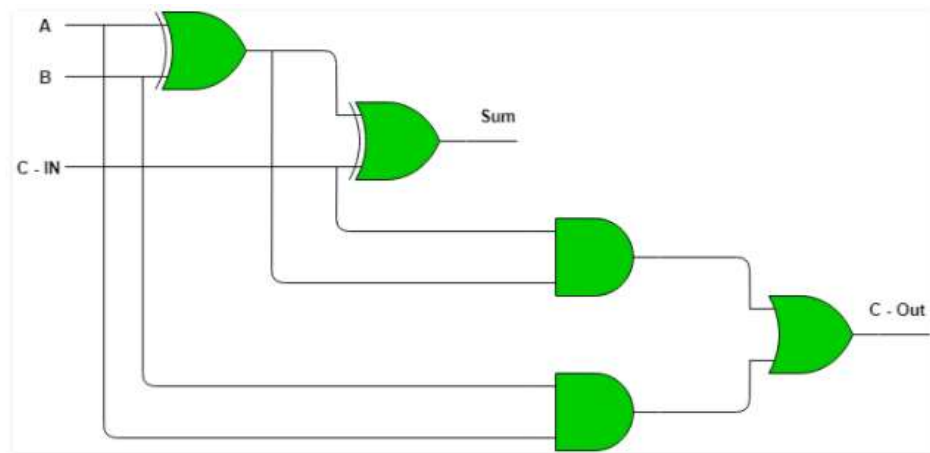
$$= A B (1 + C\text{-IN}) + A C\text{-IN} + A' B C\text{-IN}$$

$$= A B + A C\text{-IN} + A' B C\text{-IN}$$

$$= A B + A C\text{-IN} (B + B') + A' B C\text{-IN}$$

$$= A B C\text{-IN} + A B + A B' C\text{-IN} + A' B C\text{-IN}$$

LOGICAL CIRCUIT:

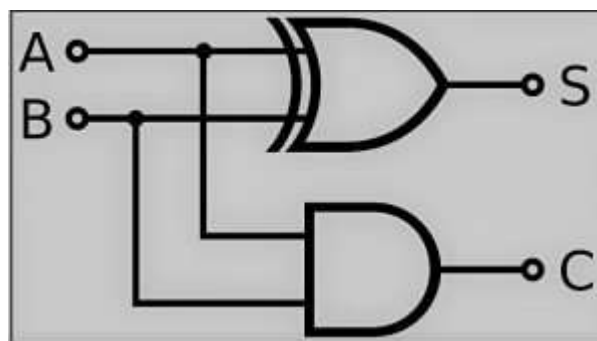


Full Adder logic circuit.

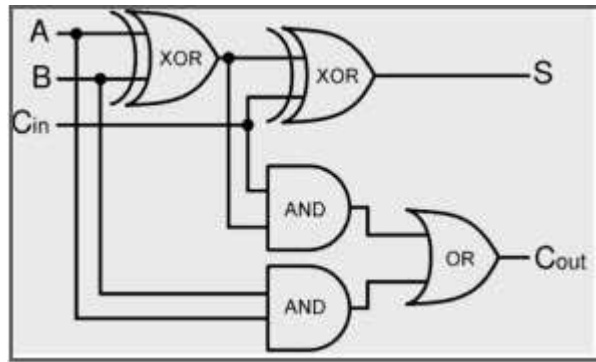
Level 1: Construct and verify the HA/FA logic circuits by using logical gates.

CIRCUIT:

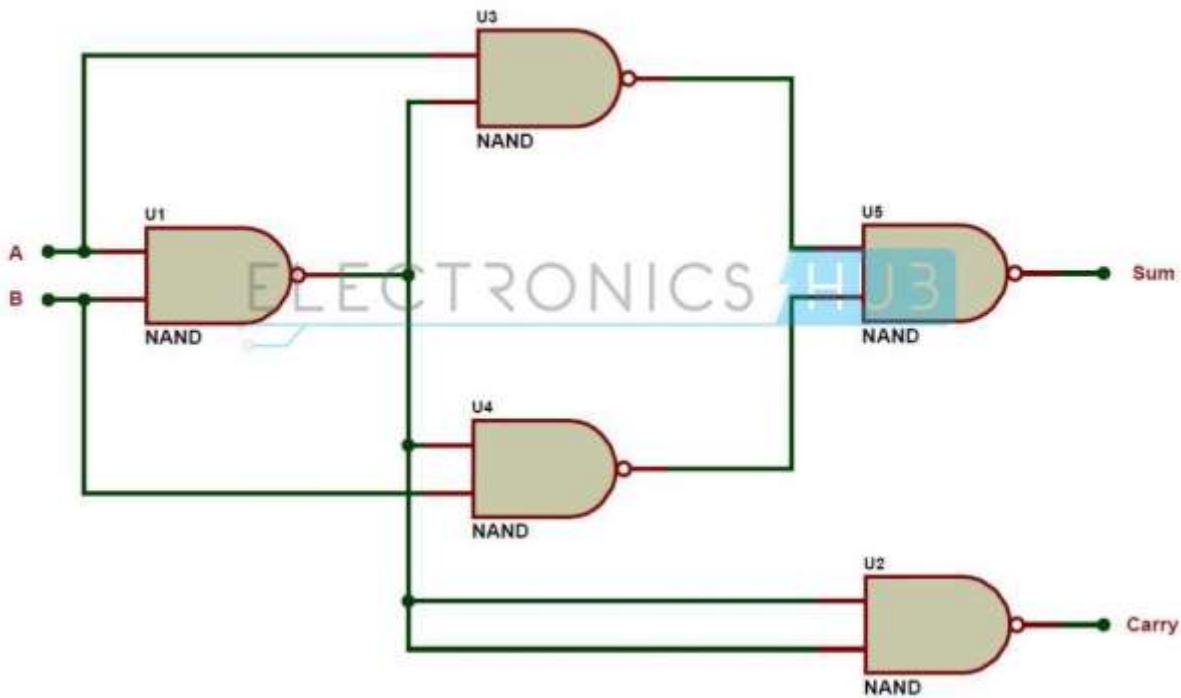
HALF ADDER:



FULL ADDER:



Level 2: Construct and verify the HA logic circuits by using Universal logic gates.



PROCEDURE FOR CONDUCTION OF EXPERIMENT:

1. Design the logic diagrams as per the problem statement.
2. Check the ICs for proper working.
3. Connect the gates as per the pin numbers required.
4. Observe the output by changing the input as per the requirement.
5. Note down the results and verify the truth table.

RESULT:

Experiment No. 4

Aim:

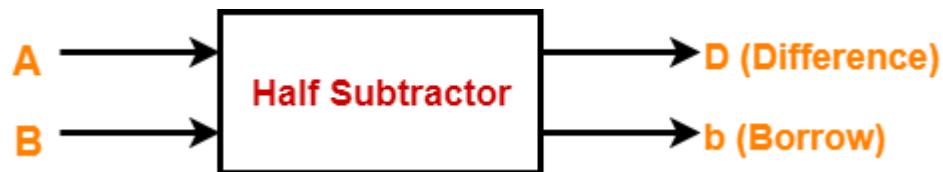
Level 1: Construct and verify the Half Subtractor /Full Subtractor logic circuits by using logical gates.

Level 2: Construct and verify the Half Subtractor logic circuits by using Universal logic gates.

THEORY:

- Half Subtractor is a combinational logic circuit.
- It is used for the purpose of subtracting two single bit numbers.
- It contains 2 inputs and 2 outputs (difference and borrow).

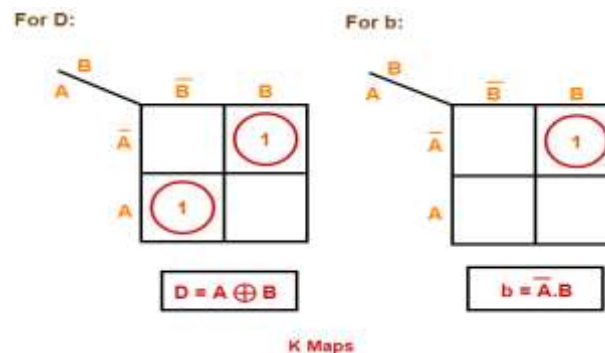
BLOCK DIAGRAM:



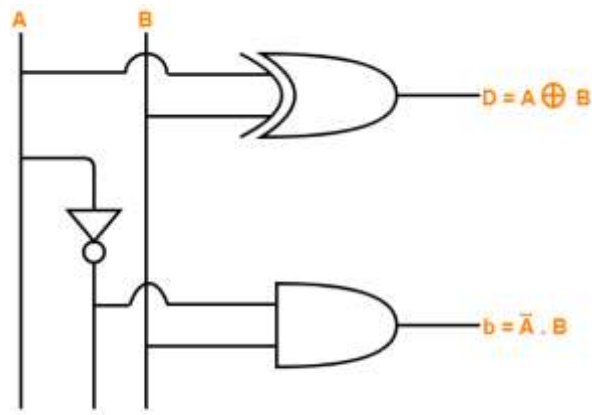
TRUTH TABLE:

Inputs		Outputs	
A	B	D (Difference)	b (Borrow)
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

LOGICAL EXPRESSION:



LOGICAL CIRCUIT:

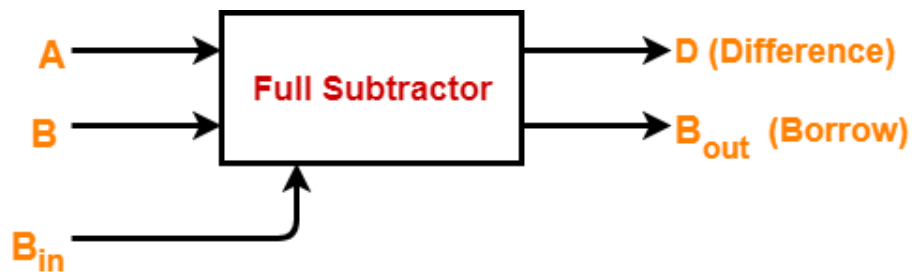


Half Subtractor Logic Diagram

Full Subtractor-:

- Full Subtractor is a combinational logic circuit.
- It is used for the purpose of subtracting two single bit numbers.
- It also takes into consideration borrow of the lower significant stage.
- Thus, full subtractor has the ability to perform the subtraction of three bits.
- Full subtractor contains 3 inputs and 2 outputs (Difference and Borrow) as shown-

BLOCK DIAGRAM:

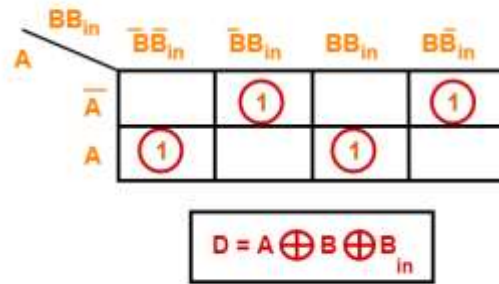


TRUTH TABLE:

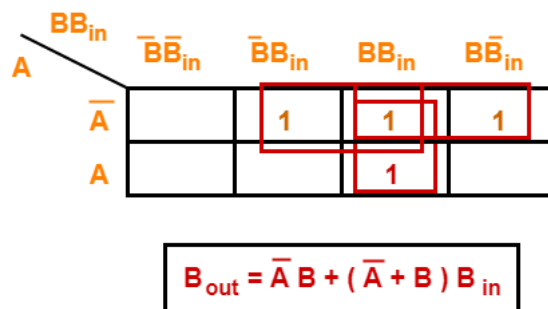
Inputs			Outputs	
A	B	B _{in}	B _{out} (Borrow)	D (Difference)
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

LOGICAL EXPRESSION:

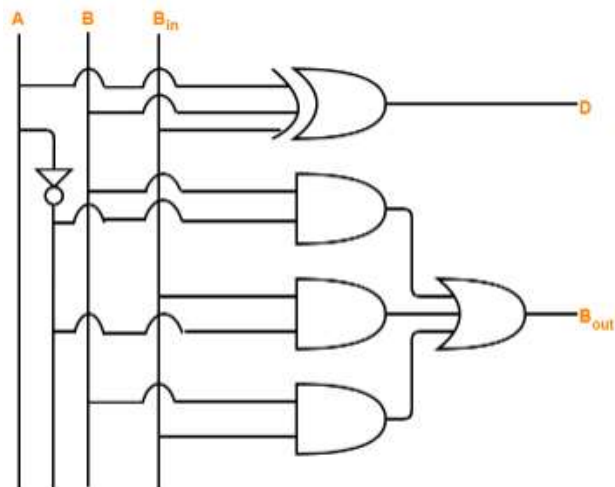
For D:



For B_{in} :



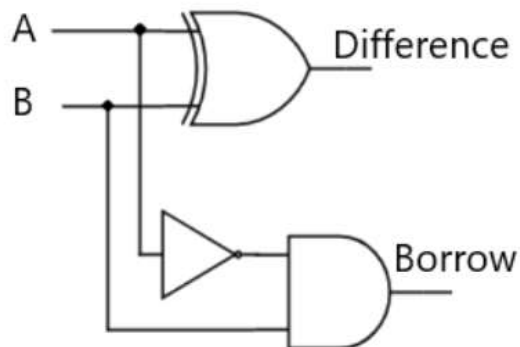
LOGICAL CIRCUIT OF FULL SUBTRACTOR:



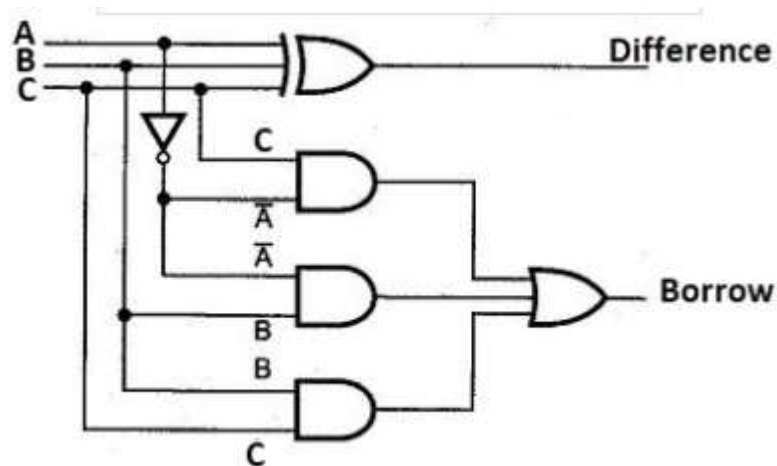
CONDUCTION OF EXPERIMENT:

Level 1: Construct and verify the Half Subtractor /Full Subtractor logic circuits by using logical gates.

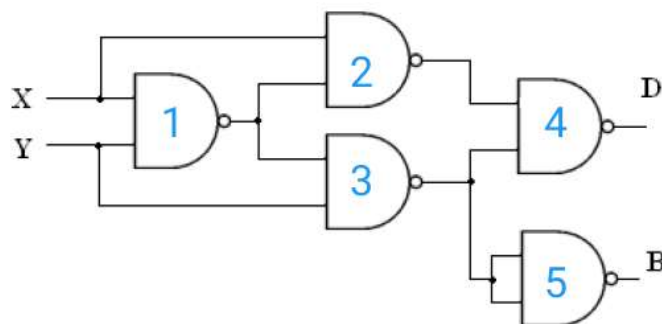
HALF SUBTRACTOR:



FULL SUBTRACTOR:



Level 2: Construct and verify the Half Subtractor logic circuits by using Universal logic gates.



Procedure

1. Place the IC on IC Trainer Kit.
2. Connect VCC and ground to respective pins of IC Trainer Kit.
3. Implement the circuit as shown in the circuit diagram.
4. Connect the inputs to the input switches provided in the IC Trainer Kit.
5. Connect the outputs to the switches of O/P LEDs
6. Apply various combinations of inputs according to the truth table and observe the condition of LEDs.
7. Note down the corresponding output readings for various combinations of inputs.
8. Power Off Trainer Kit, disconnect all the wire connections and remove IC's from IC-Base.

RESULT:

Experiment No. 5:

AIM: Construct and verify the combinational logic circuit for given specifications.

Level 1: Specifications given in the form of Truth table. Implement using basic gates.

Level 2: Specification should be extracted from the given scenario. Implement using universal gates only.

Design of Combinational Logic Circuit based on the below mentioned Scenario

An Engine has 4 failsafe sensors. The Engine should keep running unless any of the following conditions arise:

- If sensor 1 is activated.
- If sensor 2 and sensor 3 are activated at the sametime.
- If sensor 4 and sensor 3 are activated at the sametime.
- If sensors 2, 3, 4 are activated at the same time.

Design and implement the simplified logic using

- Basic gates only
- NAND Gates only (2 or 3 Input Gate)

SOLUTION:

(i) TRUTH TABLE :

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0

1	1	1	1	0
---	---	---	---	---

(ii) SIMPLIFICATION USING K-MAP :

CD \ AB	00	01	11	10
00	1 (0)	1 (1)	0 (3)	1 (2)
01	1 (4)	1 (5)	0 (7)	0 (6)
11	0 (12)	0 (13)	0 (15)	0 (14)
10	0 (8)	0 (9)	0 (11)	0 (10)

$\leftarrow A'B'D'$
 $A'C'$

$Y = A'B'D' + A'C'$

(iii) DESIGN OF LOGIC DIAGRAM USING NAND GATES :

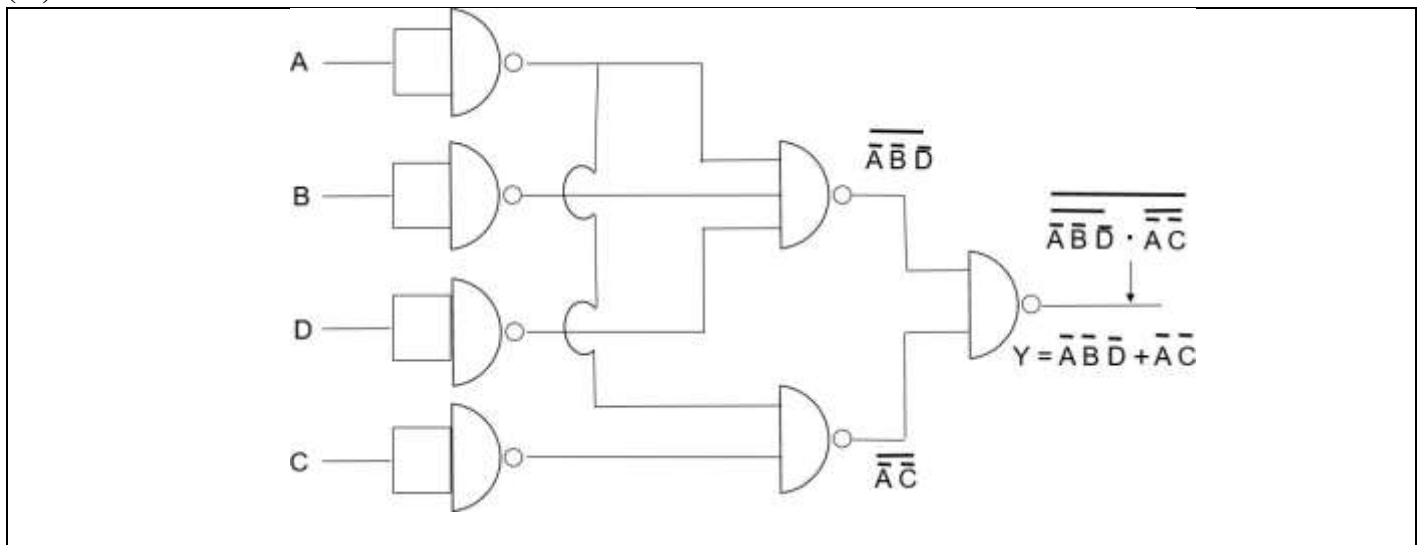


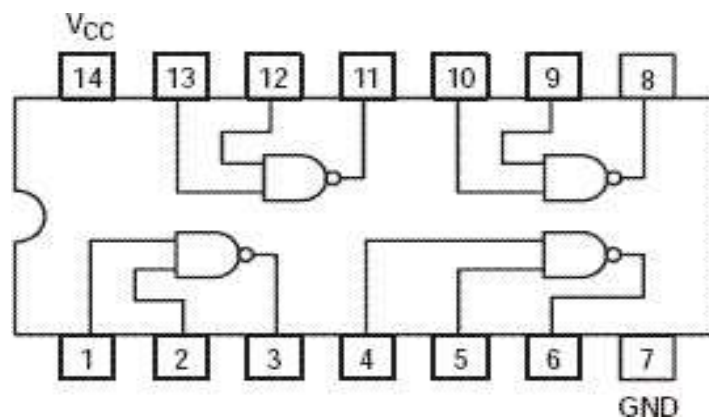
Fig. 2.1: Design of logic diagram using NAND gates

(iv) PROCEDURE FOR CONDUCTION OF EXPERIMENT:

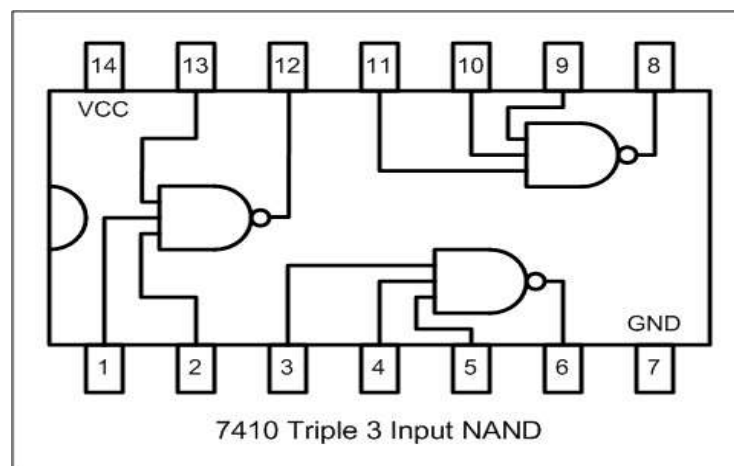
1. Plot the truth table for the problem statement.
2. Obtain the Simplified boolean expression using K-Map.
3. Design the logic diagrams as per the problem statement.
4. Check the ICs for proper working.
5. Connect the gates as per the pin numbers required.
6. Observe the output by changing the input as per the requirement.
7. Note down the results and verify the truth table.

(v) PIN DIAGRAM

(a) 7400 2-INPUT NAND



(b) 7410 3-INPUT NAND



II) Aim: Design of Combinational Logic Circuit based on the below mentioned Scenario

A digital system is to be designed in which the month of the year is given as input in four bit form. The month January is represented as '0000', February '0001' and so on. The output of the system should be '1' corresponding to the input of the month containing 31 days or otherwise it is '0'. Consider the excess numbers in the input beyond '1011' as don't care conditions for system of four variables (A, B, C, D).

Design and implement the simplified logic using

- i. Basic gates only
- ii. NAND Gates only (2 or 3 Input Gate)

SOLUTION:

a) TRUTH TABLE :

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

b) SIMPLIFICATION USING K-MAP :

CD \ AB	00	01	11	10
00	1 (0)	0 (1)	0 (3)	1 (2)
01	1 (4)	0 (5)	1 (7)	1 (6)
11	X (12)	X (13)	X (15)	X (14)
10	0 (8)	1 (9)	1 (11)	0 (10)

$$Y = A'D' + BC + AD$$

c) DESIGN OF LOGIC DIAGRAM USING NAND GATES :

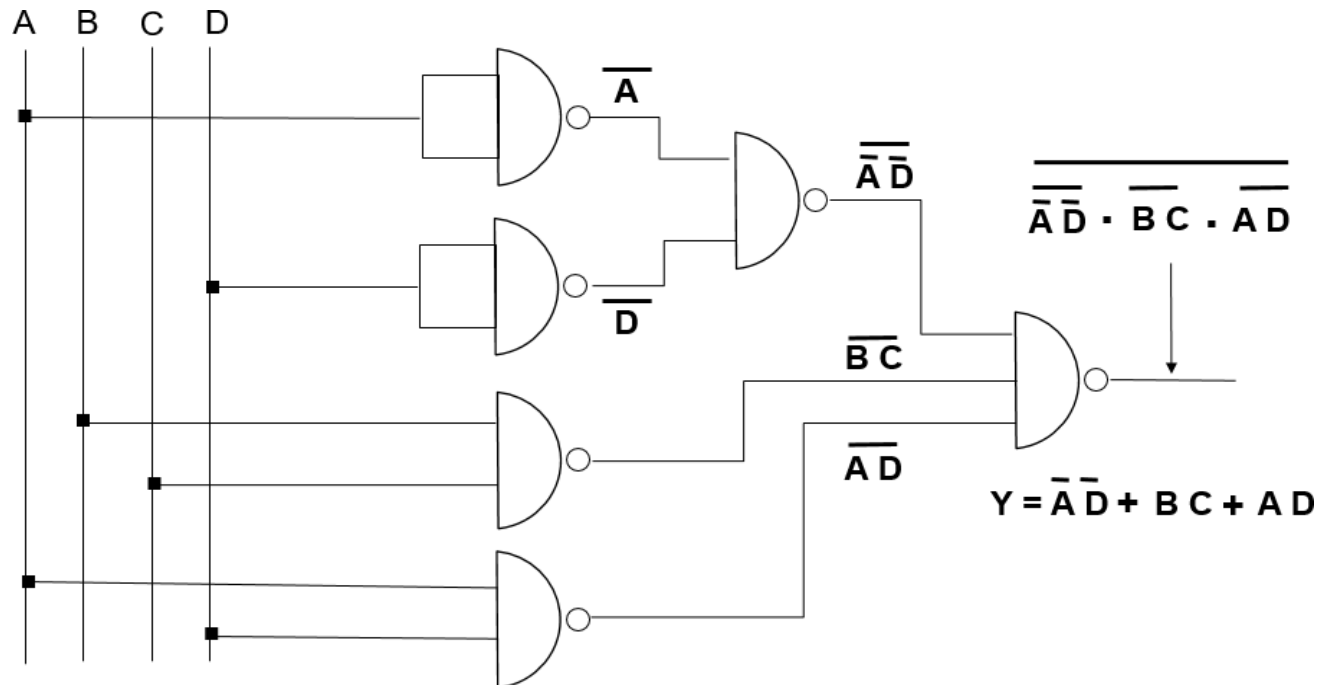


Fig.3.1: Design of logic diagram using NAND gates

d) PROCEDURE FOR CONDUCTION OF EXPERIMENT :

1. Plot the truth table for the problem statement.

2. Obtain the Simplified boolean expression using K-Map.
3. Design the logic diagrams as per the problem statement.
4. Check the ICs for proper working.
5. Connect the gates as per the pin numbers required.
6. Observe the output by changing the input as per the requirement.
7. Note down the results and verify the truth table.

RESULT

Experiment No. 6

AIM: Study of SR and D Flip flops

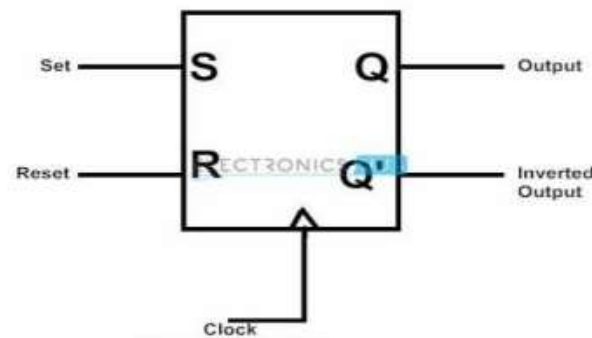
Level 1: Verify the operation of SR and D Flip-Flops USING CIRCUIT VERSE ONLINE SIMULATOR

Level 2: Construct and verify a SR Flip Flop using D Flip Flops.

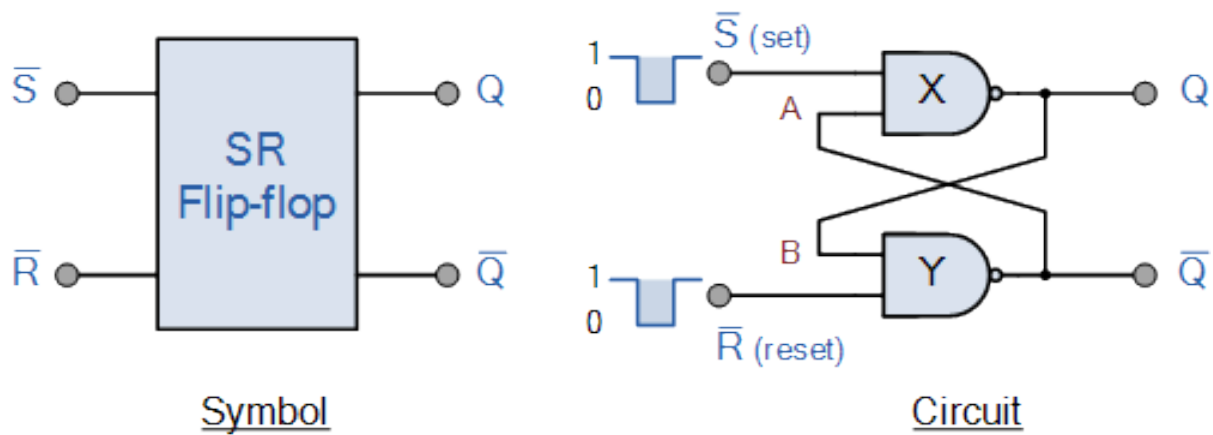
I) SR Flip flop

APPARATUS REQUIRED: - Logic trainer kit, NAND gates ICs- 7400, NOR gates ICs- 7402, wires.

THEORY: The SR flip-flop is one of the fundamental parts of the sequential circuit logic. SR flip-flop is a memory device and a binary data of 1 – bit can be stored in it. SR flip-flop has two stable states in which it can store data in the form of either binary zero or binary one. Like all flip-flops, an SR flip-flop is also an edge sensitive device. SR flip-flop is one of the most vital components in digital logic and it is also the most basic sequential circuit that is possible. The S and R in SR flip-flop means ‘SET’ and ‘RESET’ respectively. Hence it is also called Set–Reset flipflop. The symbolic representation of the SR Flip Flop is shown below.

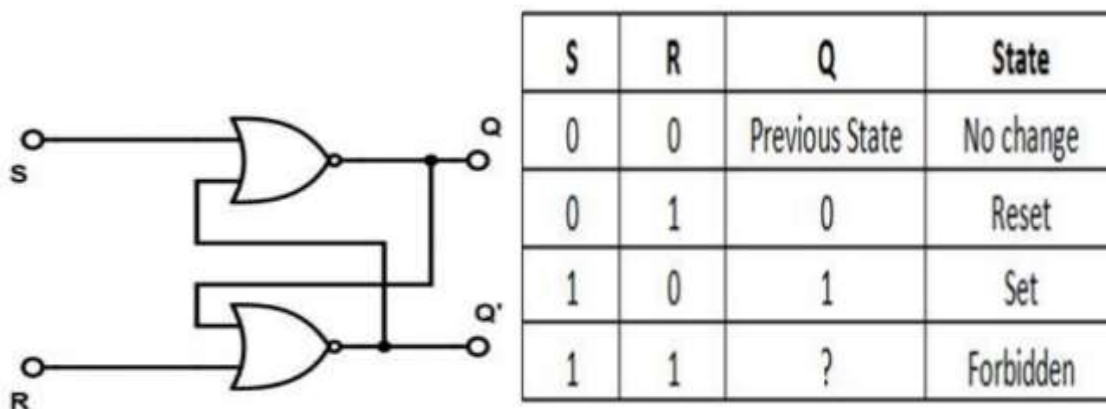


Working Principle: SR flip-flop works during the transition of clock pulse either from low to high or from high to low (depending on the design) i.e. it can be either positive edge triggered or negative edge triggered. For a positive edge triggered SR flip-flop, suppose, if S input is at high level (logic 1) and R input is at low level (logic 0) during a low to high transition on clock pulse, then the SR flip-flop is said to be in SET state and the output of the SR flip-flop is SET to 1. For the same clock situation, if the R input is at high level (logic 1) and S input is at low level (logic 0), then the SR flip-flop is said to be in RESET state and the output of the SR flip-flop is RESET to 0. The SR flip-flops can be designed by using logic gates like NOR gates and NAND gates. S-R Flip-Flop Using NAND Gate SR flip flop can be designed by cross coupling of two NAND gates. It is an active low input SR flip-flop. The circuit of SR flip-flop using NAND gates is shown in below figure:



S	R	Q	STATE
0	0	PREVIOUS STATE	NO CHANGE
0	1	0	RESET
1	0	1	SET
1	1	?	FORBIDDEN

S-R flip-flop using NOR gate



PROCEDURE:

- Connections are given as per circuit diagram.
- Logical inputs are given as per circuit diagram.
- Observe the output and verify the truth table.

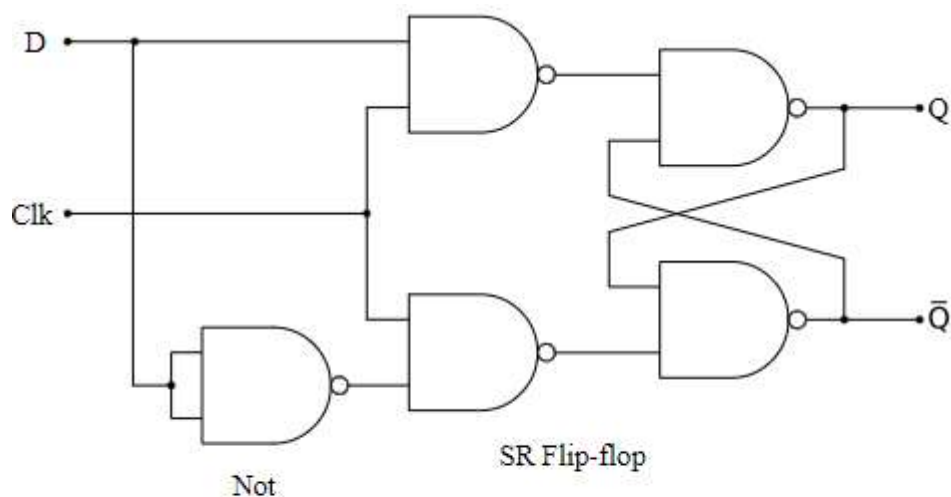
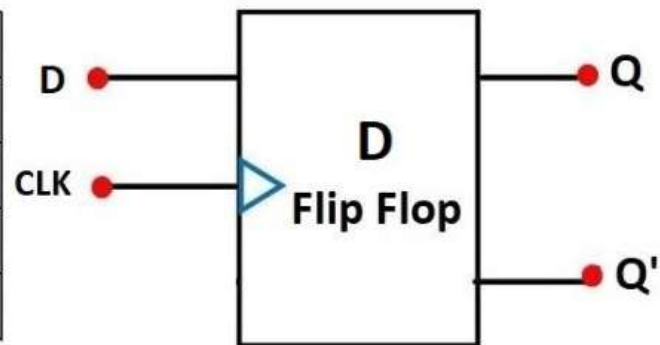
RESULT: Design of S-R Flip flop using NAND & NOR gates was verified successfully.

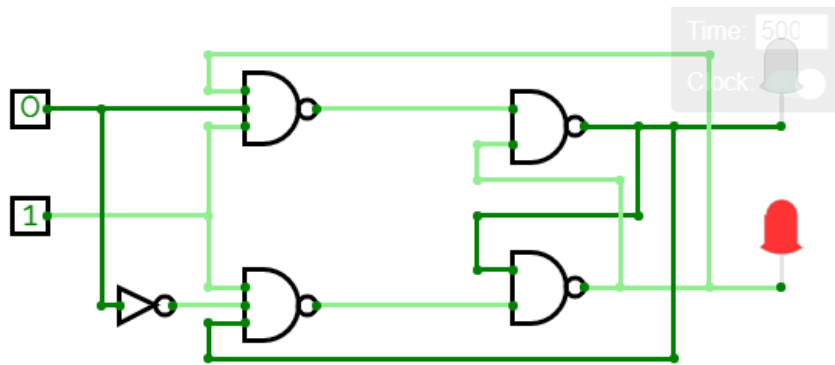
D-FF:

THEORY: D flip flop also called as delay flip flop where it can be used to introduce a delay in the digital circuit by changing the propagation delay of the flip flop. Here the input data bit at D will reflect at the output after a certain propagation delay

Truth Table of DFlip Flop

D	CLK	Q	Q'
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0





D Flip-Flop using NAND Gate

Level 2: Construct and verify a SR Flip Flop using D Flip Flop.

FLIP FLOP CONVERSIONS

PROCEDURE FOR CONVERSION

1. Draw the block diagram of the target flip flop from the given problem.
2. Write truth table for the target flip-flop.
3. Write excitation table for the available flipflop.
4. Draw k-map for target flip-flop.
5. Draw the block diagram.

TARGET FLIP FLOP: SRFF

AVAILABLE FLIPFLOP: D Flip Flop.

TRUTH TABLE FOR SRFF

CLK	S	R	Q	Q'	CONDITION
0	X	X	Q	Q'	PREVIOUS STATE
1	0	0	Q	Q'	PREVIOUS STATE
1	0	1	0	1	RESET
1	1	0	1	0	SET
1	1	1	Z	Z'	UNDEFINED STATE

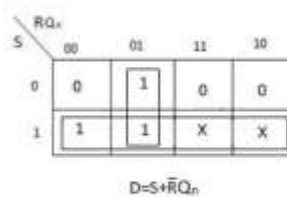
EXCITATION TABLE FOR DFF

QN	QN+1	D
0	0	0
0	1	1
1	0	0
1	1	1

$$D = Q_{N+1}$$

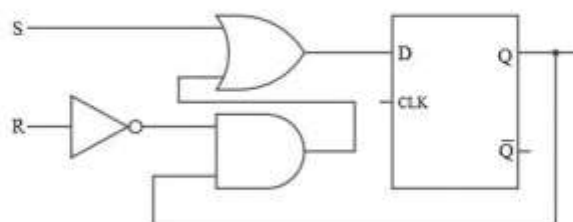
CHARACTERISTIC TABLE FOR TARGET (SRFF)					EXCITATION TABLE OF D
	S	R	QN	QN+1	$D=Q_{N+1}$
m0	0	0	0	0	0
m1	0	0	1	1	1
m2	0	1	0	0	0
m3	0	1	1	0	0
m4	1	0	0	1	1
m5	1	0	1	1	1
m6	1	1	0	X	X
m7	1	1	1	X	X

The K-Map for the required input-output relation is;



K-Map for D - SR Flip Flop using D Flip Flop

Its logic diagram can be given as;



RESULT:

Experiment No. 7

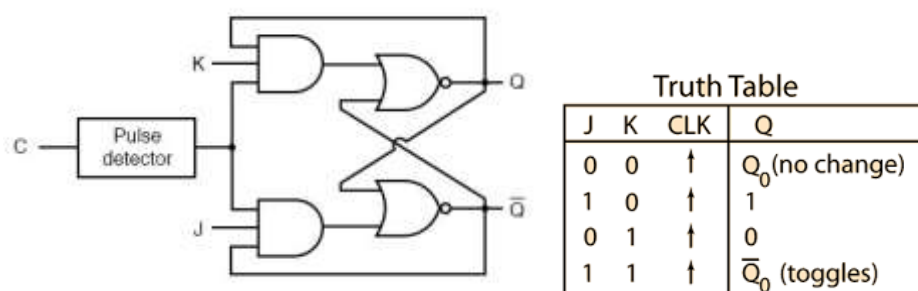
AIM: Study of JK Flip-flop and Toggle Flip-Flop.

Level 1: Verify the operation of JK Flip-flop and Toggle Flip-Flop USING ONLINE SIMULATOR










Level 2: To design and verify the circuit in which 2 JK FF are connected together in series and output of first FF is connected to input of second FF..

APPARATUS REQUIRED: - Logic trainer kit, Flip-flop ICs- 7476, wires.

THEORY: The JK flip-flop is the modified version of SR flip-flop with no invalid state; i.e. the state $J=K=1$ is not forbidden. It works such that J serves as set input and K serves as reset. The only difference is that for the combination $J=K=1$ this flip-flop; now performs an action: it inverts its state.



The flip-flop is constructed in such a way that the output Q is ANDed with K and CP. This arrangement is made so that the flip-flop is cleared during a clock pulse only if Q was previously 1. Similarly, Q' is ANDed with J and CP, so that the flip-flop is cleared during a clock pulse only if Q' was previously 1.

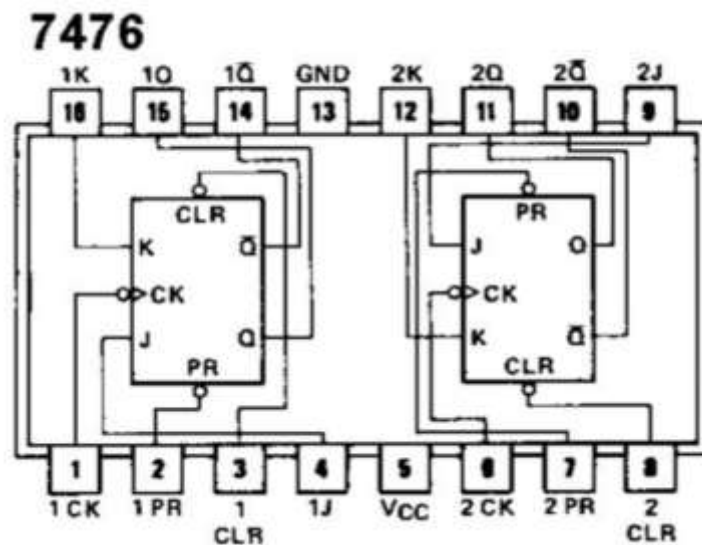
Trigger	Inputs		Output				Inference
			Present State		Next State		
CLK	J	K	Q	Q'	Q	Q'	
	x	x	-		-		Latched
	0	0	0	1	0	1	No Change
			1	0	1	0	
	0	1	0	1	0	1	Reset
			1	0	0	1	
	1	0	0	1	1	0	Set
			1	0	1	0	
	1	1	0	1	1	0	Toggles
			1	0	0	1	

When J = K = 0

When both J and K are 0, the clock pulse has no effect on the output and the output of the flip-flop is the same as its previous value. This is because when both the J and K are 0, the output of their respective AND gate becomes 0.

When J=0, K=1

When $J=0$, the output of the AND gate corresponding to J becomes 0 (i.e.) $S=0$ and $R=1$. Therefore, Q' becomes 0. This condition will reset the flip-flop. This represents the RESET state of Flip-flop.



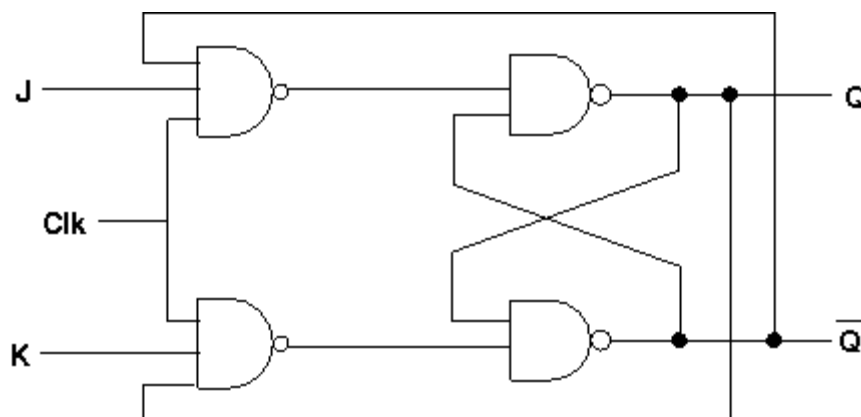
When $J=1, K=0$

In this case, the AND gate corresponding to K becomes 0 (i.e.) $S=1$ and $R=0$. Therefore, Q becomes 1. This condition will set the Flip-flop. This represents the SET state of Flip-flop.

When $J=K=1$

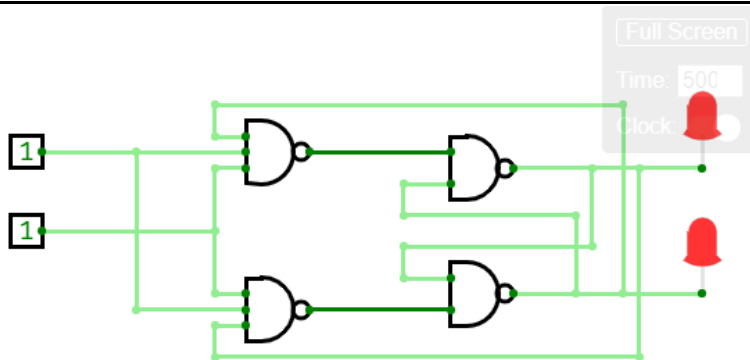
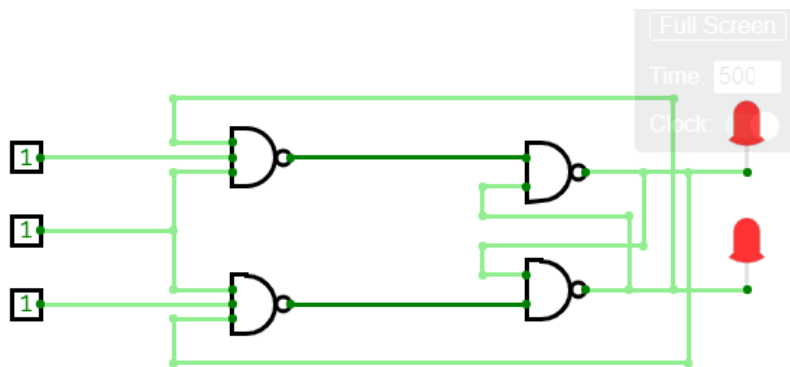
Consider the condition of $CP=1$ and $J=K=1$. This will cause the output to complement again and again. This complement operation continues until the Clock pulse goes back to 0. Since this condition is undesirable, we have to find a way to eliminate this condition and it is called as Race-around Condition. This undesirable behaviour can be eliminated by Edge triggering of JK flipflop or by using master slave JK Flip-flops.

Realization using Universal Gates and verify the truth table



SIMULATION USING CIRCUIT VERSE

JKFF USING NAND GATE



T Flip-Flop using NAND Gate

ii. Master Slave Configuration: To design and verify the circuit in which 2 JK FF are connected together in series and output of first FF is connected to input of second FF.

Experiment No. 8

AIM: Construct and verify the sequential logic circuit for given specifications

Level 1: Specifications given in the form of Truth table.

Level 2: Specification should be extracted from the given scenario.(USING ONLINE SIMULATOR ONLY)

LEVEL-1:Design 3 bit Asynchronous UP/down counter

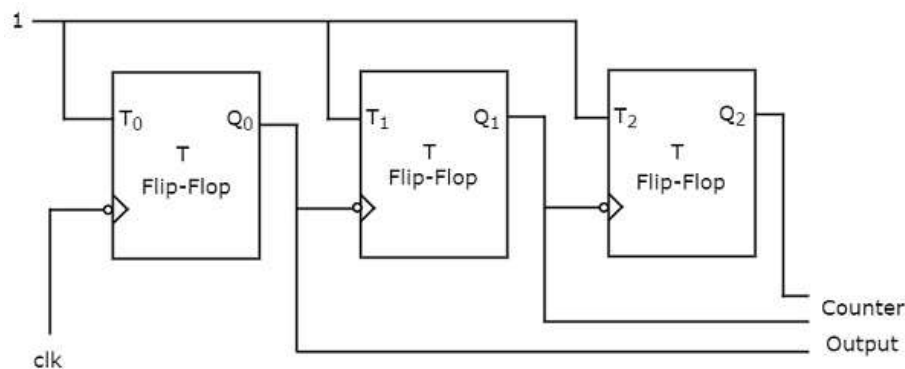
THEORY

Asynchronous counter. The output of system clock is applied as clock signal only to first flip-flop. The remaining flip-flops receive the clock signal from output of its previous stage flip-flop. Hence, the outputs of all flip-flops do not change affect at the same time.

- Asynchronous Binary up counter
- Asynchronous Binary down counter

Asynchronous Binary Up Counter

An 'N' bit Asynchronous binary up counter consists of 'N' T flip-flops. It counts from 0 to $2^N - 1$. The **block diagram** of 3-bit Asynchronous binary up counter is shown in the following figure.

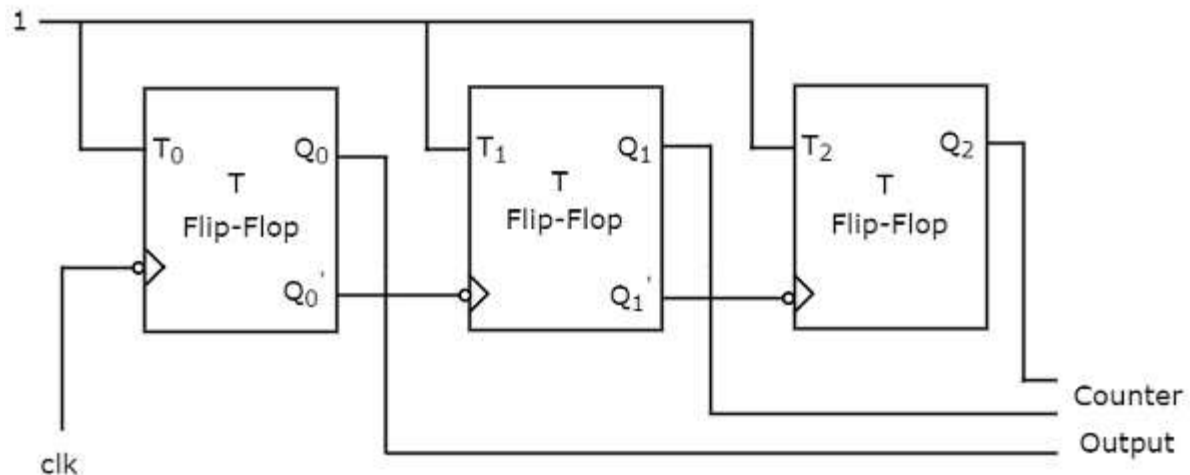


Truth table OF UP COUNTER:

CLK	Q2	Q1	Q0
0	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

Asynchronous Binary Down Counter

An 'N' bit Asynchronous binary down counter consists of 'N' T flip-flops. It counts from $2^N - 1$ to 0. The **block diagram** of 3-bit Asynchronous binary down counter is shown in the following figure.



The block diagram of 3-bit Asynchronous binary down counter is similar to the block diagram of 3-bit Asynchronous binary up counter. But, the only difference is that instead of connecting the normal outputs of one stage flip-flop as clock signal for next stage flip-flop, connect the **complemented outputs** of one stage flip-flop as clock signal for next stage flip-flop. Complemented output goes from 1 to 0 is same as the normal output goes from 0 to 1.

TRUTH TABLE OF DOWN COUNTER:

CLK	Q2	Q1	Q0
0	1	1	1
1	1	1	0
1	1	0	1
1	1	0	0
1	0	1	1
1	0	1	0
1	0	0	1
1	0	0	0

LEVEL-2 DESIGN 3BIT SYNCHRONOUS UP COUNTER USING JKFF (IC-7476)

IT REQUIRES NO OF FF –3

3BIT---→ (000-111)

EXCITATION TABLE:

	PRESENT STATE(INPUT)			NEXT STATE			FF1		FF2		FF3	
	Q2	Q1	Q0	Q2+	Q1+	Q0+	J2	K2	J1	K1	J0	K0
m0	0	0	0	0	0	1	0	X	0	X	1	X
m1	0	0	1	0	1	0	0	X	1	X	X	1
m2	0	1	0	0	1	1	0	X	X	0	1	X
m3	0	1	1	1	0	0	1	X	X	1	X	1
m4	1	0	0	1	0	1	X	0	0	X	1	X
m5	1	0	1	1	1	0	X	0	1	X	X	1
m6	1	1	0	1	1	1	X	0	X	0	1	X
m7	1	1	1	0	0	0	X	1	X	1	X	1

K-MAP FOR J2

Q2/Q1Q0	00	01	11	10
0			1	
1	x	x	x	x

$J2 = Q1 Q0$

K-MAP FOR K2

Q2/Q1Q0	00	01	11	10
0	X	X	X	X
1			1	

$K2 = Q1 Q0$

K-MAP FOR J1

Q2/Q1Q0	00	01	11	10
0		1	X	X
1		1	X	X

$J1=Q0$

K-MAP FOR K1

Q2/Q1Q0	00	01	11	10
0	X	X	1	

1	X	X	1	
---	---	---	---	--

$K1=Q0$

KMAP FOR J0

Q2/Q1Q0	00	01	11	10
0	X	1	X	1
1	1	X	1	X

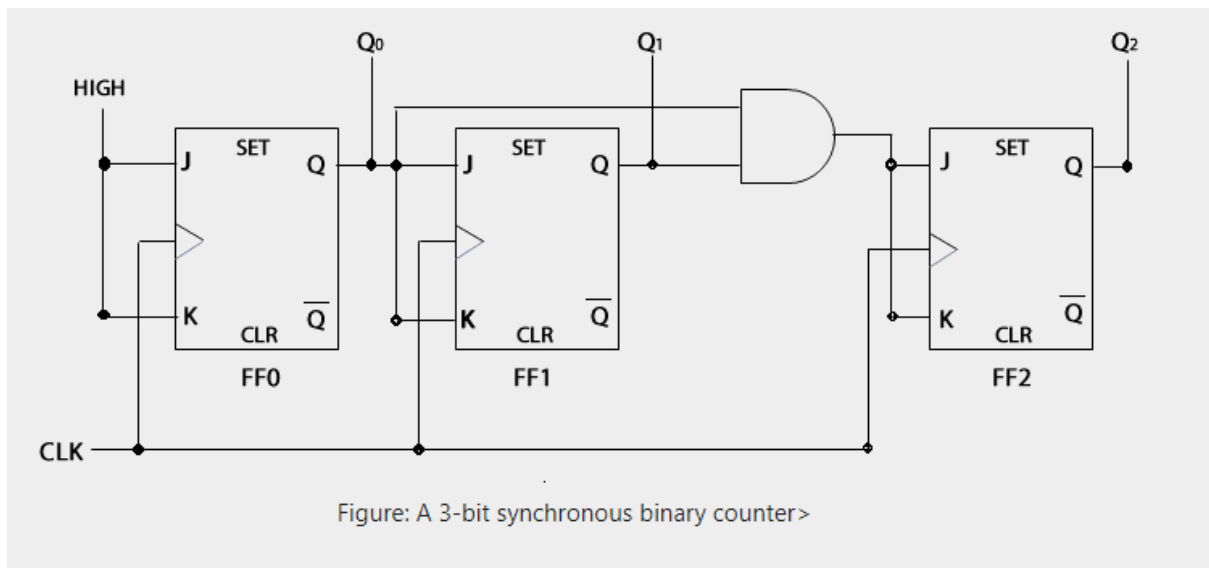
$J0=K0=1$

OUTPUT EXPRESSION

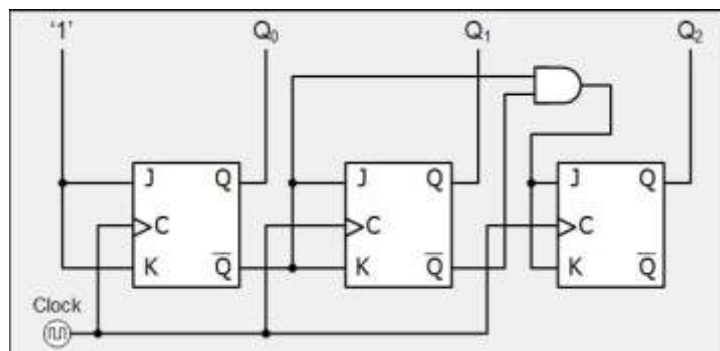
$J3 = K3 = Q2 \cdot Q1 \cdot Q0$ (4 BIT –SYNCHRONOUS COUNTER)

- $J2=K2=Q1Q0$
- $J1=K1=Q0$
- $J0=K0=1$

LOGICAL DIGRAM OF 3BIT SYNCHRONOUS COUNTER USING JKFF



LOGICAL DIGRAM OF 3BIT SYNCHRONOUS COUNTER USING JKFF



Experiment No. 9: Write the HDL coding for basic combinational logic circuits

Level 1: Dataflow Modeling

Level 2: Behavioral Modeling

Level 1: Write verilog code for basic/universal gates and their test bench for verification .
Observe the waveform.

THEORY :

- **AND GATE:** The output of an AND gate is equal to 1 only if both inputs are equal to 1. An AND gate can have many inputs. In any case, no matter how many inputs it has, the output is only equal to 1 if all the inputs are equal to 1, otherwise the output is 0.

- **OR GATE:** The output of an OR gate is equal to 1 if either of the input is equal to one. If neither of the inputs is equal to 1, the output is equal to zero. An OR gate can have as many inputs as we want. The output will be equal to 1 if any of the inputs is equal to 1.

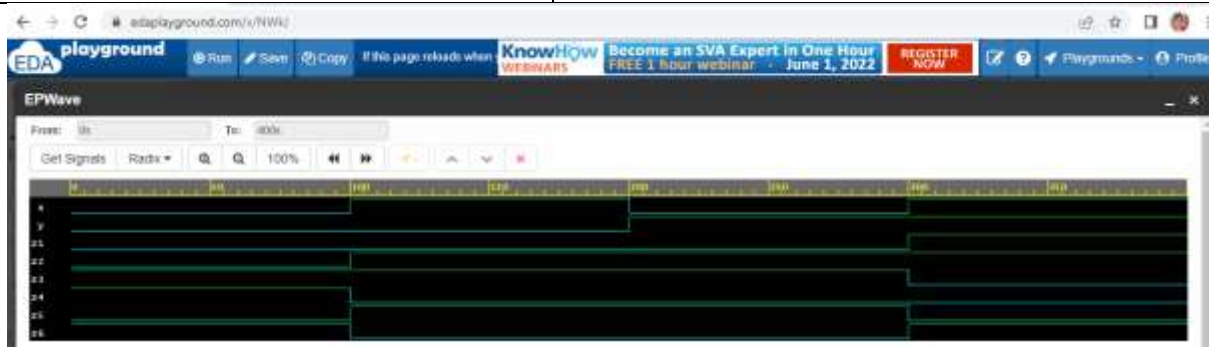
- **EX-OR GATE:** The output of an XOR gate is equal to 1 if any one of the input is equal to one and equal to zero if both inputs are equal to zero or if both inputs are equal to 1. This is the difference between an OR gate and an XOR gate, an OR gates output will equal 1 if both inputs are equal to 1.

- **EX-NOR GATE:** whose function is the inverse of the exclusive or (xor) gate .The output is one when both the inputs are same .the output is zero when the inputs are complementary to each other.

- **NAND GATE:**NAND gate behaves the same as an AND gate with a not (inverter) gate connected to the output terminal. to symbolize this output signal inversion, the nand gate symbol has a bubble on the output line. the output will be "low" (0) if and only if all inputs are "high" (1). if any input is "low" (0), the output will go "high" (1).

- **NOR GATE:**NOR gate behaves the same as an or gate with a not(inverter) gate connected to the output terminal. The output will be low(1) if and only if all inputs are low(0) and if any input is high(1),the output will go low(0).

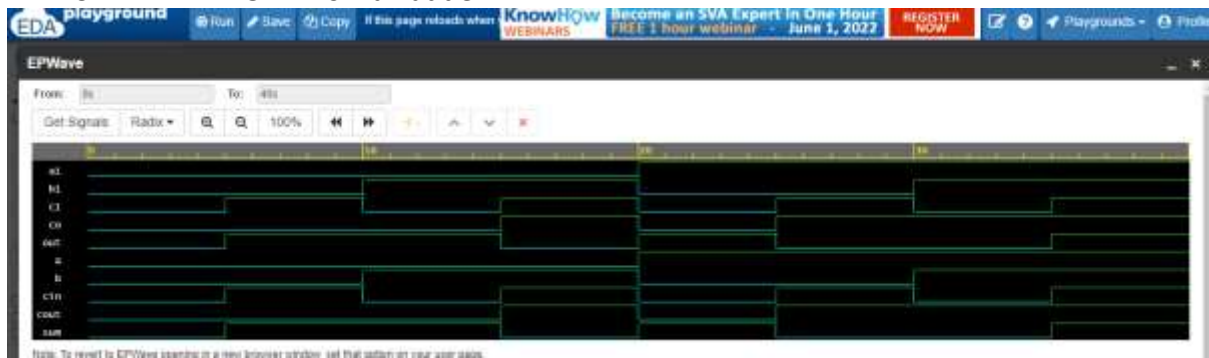
VERILOG CODE	TESTBENCH CODE
<pre>module logic_gate(x,y,z1,z2,z3,z4,z5,z6); input x,y; output z1,z2,z3,z4,z5,z6; assign z1=x&y; // and gate assign z2=x y; // or gate assign z3=~(x&y); //nand gate assign z4=~(x y); //nor gate assign z5=x^y; // exor gate assign z6=~(x^y); //exnor gate endmodule</pre>	<pre>module logicgates1; reg x,y; wire z1,z2,z3,z4,z5,z6; logic_gate uut(.x(x),.y(y),.z1(z1),.z2(z2),.z3(z3),.z4(z4),.z5(z5),.z6(z6)); initial begin x=0;y=0;#100; x=1;y=0;#100; x=0;y=1;#100; x=1;y=1;#100; end initial begin \$monitor ("%t z1 = %d z2 = %d z3 = %d z4 = %d z5 = %d z6 = %d ", \$time, z1,z2,z3,z4,z5,z6); \$dumpfile("dump.vcd"); \$dumpvars(); end endmodule</pre>



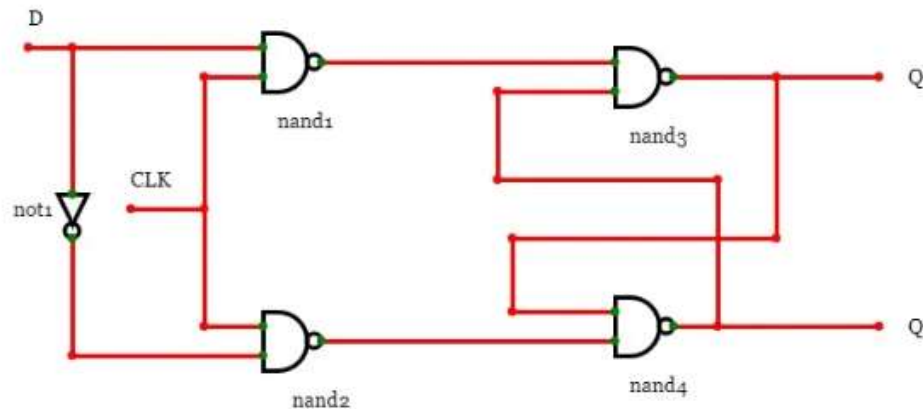
Level 2: Behavioral Modeling :write Verilog code for full adder using behavioral modelling

VERILOG CODE	TESTBENCH CODE
<p>.Data Flow Modeling:</p> <pre> module full_adder(input a,b,cin, output sum,cout); assign sum=(a^b^cin); assign cout=(a & b) (b & cin) (cin & a); endmodule </pre> <p>Behavioral Modeling</p> <pre> module full_adder(a, b, cin, sum, cout); input a , b, cin; output reg sum,cout; always@(*) begin {cout, sum} = a + b +cin; end endmodule </pre>	<pre> module top_tb; wire out; wire co; reg a1; reg b1; reg c1; full_adder dut1(.sum(out), .cout(co), .a(a1), .b(b1), .cin(c1)); initial begin a1=1'b0; b1=1'b0; c1=1'b0; #5 a1=1'b0; b1=1'b0; c1=1'b1; #5 a1=1'b0; b1=1'b1; c1=1'b0; #5 a1=1'b0; b1=1'b1; c1=1'b1; #5 a1=1'b1; b1=1'b0; c1=1'b0; #5 a1=1'b1; b1=1'b0; c1=1'b1; #5 a1=1'b1; b1=1'b1; c1=1'b0; #5 a1=1'b1; b1=1'b1; c1=1'b1; #5 a1=1'b1; b1=1'b1; c1=1'b1; end initial begin \$monitor (" %d", \$time,out); \$dumpfile("dump.vcd"); \$dumpvars(); end endmodule </pre>

EXPECTED WAVEFORM for full adder:



Aim : Write verilog code for DFF using Gate level Modeling .



The logic circuit of D Flip Flop

VERILOG CODE	TESTBENCH CODE
<pre> module d_ff_gate(q,qbar,d,clk); input d,clk; output q, qbar; wire x,dbar,y; nand nand1(x,clk,d); not not1(dbar,d); nand nand2(y,clk,dbar); nand nand3(q,qbar,y); nand nand4(qbar,q,x); endmodule </pre>	<pre> module dff_tb; wire o1,o2; reg d1,ck1; reg r; d_ff_gate dut1(.q(o1), .qbar(o2), .d(d1), .clk(ck1)); initial begin d1=1'b0; ck1=1'b0; #500 \$finish; end always #40 d1=~d1; always #5 ck1=~ck1; initial begin \$monitor (" %d", \$time,o1); \$dumpfile("dump.vcd"); \$dumpvars(); end endmodule </pre>



Write the HDL coding for JKFF using Behavioral Modeling

VERILOG CODE	TESTBENCH CODE
<pre> module jkffcode(clk,reset,jk, q); input clk,reset; input [1:0] jk; output q; reg q; always@(posedge clk) begin if(reset) q=0; case(jk) 2'b00:q=q; 2'b01:q=0; 2'b10:q=1; 2'b11:q=~q; endcase end endmodule </pre>	<pre> module jkff_test; reg[1:0] i1; reg CLK1; wire o1;; jk_ff dut1(.q(o1),.jk(i1), .clk(CLK1)); initial begin i1=1'b0; CLK1=1'b0; i2=1'b0; #500 \$finish; end always #40 i1=~i1; always #5 CLK1=~CLK1; always #160 i2=~i2; initial begin \$monitor (" %d", \$time,o1); \$dumpfile("dump.vcd"); \$dumpvars(); End endmodule </pre>

Expected waveform:

