Lab Code:20ECL402
Digital Logic Design
Lab Manual

Department of Electronics & Communication Engineering

Bapatla Engineering College :: Bapatla

(Autonomous)
G.B.C. Road, Mahatmajipuram, Bapatla-522102, Guntur (Dist.)
Andhra Pradesh, India.
E-Mail:bec.principal@becbapatla.ac.in
Web:www.becbapatla.ac.in

Contents

Bapatla Engineering College :: Bapatla

(Autonomous)

Vision

- To build centers of excellence, impart high quality education and instill high standards of ethics and professionalism through strategic efforts of our dedicated staff, which allows the college to effectively adapt to the ever changing aspects of education.

- To empower the faculty and students with the knowledge, skills and innovative thinking to facilitate discovery in numerous existing and yet to be discovered fields of engineering, technology and interdisciplinary endeavors.

Mission

- Our Mission is to impart the quality education at par with global standards to the students from all over India and in particular those from the local and rural areas.

- We continuously try to maintain high standards so as to make them technologically competent and ethically strong individuals who shall be able to improve the quality of life and economy of our country.

Bapatla Engineering College :: Bapatla

(Autonomous)

Department of Electronics and Communication Engineering

Vision

To produce globally competitive and socially responsible Electronics and Communication Engineering graduates to cater the ever changing needs of the society.

Mission

- To provide quality education in the domain of Electronics and Communication Engineering with advanced pedagogical methods.

- To provide self learning capabilities to enhance employability and entrepreneurial skills and to inculcate human values and ethics to make learners sensitive towards societal issues.

- To excel in the research and development activities related to Electronics and Communication Engineering.

Bapatla Engineering College :: Bapatla

(Autonomous)

Department of Electronics and Communication Engineering

Program Educational Objectives (PEO's)

PEO-I: Equip Graduates with a robust foundation in mathematics, science and Engineering Principles, enabling them to excel in research and higher education in Electronics and Communication Engineering and related fields.

PEO-II: Impart analytic and thinking skills in students to develop initiatives and innovative ideas for Start-ups, Industry and societal requirements.

PEO-III: Instill interpersonal skills, teamwork ability, communication skills, leadership, and a sense of social, ethical, and legal duties in order to promote lifelong learning and Professional growth of the students.

Program Outcomes (PO's)

Engineering Graduates will be able to:

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, review 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 modeling 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 Teamwork: 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.

Bapatla Engineering College :: Bapatla

(Autonomous)

Department of Electronics and Communication Engineering

Program Specific Outcomes (PSO's)

PSO1: Develop and implement modern Electronic Technologies using analytical methods to meet current as well as future industrial and societal needs.

PSO2: Analyze and develop VLSI, IoT and Embedded Systems for desired specifications to solve real world complex problems.

PSO3: Apply machine learning and deep learning techniques in communication and signal processing.

Digital Logic Design Lab
II B.Tech–II Semester Code:20ECL402)

Lectures	0	Tutorial	0	Practical	3	Credits	1.5
Continuous Internal Assessment			30	Semester End Examination(3Hours)			70

Prerequisites: Fundamentals of Digital Electronics

Course Objectives: Students will be able

- To apply knowledge of Boolean algebra fundamentals to realize digital systems
- To design and conduct experiments related to various combinational logic circuits
- To implement various sequential logic circuits.
- To design and conduct experiments related to counters and registers.

Course Outcomes: At the end of the course, student will be able to

CO1	Realize different logic gates using discrete components and universal gates
CO2	Design and test various combinational logic circuits experimentally
CO3	Realize various operations using Digital ICs experimentally
CO4	Design and test various sequential logic circuits experimentally

Mapping of Course Outcomes with Program Outcomes & Program Specific Outcomes

	PO's												**PSO's**		
CO	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**1**	**2**	**3**
CO1	2	2		3					3				2	2	
CO2	2	2	3	3					3				2	2	
CO3	2	2		3					3				2	2	
CO4	2	2	3	3					3				2	2	
AVG	2	2	3	3					3				2	2	

LIST OF LAB EXPERIMENTS

1. Realization of Gates using Discrete Components.
2. Realization of Gates using Universal Building Block (NAND only).
3. Design of Combinational Logic Circuits like Half–adder, Full–adder, Half– Subtractor and Full-Subtractor.

4. Verification of 4-bit Magnitude Comparator.
5. Design of Encoders like 4:2 and 8:3 encoder.
6. Design of Decoders like BCD to Decimal decoder.
7. Design of Code Converters (Binary to Gray & Gray-Binary).
8. Design of Multiplexers/ De Multiplexers.
9. Verification of Truth Table of Flip-Flops using Gates.
10. Design of Shift register (To Verify Serial to parallel, parallel to Serial, Serial to Serial and parallel to parallel Converters) using Flip-Flops.
11. Design of Ring & Johnson Counters using Flip-Flops.
12. Conversion of Flip-Flops (JK-T, JK –D).
13. Design of Binary/Decade Counter.
14. Design of Asynchronous Counters
15. Design of Synchronous Counters

NOTE:

A minimum of 10(Ten) experiments have to be Performed and recorded by the candidate to attain eligibility for Semester End Examination.

1. Realization of Gates using Discrete Components.

Aim:

- To construct logic gates OR, AND, NOT, NOR, and NAND using discrete components.
- Verification of truth tables of Basic OR, AND, NOT, and NAND gates.

Apparatus required:

S.No	Equipment/Component Name	Specifications/Value	Quantity
1	Digital Trainer Board		1
2	Resistors	10KΩ, 1KΩ	Each one
3	Diodes	1N4002	3
4	Transistors (n-p-n)	BC107 (or) Q2N2222	1
5	Connecting Wires		

Theory:

OR Gate:

It consists of two diodes connected in parallel and these two diodes are followed by inputs. As shown in Fig.1 If A and B are two inputs of OR gate. Then the output will be A + B. Here the inputs given to the diodes will forward bias. The diodes i.e. when the inputs are given to the diode it will simply forward biased and it will conduct. There is no blocking or drop across the diode and the input will appear across the load resistor (10 KΩ) as an output. The standard inputs are either ‘0’ or ‘1’ (5V). If the inputs A and B are 1 and 0 respectively. Then diode D1 conducted i.e., it will simply acts as a switch in ON position and D2 is not conducted i.e., OFF position. The current will flows through D1 and we will get output as high i.e., ‘1’ like that possible inputs and corresponding outputs are shown in the truth table. From the truth table it is conducted that if any one inputs is high (logic ‘1’). Otherwise the output is low (logic ‘0’).

AND Gate:

It consists of two diodes followed by two inputs which reverse bias the diodes and 5V simply is connected to the junction of diodes through a 1 KΩ resistor as shown in Fig.2. The output is taken at the diode junction point 'J' with respect to ground. A and B are inputs when the two inputs are high (logic 1) are given to the circuit then both the diodes are reverse biased and acts as a switch in OFF state. Then there is no path for 5 V supply voltage to reach the ground. It is dropped across the diodes at which the outputs is taken and we will get 5V output i.e., logic '1' for the inputs A and B are 1 and 1 respectively output is '1'. If anyone input is low (logic 0) then the corresponding diode is forward biased by 5V supply and the circuit. If the input is low the output is low. If the both inputs are high then only the output is high.

NOT Gate:

It consists of a transistor, which is connected in common emitter configuration as shown in Fig.3. If transistor is operated in OFF, ON and saturation regions. In these regions the transistor is simply acts as a switch. If 'A' is the input to the NOT gate the output ' Ā'. In this circuit the output is taken at the collector. When the input is low (logic '0') the transistor is in cut off region i.e., across the collector resistor i.e., the output is high (logic '1') i.e., output is complement of the input. If the input is high then the transistor is entered into saturation region ON state, then there is no drop across the output resistor. So the output is low. If the input is high output is low. Therefore the output of a NOT gate is a complement of input shown in truth table.

NOR Gate:

NOR gate performs the NOR operation i.e. negation of OR operation. It means that Combination of OR gate and NOT gate is NOR gate.

NAND Gate:

NAND gate performs the NAND operation i.e. negation of AND operation. It means that Combination of AND gate and NOT gate is NAND gate.

Circuit Diagrams:

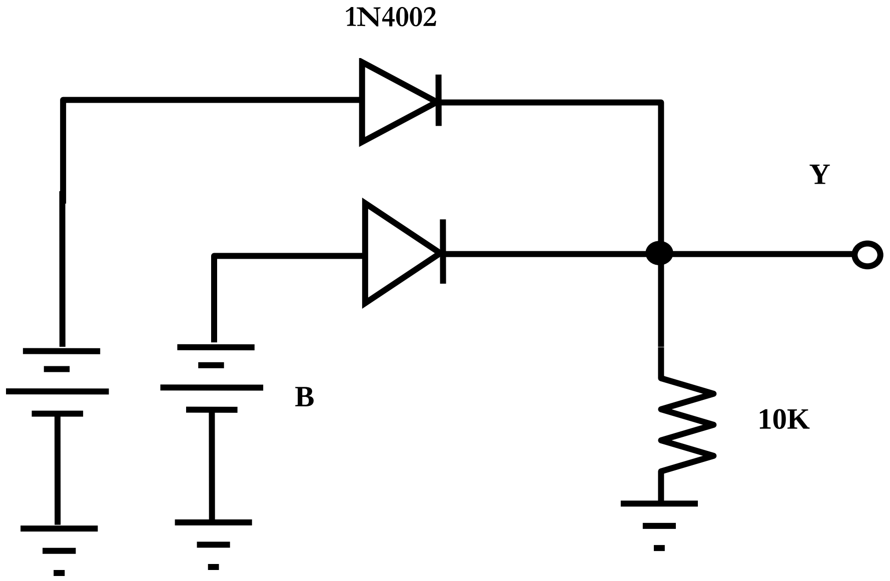


Fig 1: OR Gate

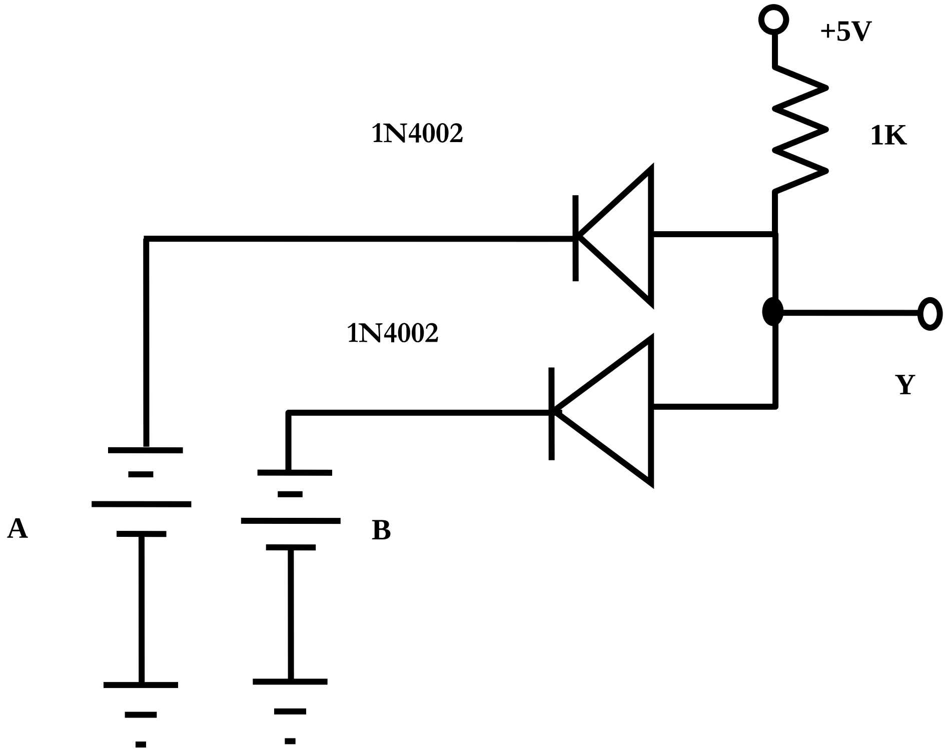


Fig 2: AND Gate

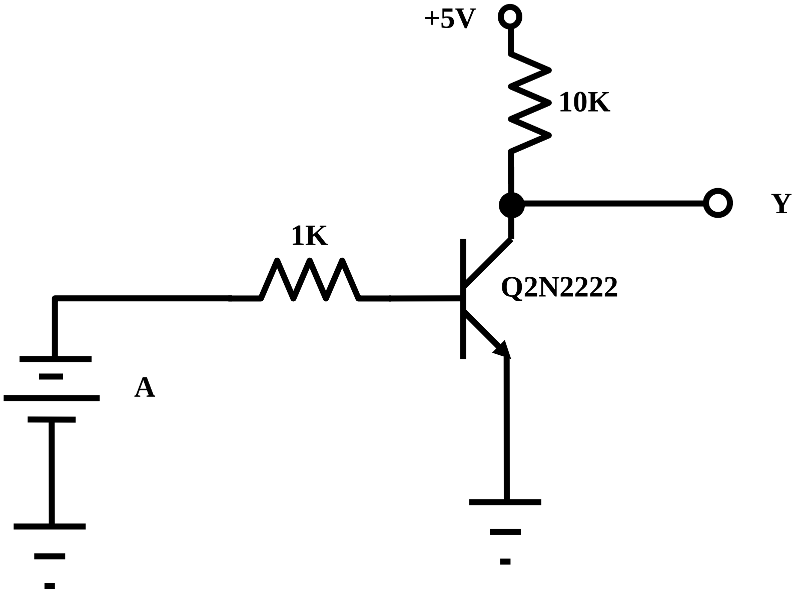


Fig 3: NOT Gate

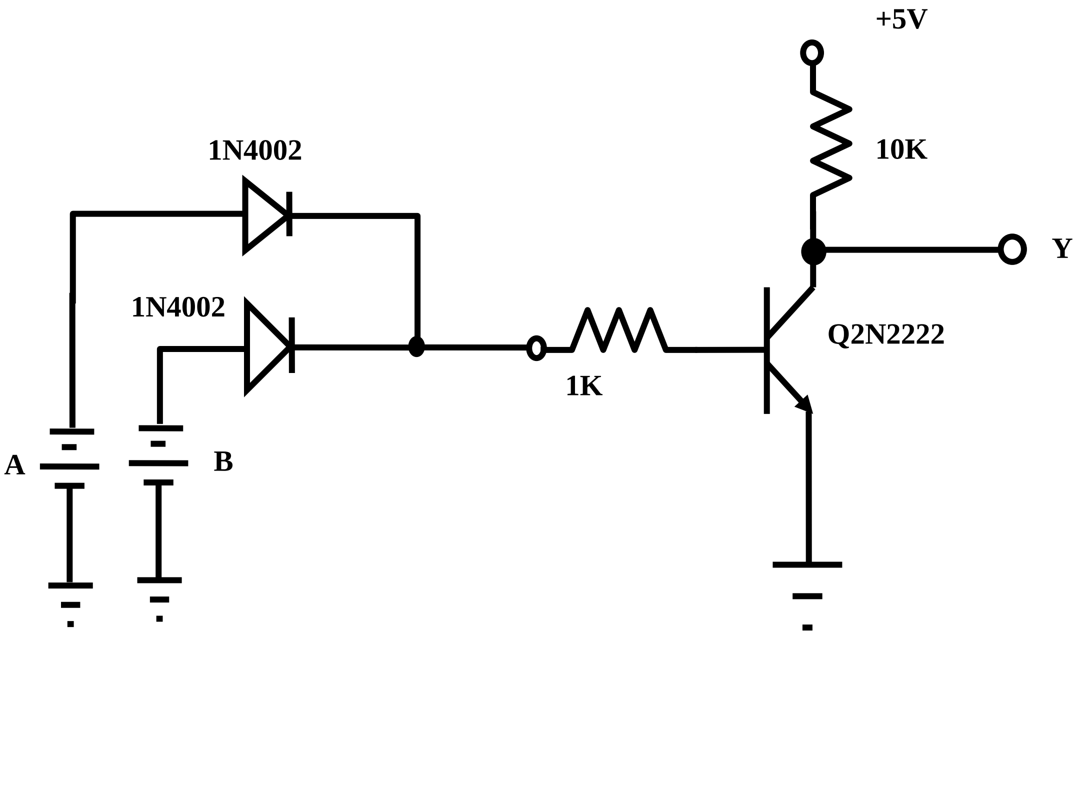


Fig 4: NOR Gate

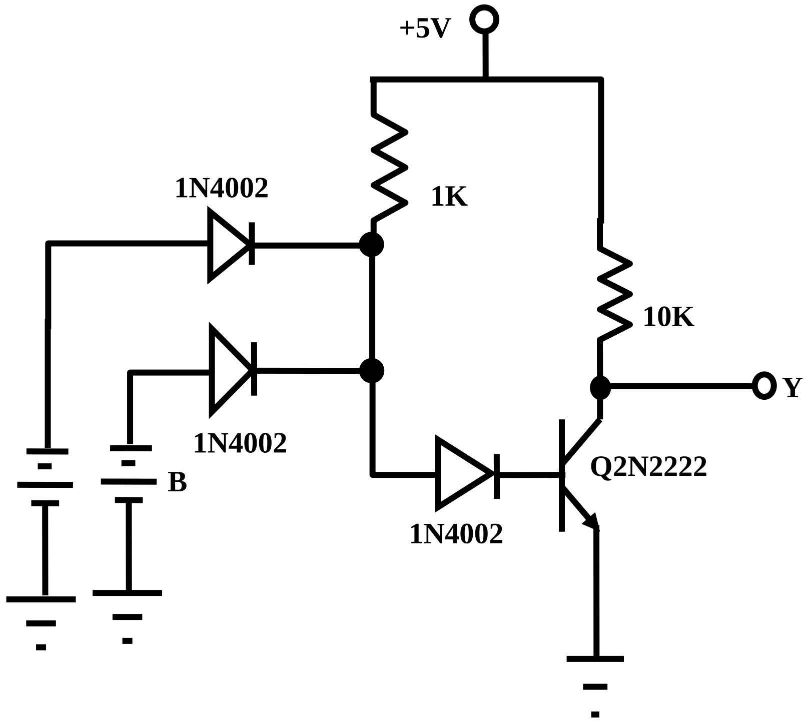


Fig 4: NAND Gate

Truth Tables:

OR Gate

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

AND Gate

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

NOT Gate:

A	Y
0	1
1	0

NOR Gate:

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

NAND Gate:

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Procedure:

- Connect the Circuits as shown above.
- Apply different combinations of inputs and observe their outputs
- Compare with standard truth tables2

Precautions:

- Check the connections before giving the power supply.
- Readings should be observed carefully.

Result:

- Basic Logic Gates are constructed using discrete components and their truth tables are verified

Viva-Voce Questions:

1. Differentiate between positive and negative logic.
2. What is OR Gate? Draw its symbol and write truth table for 3 input OR Gate.
3. What is Exclusive-OR Gate? Draw its symbol.
4. Write a truth table of 2 input Ex-OR Gate
5. What is AND function? Draw symbol of 2 input AND Gate and write its truth table.

2. Realization of Gates using Universal Building Block (NAND only).

Aim:

- To construct logic gates NOT, AND, OR, EX-OR, and EX-NOR using NAND Gates
- Verification of truth tables of NOT, AND, OR, EX-OR, and EX-NOR

Apparatus Required:

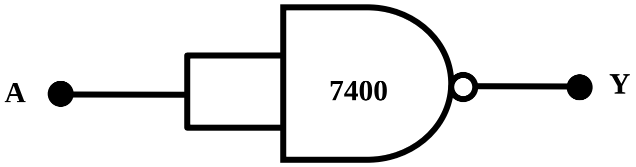
- Digital trainer board
- IC 7400
- Connecting wires

Theory:

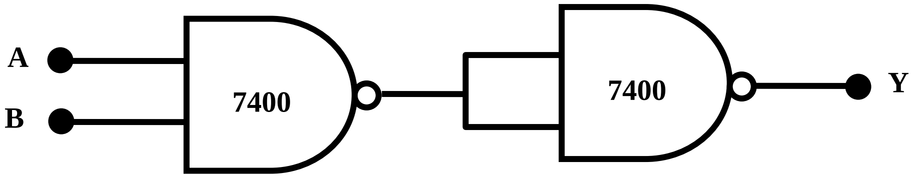
Digital circuits are more frequently constructed with NAND or NOR gates than with AND and OR gates. NAND and NOR gates are easier to fabricate with electronic components and are the basic gates used in all IC digital logic families. So the Boolean functions given in terms of AND, OR and NOT are converted into equivalent gates can be released by because all the available gates can be realized by using NAND and NOR gates that’s why they are called universal gates.

Circuit Diagrams:

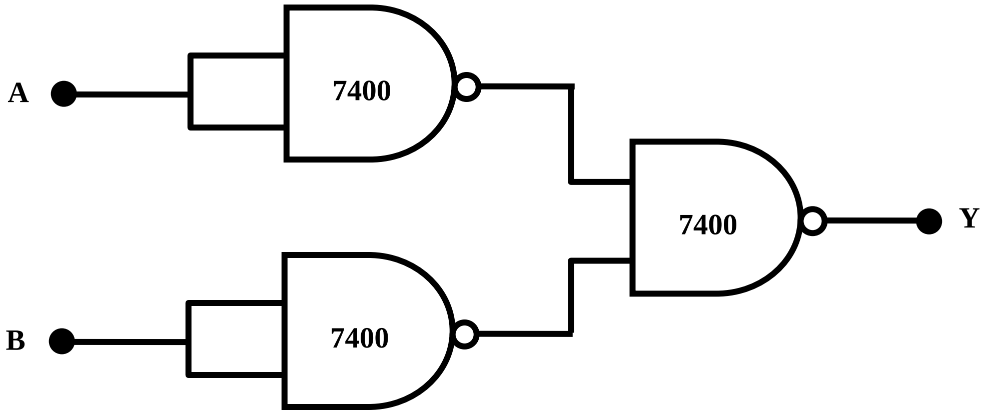
NOT Gate



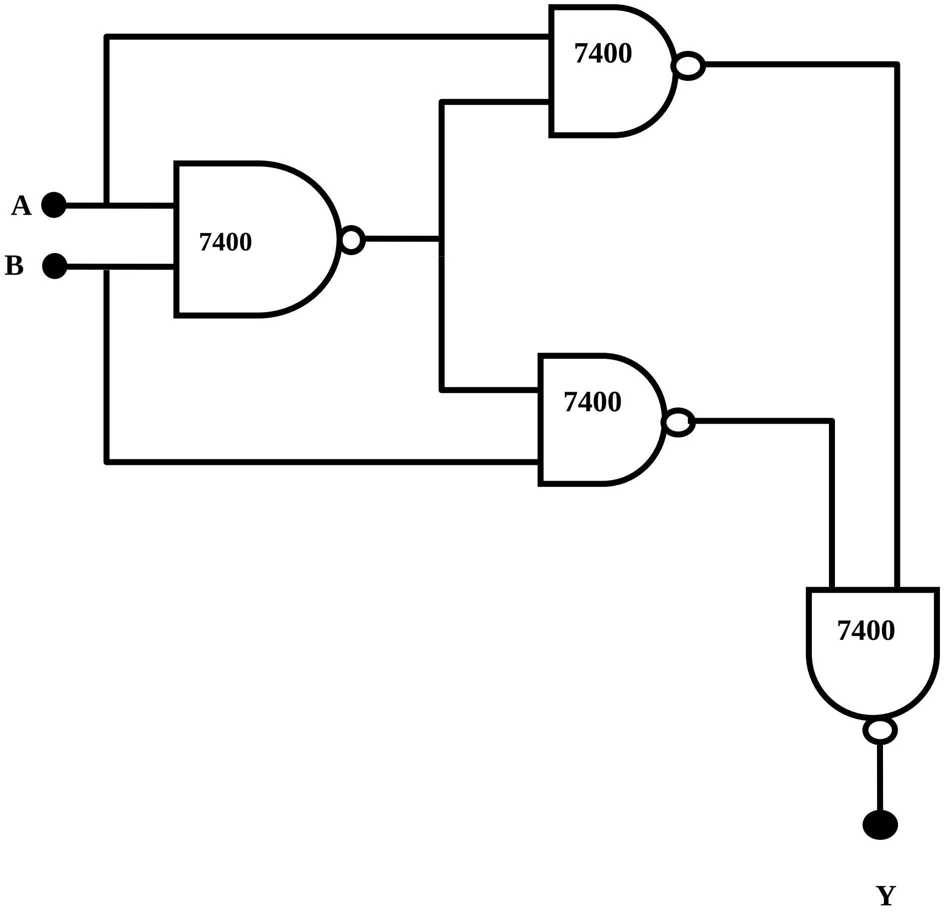
AND Gate



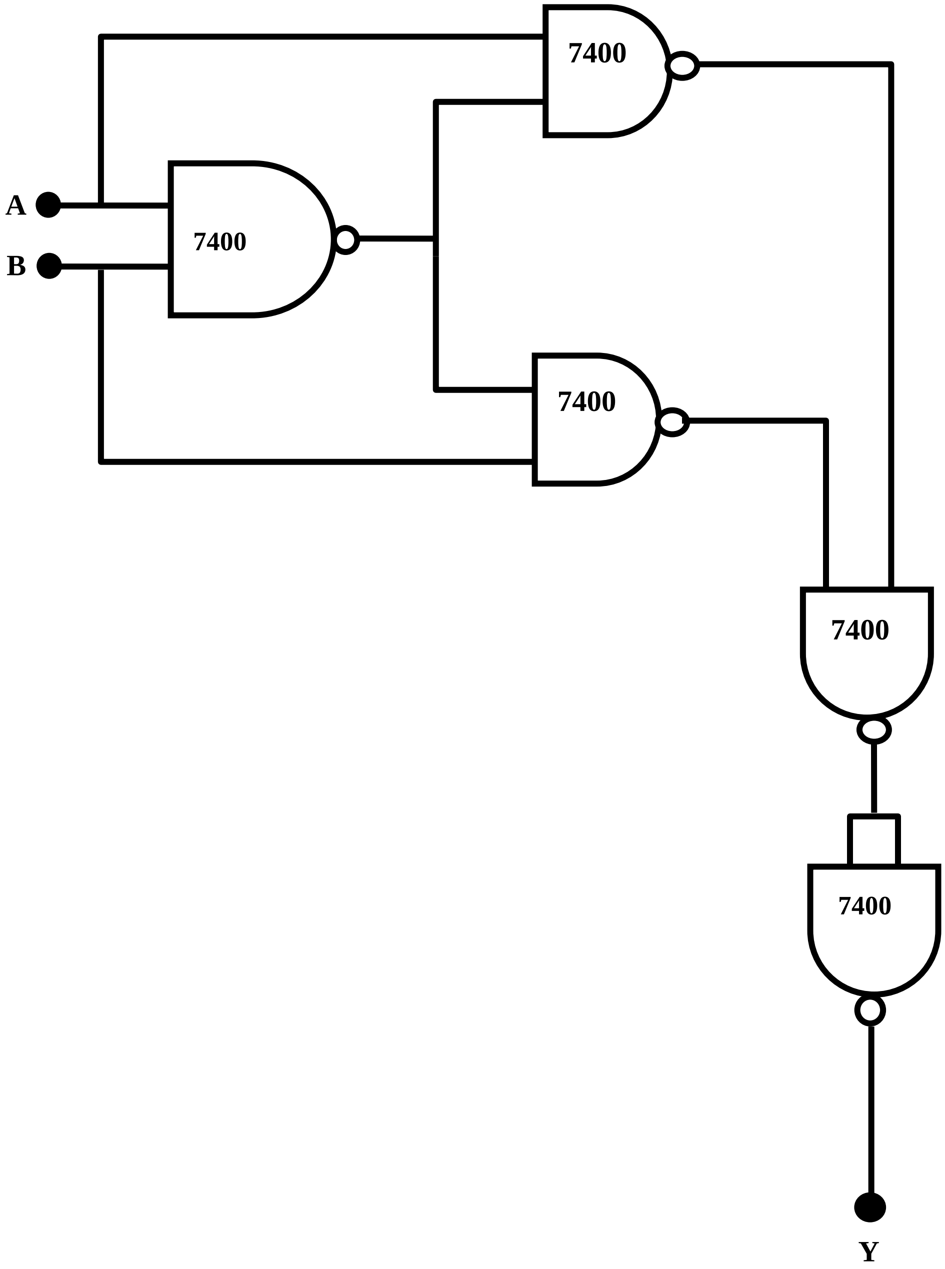
OR Gate



Ex-OR Gate



EX-NOR Gate



Truth Tables:

OR Gate

A	**B**	**Y**
0	0	0
0	1	1
1	0	1
1	1	1

AND Gate

A	**B**	**Y**
0	0	0
0	1	0
1	0	0
1	1	1

NOT Gate:

A	**Y**
0	1
1	0

Ex-OR Gate:

A	**B**	**Y**
0	0	0
0	1	1
1	0	1
1	1	0

Ex-NOR Gate:

A	**B**	**Y**
0	0	1
0	1	0
1	0	0
1	1	1

Procedure:

- Connect the Circuits as shown above.
- Apply different combinations of inputs and observe their outputs
- Compare with standard truth tables2

Precautions:

- Check the connections before giving the power supply.
- Readings should be observed carefully.

Result:

- Basic Logic Gates are constructed using discrete components and their truth tables are verified

Viva-Voce Questions:

1. What is OR Gate? Draw its symbol and write truth table for 3 input OR Gate.
2. What is Exclusive-OR Gate? Draw its symbol.
3. Write a truth table of 2 input Ex-OR Gate
4. Which Gate can be used as an equality detector?
5. Why are NOR and NAND Gates are called universal gates?
6. Draw a symbol of 2 input NOR Gate and write its truth table.
7. Draw a symbol of 2 input NAND Gate and write its truth table.

3. Design of Combinational Logic Circuits

Aim:

- To Design and construct Half-adder, Half-subtractor, Full-adder, and Full-subtractor circuits
- Verification of truth tables of Half-adder, Full-adder, Half-subtractor and Full-subtractor circuits

Apparatus Required:

- Digital trainer board
- ICs- 7486, 7408, 7432, 7404
- Connecting wires

Theory:

Half-adder:

Half-adder is a combinational logic circuit that performs the addition of two bits and produces the outputs sum and carry.

Logical expressions for Half-adder are given below:

Sum Expression: $X \oplus Y$

Carry Expression: XY

Full-adder:

Full-adder is a combinational logic circuit that performs the addition of three bits and produces the outputs sum and carry.

Logical expressions for full-adder are given below:

Sum Expression: $X \oplus Y \oplus Z$

Carry Expression: $(X \oplus Y)Z + XY$

Half-Subtractor:

Half-subtractor is a combinational logic circuit that performs the subtraction of two bits and produces the outputs difference and barrow.

Logical expressions for Half-subtractor are given below:

Difference expression: $X \oplus Y$

Barrow Expression: $\overline{X}Y$

Full-subtractor:

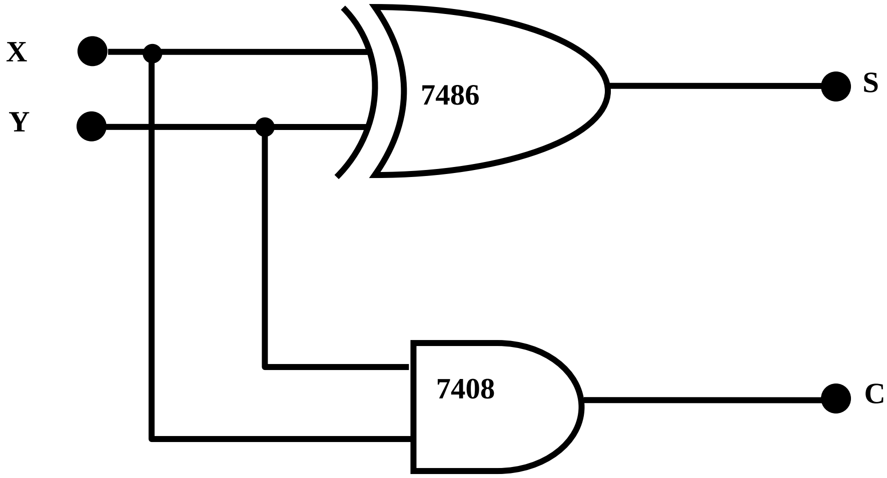
Full-subtractor is a combinational logic circuit that performs the subtraction of three bits and produces the outputs difference and barrow.
Logical expressions for full-subtractor are given below:

Difference expression: $X \oplus Y \oplus Z$

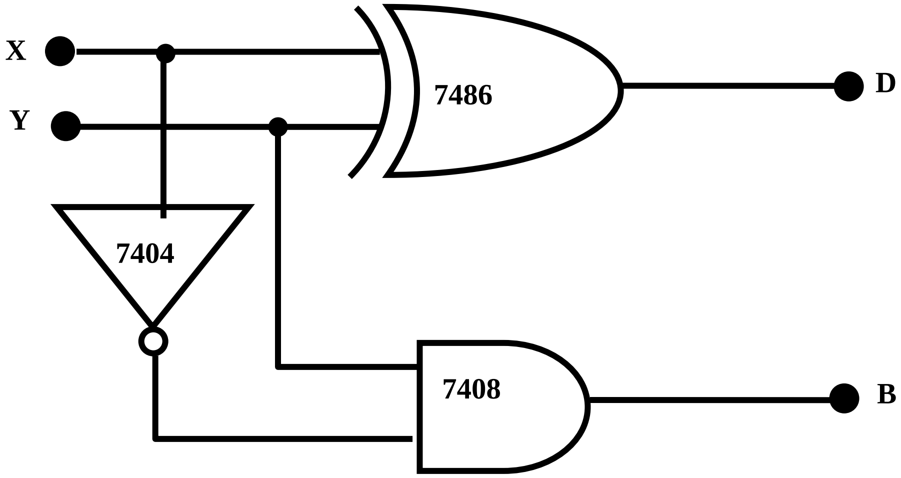
Barrow Expression: $\overline{X}Y + Z(\overline{X \oplus Y})$

Circuit Diagrams:

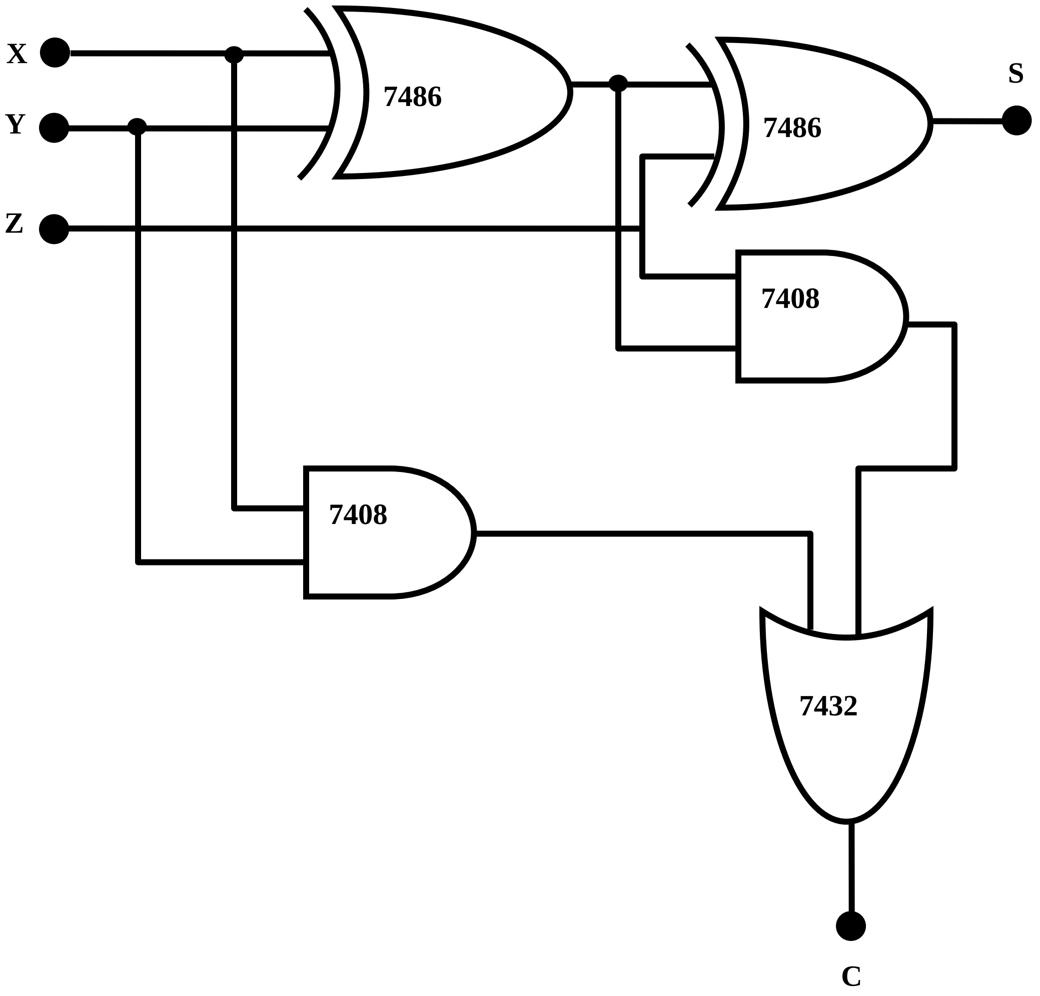
Half-adder



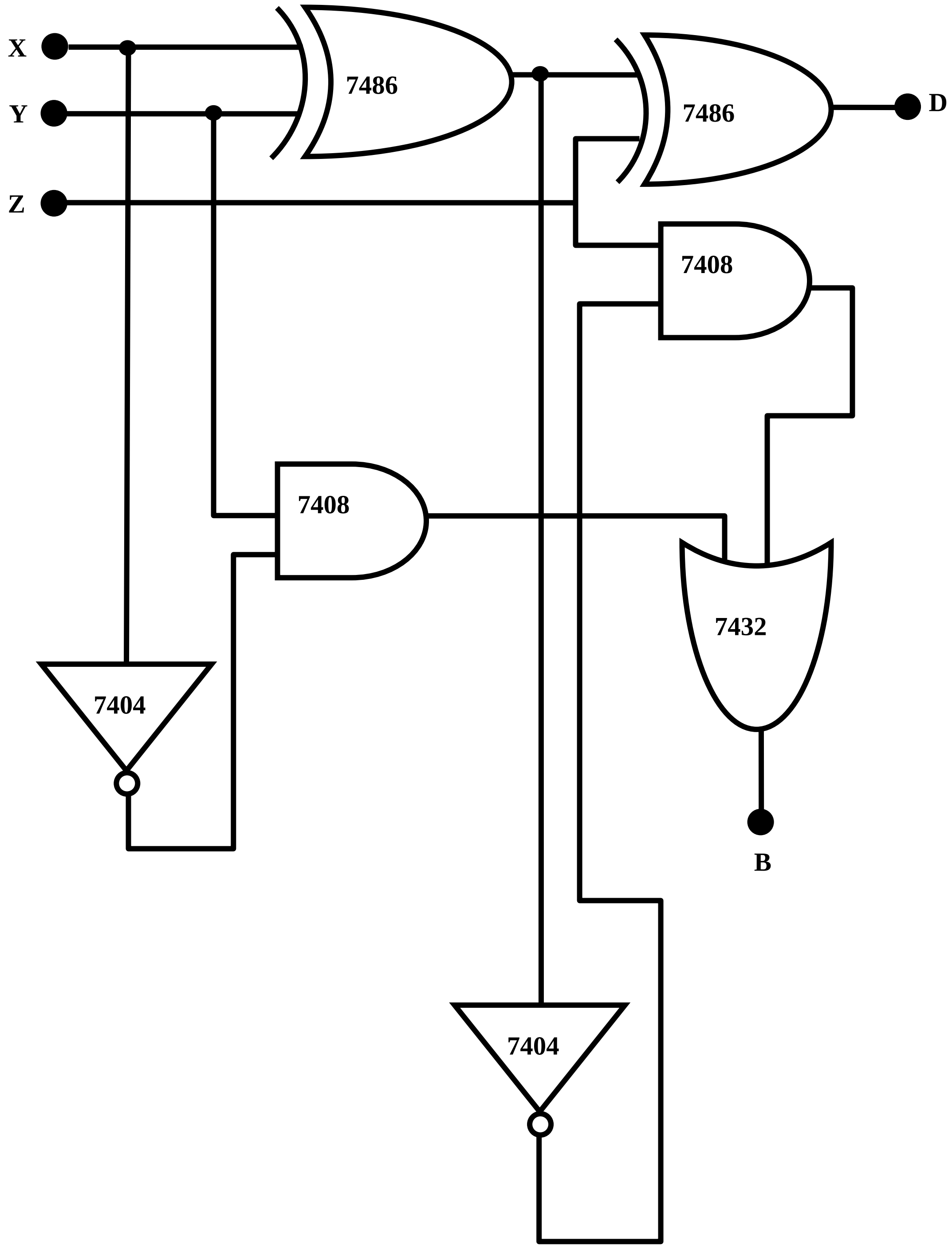
Half-subtractor



Full-adder



Full-subtractor



Truth tables:

Half-adder:

X	Y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Half-subtractor:

X	Y	D	B
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Full-adder:

X	Y	Z	S	C
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

Full-subtractor:

X	Y	Z	D	B
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Procedure:

- Construct the arithmetic circuits
- Connect the constructed circuits as shown above
- Apply different combinations of inputs and observe their outputs
- Compare with standard truth tables

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly
- Readings should be observed carefully.

Result:

- Combinational Logic circuits such as Half-adder, Half-subtractor, Full-adder, and Full-subtractor circuits are constructed and their truth tables are verified

Viva-Voce Questions:

1. What is a combinational logic circuit?
2. Give the expression for sum and carry of a Half-adder
3. Give the expression for sum and carry of a Full-adder
4. Give the expression for difference and barrow of a Half-subtractor
5. Give the expression for difference and barrow of a Full-subtractor
6. Present the steps involved in the design of Combinational circuits.

4. Verification of 4-bit Magnitude comparator IC

Aim:

- To study the operation 4-bit Magnitude comparator IC 7485

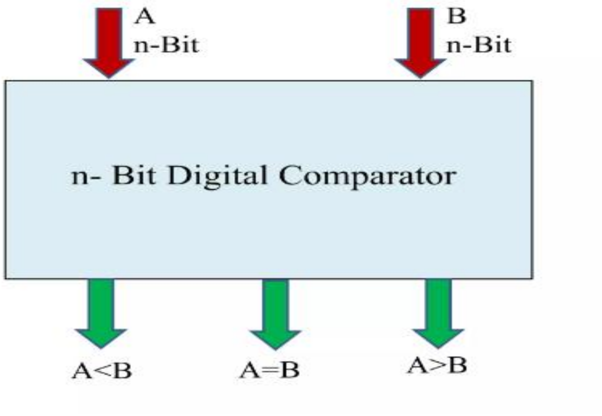
Apparatus Required:

- Digital trainer board
- ICs- 7485
- Connecting wires

Theory:

Magnitude Comparator:

Magnitude comparator is a type of Combinational logic circuit; It Basically compares two binary numbers and determines their relative magnitude. It gives output whether one number is greater than the other, or less than or equal. These comparators are used in digital systems, such as for sorting networks, and decision-making circuits to handle numerical comparisons perfectly without any error.



The circuit works by comparing the bits of the two numbers starting from the most significant bit (MSB) and moving toward the least significant bit (LSB). At each bit position, the two corresponding bits of the numbers are compared. If the bit in the first number is greater than the corresponding bit in the second number, the A>B output is set to 1, and the circuit immediately determines that the first number is greater than the second. Similarly, if the bit in the second number is greater than the corresponding bit in the first number, the A<B output is set to 1, and the circuit immediately determines that the first number is less than the second.

If the two corresponding bits are equal, the circuit moves to the next bit position and compares the next pair of bits. This process continues until all the bits have been compared. If at any point in the comparison, the circuit determines that the first number is greater or less than the second number, the comparison is terminated, and the appropriate output is generated.

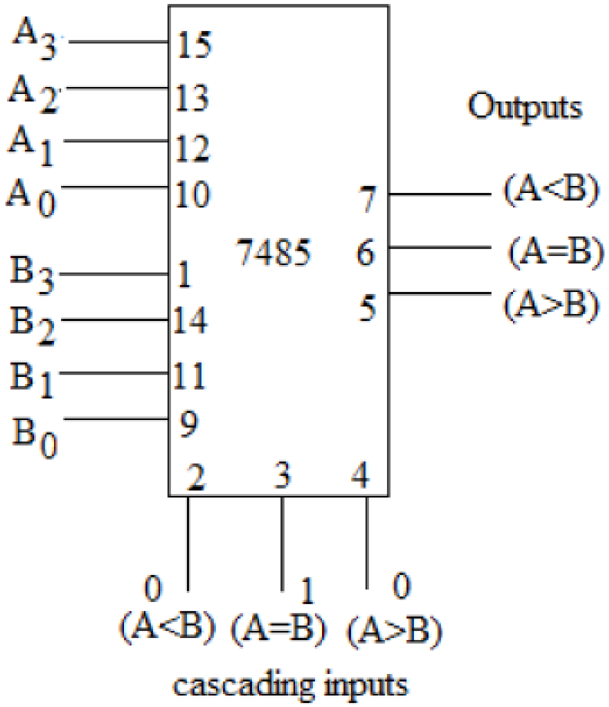
If all the bits are equal, the circuit generates an A=B output, indicating that the two numbers are equal.

There are different ways to implement a magnitude comparator, such as using a combination of XOR, AND, and OR gates, or by using a cascaded arrangement of full adders. The choice of implementation depends on factors such as speed, complexity, and power consumption.

4-Bit Magnitude Comparator IC:

The 74F85 is a 4-bit magnitude comparator that can be expanded to almost any length. It compares two 4-bit binary, BCD, or other monotonic codes and presents the three possible magnitude results at the outputs. The 4-bit inputs are weighted (A0–A3) and (B0–B3) where A3 and B3 are the most significant bits. The operation of the74F85 is described in the Function Table, showing all possible logic conditions. The upper part of the table describes the normal operation under all conditions that will occur in a single device or in a series expansion scheme. In the upper part of the table the three outputs are mutually exclusive. In the lower part of the table, the outputs reflect the feed-forward conditions that exist in the parallel expansion scheme. The expansion inputs IA>B, and IA=B and IA<B are the least significant bit positions. When used for series expansion, the A>B, A=B and A<B outputs of the lease significant word are connected to the corresponding IA>B, IA=B and IA<B inputs of the next higher stage. Stages can be added in this manner to any length, but a propagation delay penalty of about 15ns is added with each additional stage. For proper operation, the expansion inputs of the least significant word should be tied as follows: IA>B = Low

Circuit Diagram:



4-bit Comparator

Procedure:

- Connect the circuit as shown in fig. Feed the 4-bit binary words A_0, A_1, A_2 , A_3 and B_0, B_1 , B_2 , B_3 from the logic input switches.
- Pin 3 of IC 7485 should be at logic 1 to enable compare operation.
- Observe the output A>B, A=B , and A<B on logic indicators. The outputs must be 1 or 0 respectively.
- Repeat the steps 1 ,2 and 3 for various inputs A_0 ,A_1 , A_2 , A_3 and B_0 , B_1 , B_2 , B_3 and observe the outputs at A>B , A=B and A<B .

Precautions:

- Check the connections before giving the power supply.
- Place the IC on the board properly
- Readings should be observed carefully.

Result:

- 4-bit Magnitude comparator IC 7485 operation is verified for different combinations of the two 4-bit numbers A and B

Viva-Voce Questions:

1. What is a combinational logic circuit?
2. What is a Magnitude comparator?
3. Which logic gate is used as equality detector?
4. What are the applications of a comparator?
5. Draw the logic diagram of a 1-bit comparator.

5. Design of 4-2 and 8-3 Encoders

Aim:

- To Design and construct 4-2 and 8-3 encoder circuits
- Verification of truth tables of 4-2 and 8-3 encoder circuits

Apparatus Required:

- Digital trainer board
- ICs- 7432, 4072
- Connecting wires

Theory:

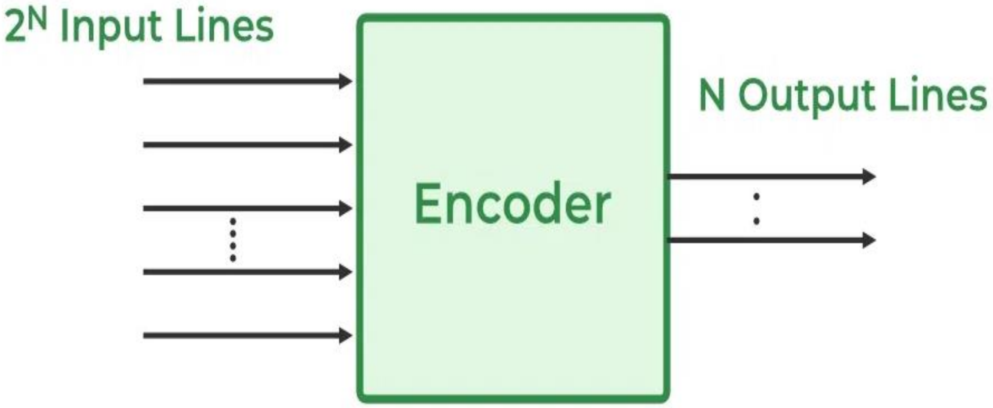
Encoder:

An encoder is a digital circuit that converts a set of binary inputs into a unique binary code. The binary code represents the position of the input and is used to identify the specific input that is active. Encoders are commonly used in digital systems to convert a parallel set of inputs into a serial code.

The basic principle of an encoder is to assign a unique binary code to each possible input. For example, a 2-to-4 line encoder has 2 input lines and 4 output lines and assigns a unique 4-bit binary code to each of the 2^2 = 4 possible input combinations. The output of an encoder is usually active low, meaning that only one output is active (low) at any given time, and the remaining outputs are inactive (high). The active low output is selected based on the binary code assigned to the active input.

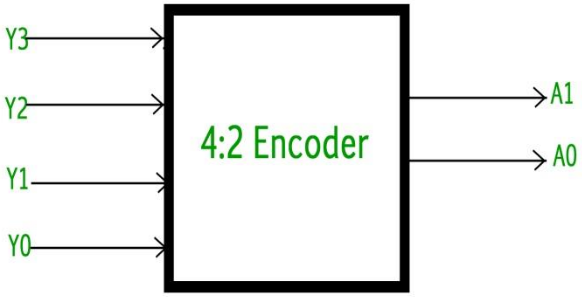
There are different types of encoders, including priority encoders, which assign a priority to each input, and binary-weighted encoders, which use a binary weighting system to assign binary codes to inputs. In summary, an encoder is a digital circuit that converts a set of binary inputs into a unique binary code that represents the position of the input. Encoders are widely used in digital systems to convert parallel inputs into serial codes.

An Encoder is a combinational circuit that performs the reverse operation of a Decoder. It has a maximum of 2^n input lines and ‘n’ output lines, hence it encodes the information from 2^n inputs into an n-bit code. It will produce a binary code equivalent to the input, which is active High. Therefore, the encoder encodes 2^n input lines with ‘n’ bits.



4-2 Encoder:

The 4 to 2 Encoder consists of four inputs Y3, Y2, Y1 & Y0, and two outputs A1 & A0. At any time, only one of these 4 inputs can be '1' in order to get the respective binary code at the output. The figure below shows the logic symbol of the 4 to 2 encoder.



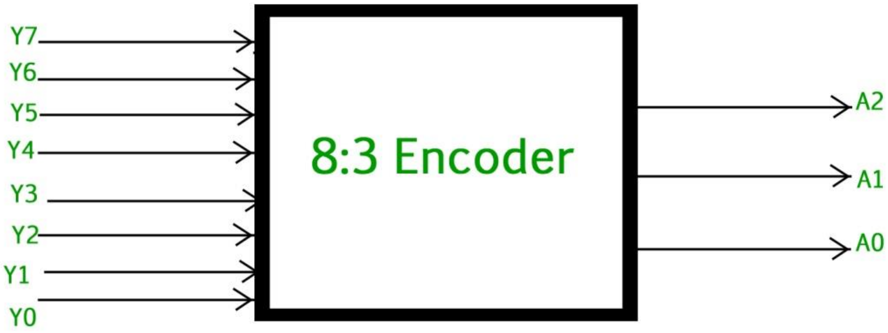
Logical expressions for outputs A1 and A0 are given below:

$$A1 = Y3 + Y2$$
$$A0 = Y3 + Y1$$

The above two logical expressions A1 and A0 can be implemented using two input OR gates.

8-3 Encoder:

The 8 to 3 Encoder or octal to Binary encoder consists of 8 inputs: Y7 to Y0 and 3 outputs: A2, A1 & A0. Each input line corresponds to each octal digit and three outputs generate corresponding binary code. The figure below shows the logic symbol of octal to the binary encoder.



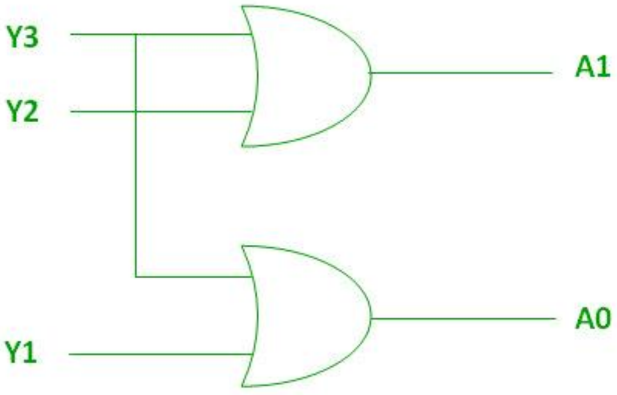
Logical expressions for outputs A2, A1 and A0 are given below:

$$A2 = Y4 + Y5 + Y6 + Y7$$
$$A1 = Y2 + Y3 + Y6 + Y7$$
$$A0 = Y1 + Y3 + Y5 + Y7$$

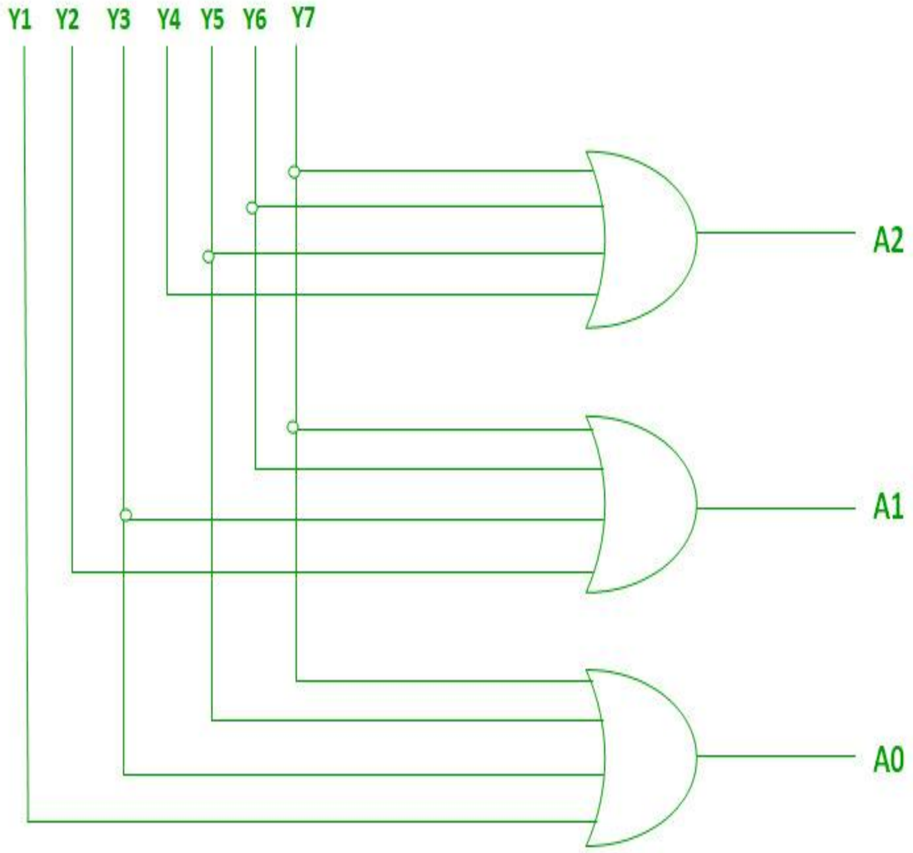
The above three logical expressions A2, A1, and A0 can be implemented using four input OR gates.

Circuit Diagrams:

4-2 Encoder



8-3 Encoder



Truth tables:

4-2 Encoder:

Inputs				**Outputs**	
Y0	**Y1**	**Y2**	**Y3**	**A1**	**A0**
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

8-3 Encoder:

Inputs								Outputs		
Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7	A2	A1	A0
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Procedure:

- Construct the encoder circuits
- Connect the constructed circuits as shown above
- Apply different combinations of inputs and observe their outputs
- Compare with standard truth tables

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- Combinational Logic circuits such as 4-2 and 8-3 encoders are constructed and their truth tables are verified

Viva-Voce Questions:

1. What is a combinational logic circuit?
2. Distinguish Encoder and Decoder
3. How Boolean functions are implemented using decoders?
4. What are the limitations of Encoder?
5. What is a Priority Encoder?

6. Design of BCD-Decimal Decoder

Aim:

- To Design and construct BCD-Decimal decoder
- Verification of truth table of BCD-Decimal decoder Circuit

Apparatus Required:

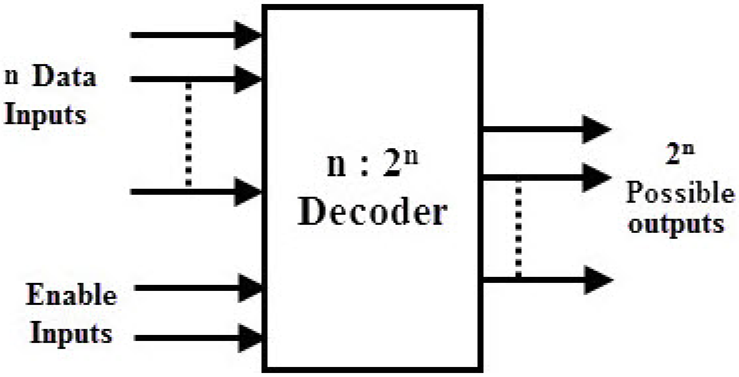
- Digital trainer board
- ICs- 7404, 7421
- Connecting wires

Theory:

Decoder:

A decoder is a type of logic circuit, which converts binary numbers (or binary inputs) to decimal numbers (or decimal outputs). A decoder is actually a code translator, which consists of multiple inputs and outputs and by means of which, some specific number can also be determined. In other words, a decoder is a combinational logic circuit, which detects the presence of a word or binary number. Its input is a parallel binary signal while its output is a single binary signal, which represents the presence or non–presence of a particular binary number. Remember that circuits, both of demultiplexer and decoder are analogous (or decoder and demultiplexer both are similar) except that data input or enabled input or data line does not exist in a decoder and control input bits produce an active output line (i.e. decoder provides just one output at a given time)

We know that information in digital systems is represented by means of binary codes. An “n” bits binary code is capable of 2^n binary elements of coded information. As such, a decoder is a combinational circuit, which changes binary information via “n” input lines up to a maximum of 2^n output lines. The basic function of a decoder is to ascertain the presence of a distinct combination of its input bits (code) and reveal the presence of this code via a certain output level. A decoder generally consists on “n” input lines for control or handling of “n” bits and one 2^n output lines for showing the presence of one or more “n” bit combinations. Decoders are mostly used in computers, calculators and digital clocks etc. As decoders are used for converting binary numbers into decimal numbers, therefore they are normally mounted on the output stage of a digital circuit. Remember that AND gates are applied in a decoder circuit. As output of an AND gate is binary 1 when all its inputs are also binary 1, thus presence or existence of any binary number can be determined by connecting the input of AND gate with its source in an accurate manner. The following figure shows the block diagram of a digital decoder.



BCD-Decimal decoder:

BCD (which is an abbreviation of the binary coded decimal) - BCD-to-decimal is a logic circuit, which can convert every BCD code (8421) to any one of the 10 possible decimal digits (0 – 9). It is generally known as a 4-line-to-10-line decoder or 1 of 10 decoder.

We know that largest and most common application of a decoder circuit is to convert binary numbers to equivalent decimal numbers. For this purpose, BCD to-decimal decoder is extensively used. In order to represent first 10 BCD numbers 0 -9 into ten possible outputs, ten AND gates are used which analyze inputs and provide 10 possible outputs from 0 – 9 by means of decoding it. In other words, as a BCD code represents just 10 decimal numbers 0 – 9, therefore only 10 decoding gates are required for a BCD to decimal decoder circuit. When a BCD number is applied on a decoder’s input, one of ten output lines becomes high (in case of AND gate decoder circuit) which represent the presence of a specific BCD number on input. The output of such decoders is used in order to irradiate its corresponding decimal numbers, so that they could be easily read. Another important point should be kept in mind about BCD that BCD digits are from 0000 to 1001 and all other digits above it (i.e. 1010 to 1111) are not included in BCD code, because the largest digit which can be included in a BCD code is 9.

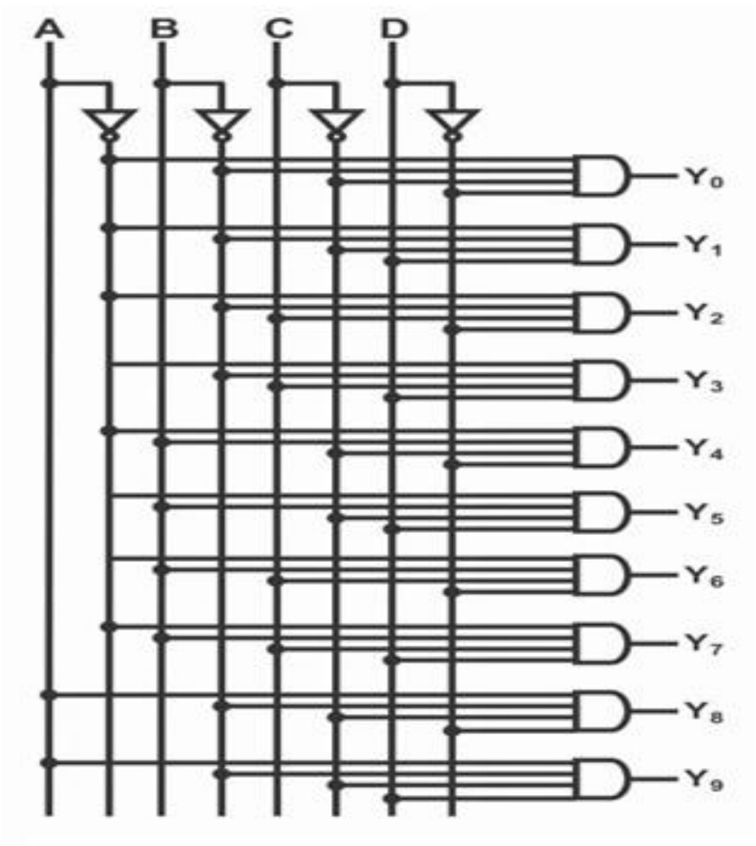
Logical expressions for outputs Y0 to Y9 are given below:

$$Y0 = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D}$$
$$Y1 = \overline{A}\,\overline{B}\,\overline{C}D$$
$$Y2 = \overline{A}\,\overline{B}C\overline{D}$$
$$Y3 = \overline{A}\,\overline{B}CD$$
$$Y4 = \overline{A}B\overline{C}\,\overline{D}$$
$$Y5 = \overline{A}B\overline{C}D$$
$$Y6 = \overline{A}BC\overline{D}$$
$$Y7 = \overline{A}BCD$$
$$Y8 = A\overline{B}\,\overline{C}\,\overline{D}$$
$$Y9 = A\overline{B}\,\overline{C}D$$

The above 10 logical expressions Y0 and Y9 can be implemented using four input AND gates.

Circuit Diagram:

BCD-Decimal Decoder



Truth tables:

BCD-Decimal Decoder:

Inputs				Outputs									
A	**B**	**C**	**D**	**Y0**	**Y1**	**Y2**	**Y3**	**Y4**	**Y5**	**Y6**	**Y7**	**Y8**	**Y9**
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1

Procedure:

- Construct the decoder circuit
- Connect the constructed circuits as shown above
- Apply different combinations of inputs and observe their outputs
- Compare with standard truth table

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- Combinational Logic circuit BCD-decimal decoder is constructed and its truth table is verified

Viva-Voce Questions:

1. What is a combinational logic circuit?
2. Distinguish Encoder and Decoder
3. How Boolean functions are implemented using decoders?
4. What are the limitations of Encoder?
5. Distinguish normal binary and BCD codes.

7. Design of Code converters

Aim:

- To Design and construct 4:1 Multiplexer and 1:4 De-Multiplexer circuits
- Verification of truth tables of 4:1 Multiplexer and 1:4 De-Multiplexer circuits

Apparatus Required:

- Digital trainer board
- ICs- 7486
- Connecting wires

Theory:

Code Converters are the digital circuits that are designed to translate data representation from one format to the other format. The Binary code is the numerical system used in digital electronics. It only consists of two Symbols which are 0 and 1. The binary code is serves as the basis for encoding text, number and various other types of data in the digital devices. Gray code system is a binary number system in which every successive pair of numbers differs by only one bit. It is also named as unit Hamming distance code or Reflecting code or cyclic code. The Binary-to-Gray code converter takes binary input and translates it to its corresponding gray code representation. The Gray-to-Binary code converter takes gray input and translates it to its corresponding binary code representation.

Binary-Gray Code Converter:

Binary-Gray code converter is a combinational logic circuit that converts the normal binary code into equivalent gray code

Logical expressions for gray coded bits of 4-bit binary to gray code converter are given below:

$$G3 = B3$$
$$G2 = B3 \oplus B2$$
$$G1 = B2 \oplus B1$$
$$G0 = B1 \oplus B0$$

Gray-Binary Code Converter:

Gray-Binary code converter is a combinational logic circuit that converts the gray code into equivalent normal binary code

Logical expressions for binary coded bits of 4-bit gray to binary code converter are given below:

$B3 = G3$

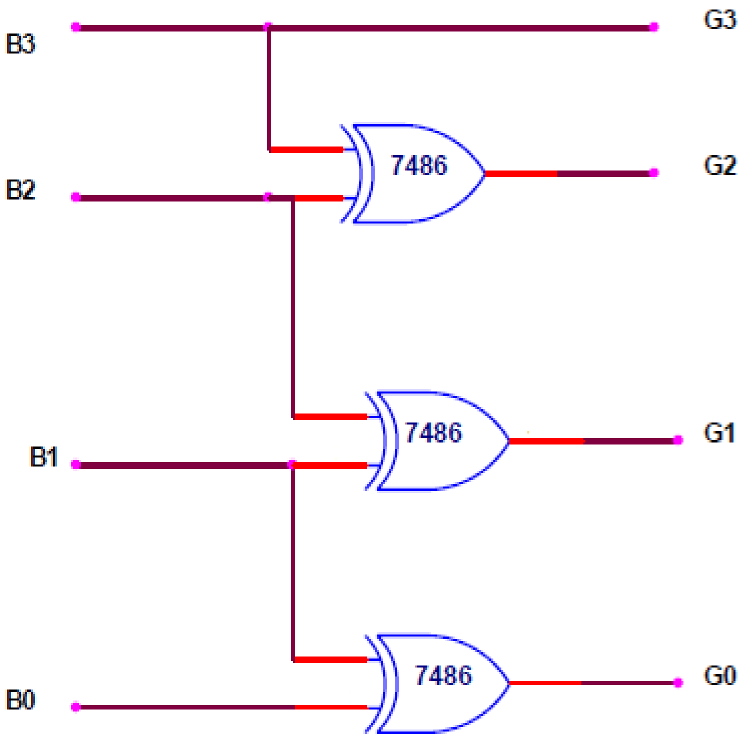
$B2 = B3 \oplus G2$

$B1 = B2 \oplus G1$

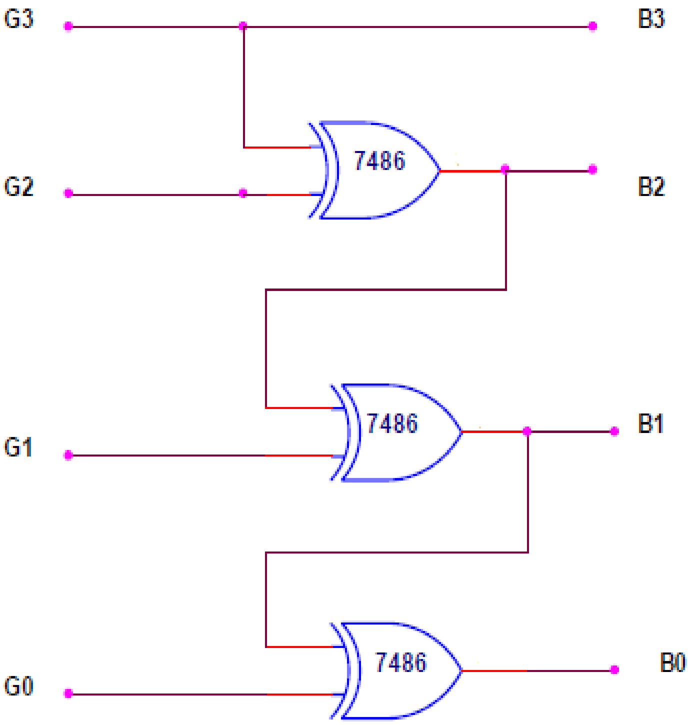
$B0 = B1 \oplus G0$

Circuit Diagrams:

Binary to Gray Code Converter



Gray to Binary Code Converter



Truth tables:

4-bit Binary-Gray code converter:

Inputs				Outputs			
B3	**B2**	**B1**	**B0**	**G3**	**G2**	**G1**	**G0**
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	0
0	1	0	1	0	1	1	1
0	1	1	0	0	1	0	1
0	1	1	1	0	1	0	0
1	0	0	0	1	1	0	0
1	0	0	1	1	1	0	1
1	0	1	0	1	1	1	1
1	0	1	1	1	1	1	0
1	1	0	0	1	0	1	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	0	1
1	1	1	1	1	0	0	0

4-bit Gray - Binary code converter:

Inputs				**Outputs**			
G3	**G2**	**G1**	**G0**	**B3**	**B2**	**B1**	**B0**
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	1
0	0	1	1	0	0	1	0
0	1	0	0	0	1	1	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	0	0
0	1	1	1	0	1	0	1
1	0	0	0	1	1	1	1
1	0	0	1	1	1	1	0
1	0	1	0	1	1	0	0
1	0	1	1	1	1	0	1
1	1	0	0	1	0	0	0
1	1	0	1	1	0	0	1
1	1	1	0	1	0	1	1
1	1	1	1	1	0	1	0

Procedure:

- Construct the code converter circuits
- Connect the constructed circuits as shown above
- Apply different combinations of inputs and observe their outputs
- Compare with standard truth tables

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- Combinational Logic circuits such as Binary-Gray &Gray-Binary code converters are constructed and their truth tables are verified

Viva-Voce Questions:

1. What is a Gray code?
2. Convert the decimal number 25 into equivalent gray coded number
3. Convert the $(100100)_2$ into Gray code
4. Convert $(11001)_{Gray}$ into Binary
5. Mention the applications of Gray codes.

8. Design of Multiplexer and De-Multiplexer

Aim:

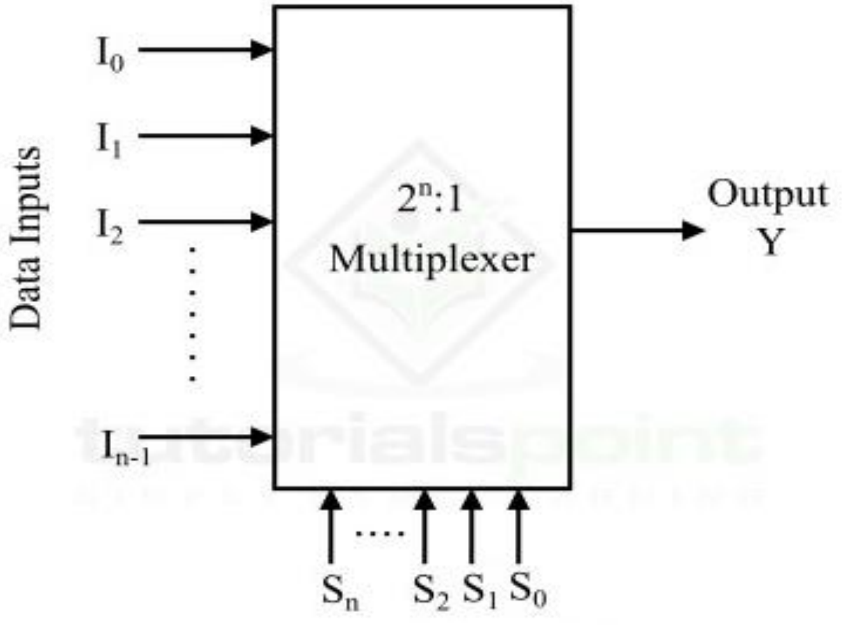
- To Design and construct Binary-Gray & Gray-Binary Code converter circuits
- Verification of truth tables of Binary-Gray & Gray-Binary converter circuits

Apparatus Required:

- Digital trainer board
- ICs- 7404,7411,7432,7408
- Connecting wires

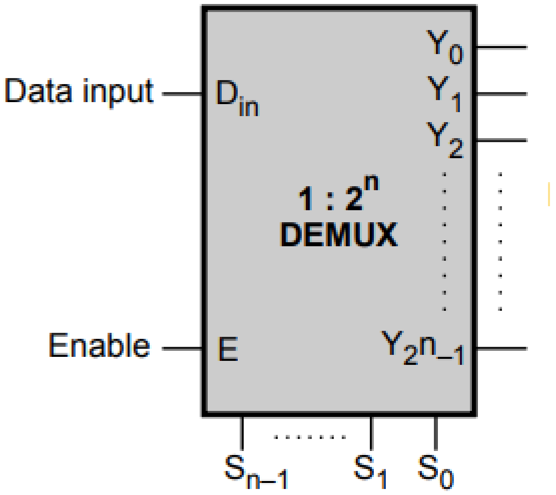
Theory:

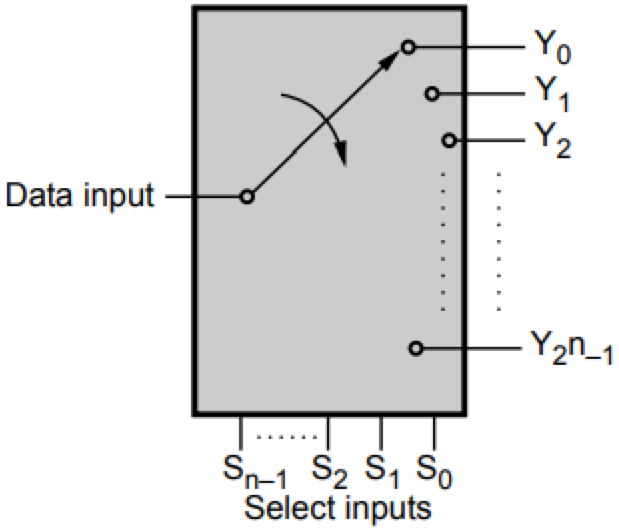
A multiplexer is a combinational circuit that has many data inputs and a single output, depending on control or select inputs. Multiplexers are also known as data selector or parallel-to-serial converters, many-to-one circuits, and universal logic circuits. In general an N:1 Multiplexer has '2^n' inputs, 'n' selections lines and One output line. They are mainly used to increase the amount of data that can be sent over a network within a certain amount of time and bandwidth. The following figure shows the block diagram of a Multiplexer



Digital Multiplexer

De-Multiplexer is a data distributor combinational circuit. It works in a reverse way of the Multiplexer. The DEMUX has 1 input port and 2^n output lines. Here 'n' signifies the selection line for a DEMUX. As per the selection line value, the DEMUX input lines will be connected to receive the output. Demultiplexer receives digital information from a single source and converts it into several sources. The following figures shows the block diagram & equivalent circuit of De-Multiplexer

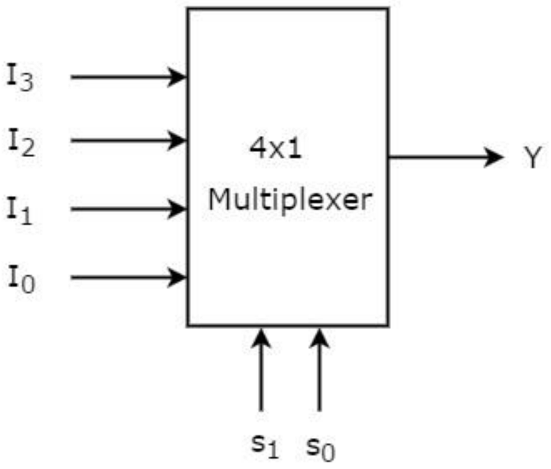




4:1 Multiplexer:

The 4×1 Multiplexer which is also known as the 4-to-1 multiplexer. It is a multiplexer that has 4 inputs and a single output. The Output is selected as one of the 4 inputs which is based on the selection inputs. The number of the Selection lines will depend on the number of the input which is determined by the equation

$\log_2 2^n$ .So we have two selection lines for 4:1 MUX. The following figure shows the block diagram of a 4:1 MUX



The output of the multiplexer is determined by the binary value of the selection lines

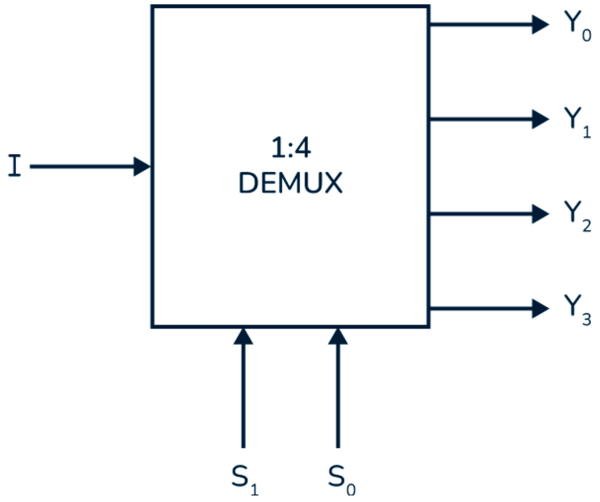
- When S_1S_0=00, the input I_0 is selected.
- When S_1S_0=01, the input I_1 is selected.
- When S_1S_0=10, the input I_2 is selected.
- When S_1S_0=11, the input I_3 is selected.

The logical expression for output Y is given below:

$$Y = \overline{S}_1\overline{S}_0 I_0 + \overline{S}_1 S_0 I_1 + S_1\overline{S}_0 I_2 + S_1 S_0 I_3$$

1:4 De-Multiplexer:

A 1:4 DEMUX has only one input which is denoted as I. There are two selection lines i.e. S_1 and S_0. At last, the DEMUX has output lines including Y_3, Y_2, Y_1 &Y_0. The following figure shows the block diagram of a 1:4 De-MUX.

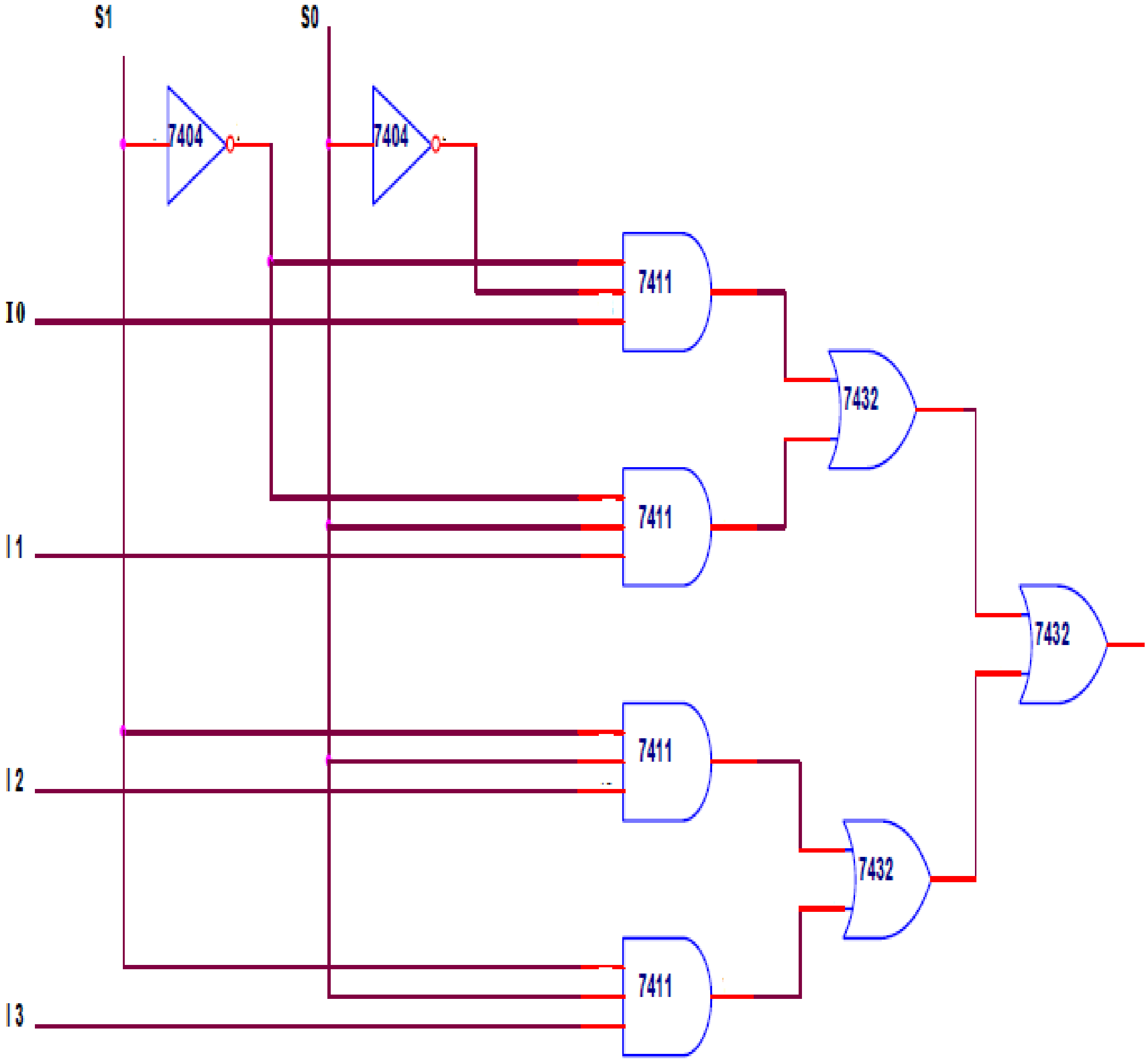


The logical expressions for outputs are given below:

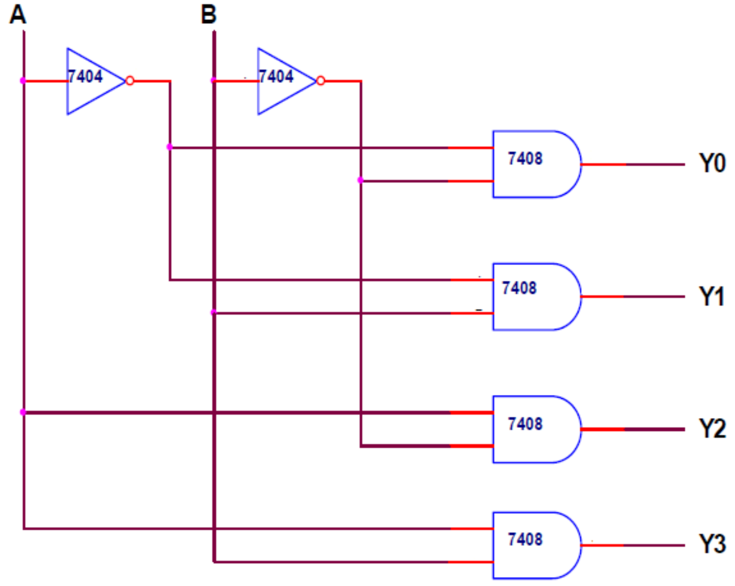
$$Y_0 = \overline{S}_1\overline{S}_0 = \overline{A}\overline{B}$$
$$Y_1 = \overline{S}_1 S_0 = \overline{A}B$$
$$Y_2 = S_1\overline{S}_0 = A\overline{B}$$
$$Y_3 = S_1 S_0 = AB$$

Circuit Diagrams:

4:1 Multiplexer



1:4 De-MUX



Truth tables:

4:1 MUX:

Select	Inputs	Output
S1	S0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

1:4 De-MUX

Inputs		Outputs			
S_1 (A)	S_0 (B)	Y_0	Y_1	Y_2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Procedure:

- Construct the Multiplexer and De-Multiplexer logic circuits
- Connect the constructed circuits as shown above
- Apply different combinations of inputs and observe their outputs
- Compare with standard truth tables

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- Combinational Logic circuits such as 4:1 MUX and 1:4 De-MUX circuits are constructed and their truth tables are verified

Viva-Voce Questions:

1. What is a MUX?
2. What is De-MUX?
3. Distinguish Decoder and De-MUX
4. Implement half-adder using 4:1 MUX
5. Compare Logic design using decoder and multiplexer

9. Verification of Truth tables of Flip-flops

Aim:

- To verify the truth tables of flip-flops that are constructed using logic gates

Apparatus Required:

- Digital trainer board
- ICs- 7400,7410
- Connecting wires

Theory:

Flip-flops:

A flip-flop is a circuit with two stable states that can be used to store binary data. The stored data can be changed by applying varying inputs. Flip-flops and latches are fundamental building blocks of digital systems used in computers, communications, and many other types of systems. Both are used as data storage elements. The primary difference between a latch and a flip-flop is a gating or clocking mechanism. The difference between latch and flip flop is that a latch is level-triggered (outputs can change as soon as the inputs change) and Flip-Flop is edge-triggered (only changes state when a control signal goes from high to low or low to high).

SR Flip-flop:

This is the most common flip-flop among all. This simple flip-flop circuit has a set input (S) and a reset input (R). In this system, when you Set "S" as active, the output "Q" would be high, and "Q'" would be low. Once the outputs are established, the wiring of the circuit is maintained until "S" or "R" goes high, or power is turned off.

The characteristic equation for SR flip-flop is $Q_{n+1} = S + \overline{R}Q_n$

JK Flip-flop:

The JK flip-flop augments the behavior of the SR flip-flop (J=Set, K=Reset) by interpreting the J = K = 1 condition as a "flip" or toggle command. Specifically, the combination J = 1, K = 0 is a command to set the flip-flop; the combination J = 0, K = 1 is a command to reset the flip-flop; and the combination J = K = 1 is a command to toggle the flip-flop, i.e., change its output to the logical complement of its current value. Setting J = K = 0 maintains the current state. To synthesize a D flip-flop, simply set K equal to the complement of J. Similarly, to synthesize a T flip-flop, set K equal to J. The JK flip-flop is therefore a universal flip-flop, because it can be configured to work as an SR flip-flop, a D flip-flop, or a T flip-flop.

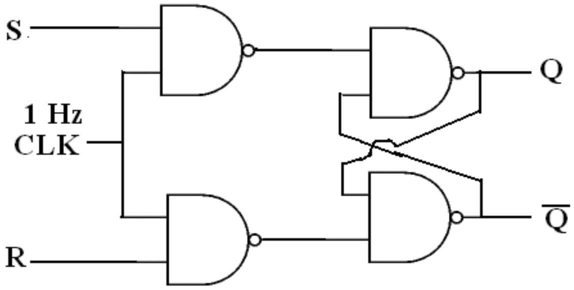
The characteristic equation of the JK flip-flop is $Q_{n+1} = J\overline{Q}_n + \overline{K}Q_n$

D Flip-flop:

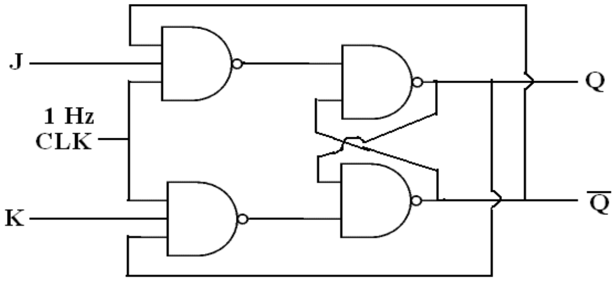
A D-type flip-flop is a clocked flip-flop which has two stable states. A D-type flip-flop operates with a delay in input by one clock cycle. Thus, by cascading many D-type flip-flops delay circuits can be created, which are used in many applications such as in digital television systems. Characteristic Equation is Q_{n+1}= D.

Circuit Diagrams:

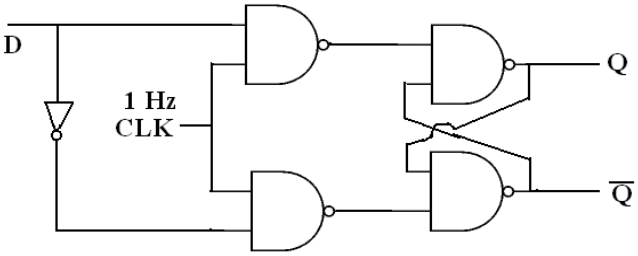
SR Flip-flop



JK Flip-flop:



D Flip-flop:



Truth tables:

SR Flip-flop:

S	R	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	Forbidden

JK Flip-flop

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_n$

D Flip-flop:

D	Q_{n+1}
0	0
1	1

Procedure:

- Construct the Flip-flop circuits using logic gates
- Connect the constructed circuits as per diagram shown above
- Verify the truth table by applying inputs and clock pulse.

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- SR, JK, D flip-flops are constructed and their truth tables are verified

Viva-Voce Questions:

1. What is a Flip-flop?
2. What is a Latch?
3. Distinguish Flip-flop and Latch
4. What is a Characteristic equation of a Flip-flop?
5. What is meant by Race around condition in JK fFlip-flop?

10. Design of Shift registers using Flip-flops

Aim:

To construct the following 4 bit shift registers (i) Serial in serial out (ii) Serial in parallel out (iii) Parallel in serial out (iv) Parallel in parallel out

Apparatus Required:

- Digital trainer board
- ICs- 7474,7432
- Connecting wires

Theory:

The Shift Register is another type of sequential logic circuit that can be used for the storage or the transfer of data in the form of binary numbers. This sequential device loads the data present on its inputs and then moves or "shifts" it to its output once every clock cycle, hence the name "shift register". A shift register basically consists of several single bit "D-Type Data Latches", one for each data bit, either a logic "0" or a "1", connected together in a serial type daisy-chain arrangement so that the output from one data latch becomes the input of the next latch and so on. Data bits may be fed in or out of a shift register serially, that is one after the other from either the left or the right direction, or all together at the same time in a parallel configuration.

The number of individual data latches required to make up a single Shift Register device is usually determined by the number of bits to be stored with the most common being 8-bits (one byte) wide constructed from eight individual data latches.

Shift Registers are used for data storage or for the movement of data and are therefore commonly used inside calculators or computers to store data such as two binary numbers before they are added together, or to convert the data from either a serial to parallel or parallel to serial format. The individual data latches that make up a single shift register are all driven by a common clock (Clk) signal making them synchronous devices.

Shift register IC's are generally provided with a *clear* or *reset* connection so that they can be "SET" or "RESET" as required. Generally, shift registers operate in one of four different modes with the basic movement of data through a shift register being.

Serial-in to Serial-out (SISO):

The data is shifted serially “IN” and “OUT” of the register, one bit at a time in either a left or right direction under clock control.

Serial-in to Parallel-out (SIPO):

The register is loaded with serial data, one bit at a time, with the stored data being available at the output in parallel form.

Parallel-in to Serial-out (PISO):

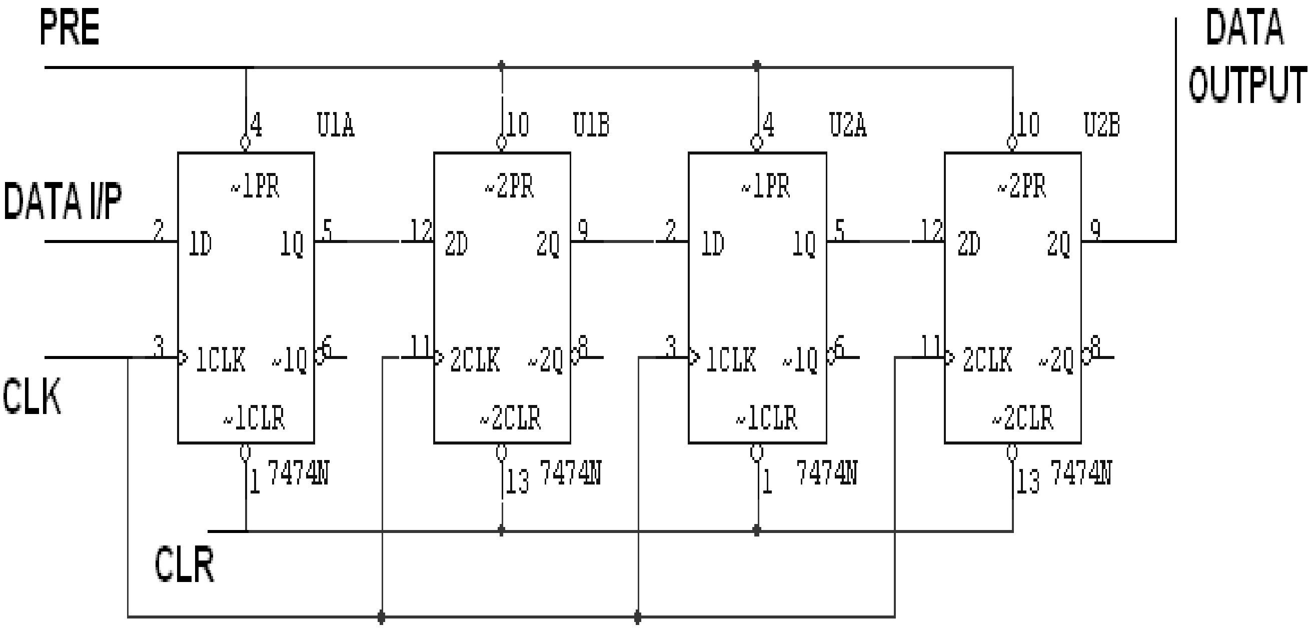
The parallel data is loaded into the register simultaneously and is shifted out of the register serially one bit at a time under clock control.

Parallel-in to Parallel-out (PIPO):

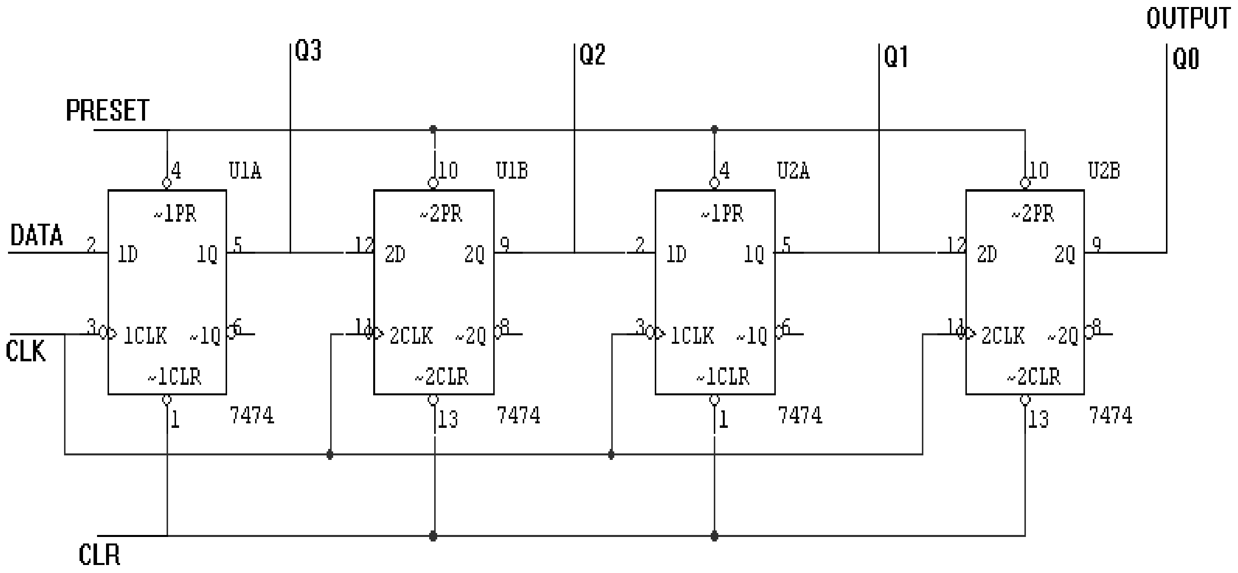
The parallel data is loaded simultaneously into the register, and transferred together to their respective outputs by the same clock pulse.

Circuit Diagrams:

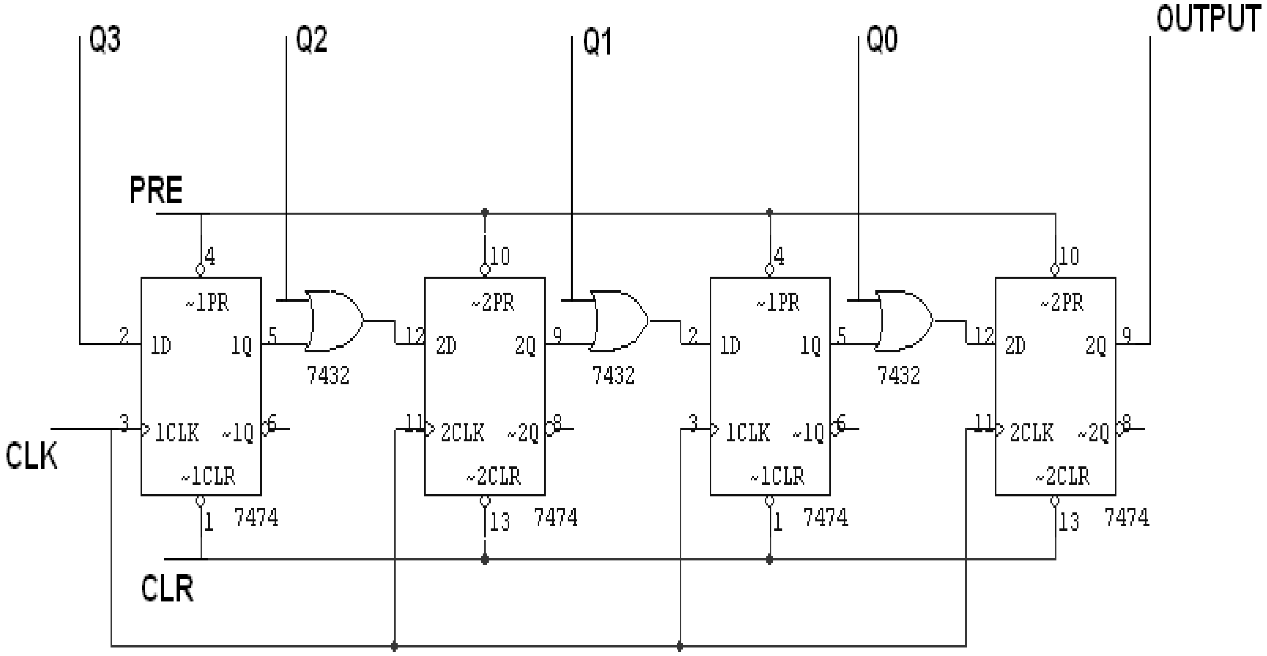
SISO Shift Register



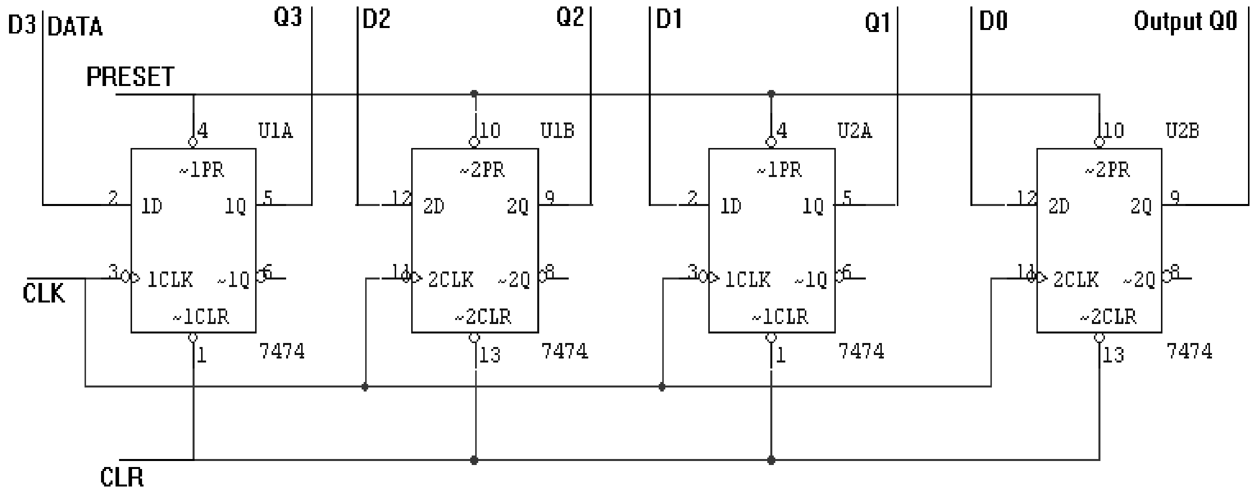
SIPO Shift Register



PISO Shift Register



PIPO Shift Register



Truth tables:

SISO Shift Register:

CLK	Serial in	Serial out
1	1	0
2	0	0
3	0	0
4	1	1
5	X	0
6	X	0
7	X	1

SIPO Shift Register:

CLK	DATA	OUTPUT			
		Q_A	Q_B	Q_C	Q_D
1	1	1	0	0	0
2	0	0	1	0	0
3	0	0	0	1	1
4	1	1	0	0	1

PISO Shift Register:

CLK	Q3	Q2	Q1	Q0	O/P
0	1	0	0	1	1
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	1

PIPO Shift Register:

CLK	DATA INPUT				OUTPUT			
	D_A	D_B	D_C	D_D	Q_A	Q_B	Q_C	Q_D
1	1	0	0	1	1	0	0	1
2	1	0	1	0	1	0	1	0

Procedure:

- Connections are given as per circuit diagram.
- For each shift register, inputs and outputs are connected with at most care.
- The input bits to the shift registers are given.
- Observe the output and verify the truth table.
- Continue this for different inputs.

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- Constructed different types of 4 bit shift registers and verified the truth tables.

Viva-Voce Questions:

1. What is a Register?
2. What is a Shift Register?
3. List Different types of shift register?
4. What is a bi-directional shift register?
5. What is a Universal Shift register?
6. List out various shift counters.

11. Design of Ring & Johnson Counters using Flip-flops

Aim:

- To construct a Ring Counter and Johnson counter
- To verify the truth tables of Ring and Johnson counters.

Apparatus Required:

- Digital trainer board
- ICs- 7474
- Connecting wires

Theory:

Shift Register Counter:

A shift register counter is basically a shift register with the serial output connected back to the serial input to produce special sequences. These devices are often classified as counters because they exhibit a specified sequence of states. Two of the most common types of shift register counters, the Johnson counter and the ring counter.

Ring Counter:

A ring counter is formed by feeding the output of a shift register to its own input. Here the last output i.e. Q_D in a shift register is connected back to the serial input. The data pattern enclosed within the shift register will re-circulate with respect to the clock pulse. Ring counter is one of the shift register applications. A ring counter has N states where 'N' is the number of flip-flops.

The synchronous Ring Counter is preset so that exactly one data bit in the register is set to logic "1" with all the other bits reset to "0". To achieve this, a "CLEAR" signal is firstly applied to all the flip-flops together in order to "RESET" their outputs to a logic "0" level and then a "PRESET" pulse is applied to the input of the first flip-flop (FFA) before the clock pulses are applied. This then places a single logic "1" value into the circuit of the ring counter.

So on each successive clock pulse, the counter circulates the same data bit between the four flip-flops over and over again around the "ring" every fourth clock cycle. But in order to cycle the data correctly around the counter we must first "load" the counter with a suitable data pattern as all logic “0’s” or all logic “1’s” outputted at each clock cycle would make the ring counter invalid.

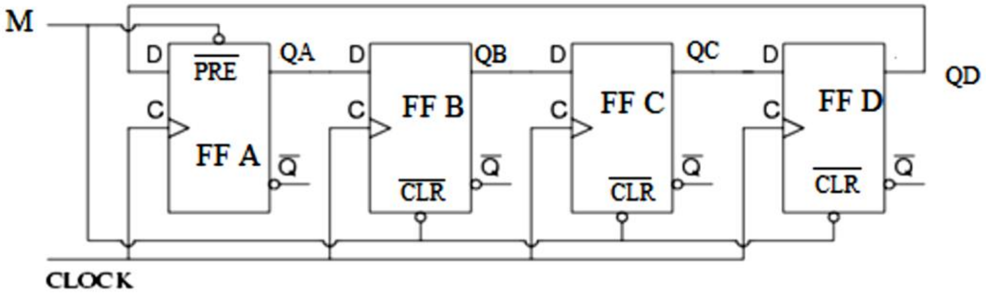
Twisted Ring Counter (or) Johnson Counter:

The Johnson digital counter or Twisted Ring Counter is a synchronous shift register with feedback from the inverted output () of the last flip-flop. of the last flip flop is connected back to the input D of the first flip-flop. This inversion of Q before it is fed back to input D causes the counter to “count” in a special way. The main benefit of this type of counter is that it only needs half the number of flip flops compared to that of standard ring counter to represent many states. So an n-stage Johnson counter gives a sequence of 2n different states and can therefore be treated as a “Mod 2n counter” whereas an n-stage ring counter has only ‘n’ states that is “Mod n counter”.

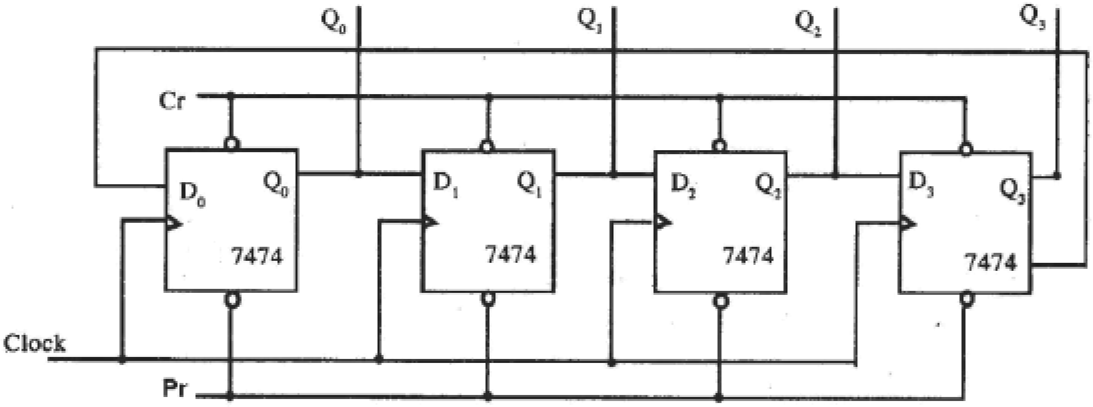
It can be implemented using D-type flip-flops (or JK-type flip-flops). The output of each flip flop is connected to the input of the next. The complementary output of the last flip-flop is connected to the first input.

Circuit Diagrams:

Ring Counter

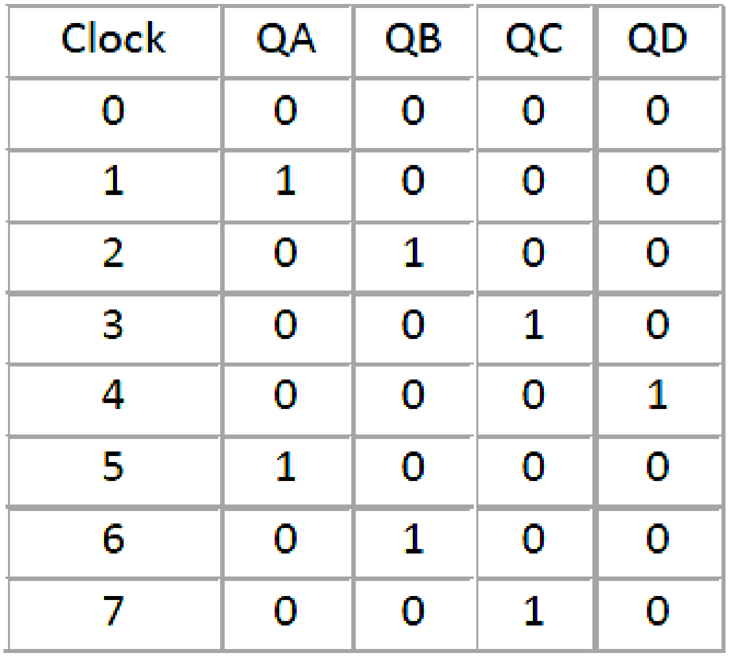


Johnson Counter



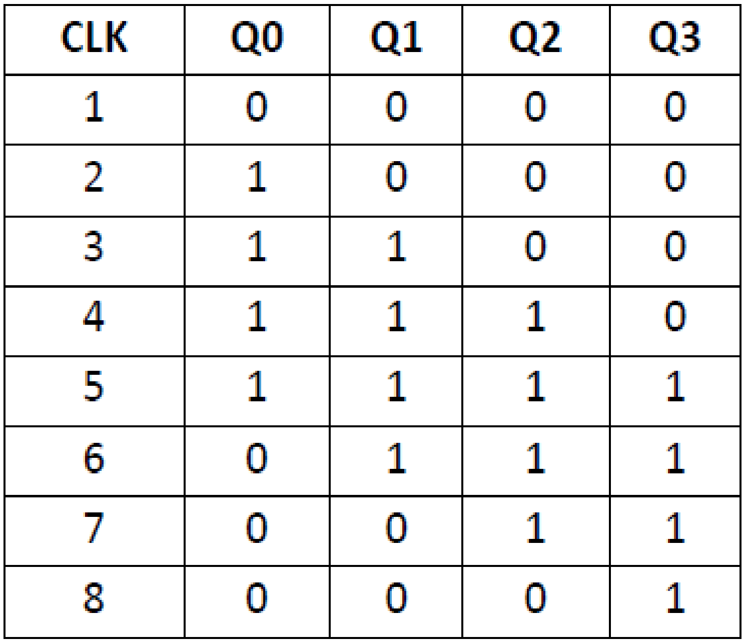
Truth tables:

Ring Counter:



Clock	QA	QB	QC	QD
0	0	0	0	0
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0

Johnson Counter:



CLK	Q0	Q1	Q2	Q3
1	0	0	0	0
2	1	0	0	0
3	1	1	0	0
4	1	1	1	0
5	1	1	1	1
6	0	1	1	1
7	0	0	1	1
8	0	0	0	1

Procedure:

Ring Counter:

- Connections are made as per the circuit diagram.
- Apply the data 1000 at flip-flop A, B, C and D respectively.
- In the beginning, clear all flip flops using $\overline{CLR}$ input by applying '0' to all clear inputs. After that, keep it at logic state '1'. In order to make Q_A high to begin, we should keep $\overline{PRE}$ of flip-flop A at "0" for a time period which is less than the clock duration. After that keep $\overline{PRE}$ = 1 for proper working. For this, keep the mode input M = 1, apply one clock pulse. (<16Hz).
- Now the mode M is made 0 and clock pulses are applied one by one and the truth table is verified.
- Observe the output and verify the truth table.

Johnson Counter:

- Connections are made as per the circuit diagram.
- In the beginning, clear all flip flops using CLR input.
- In order to make Q1 high to begin, we should keep $\overline{PRE}$ = 0 for a time period which is less than the clock duration. After that keep $\overline{PRE}$ = 1 for proper working
- Observe the output and verify the truth table.

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- Constructed Ring and Johnson counters and verified their truth tables.

Viva-Voce Questions:

1. List out various shift counters.
2. Distinguish Ring and Johnson counters

12. Conversion of Flip-flops (JK-D & JK-T)

Aim:

- To Convert JK flip-flop into T and D flip-flop
- To verify the truth tables of T and D flip-flops

Apparatus Required:

- Digital trainer board
- ICs- 7400,7410
- Connecting wires

Theory:

Flip-flops:

A flip-flop is a circuit with two stable states that can be used to store binary data. The stored data can be changed by applying varying inputs. Flip-flops and latches are fundamental building blocks of digital systems used in computers, communications, and many other types of systems. Both are used as data storage elements. The primary difference between a latch and a flip-flop is a gating or clocking mechanism. The difference between latch and flip flop is that a latch is level-triggered (outputs can change as soon as the inputs change) and Flip-Flop is edge-triggered (only changes state when a control signal goes from high to low or low to high).

JK Flip-flop:

The JK flip-flop augments the behavior of the SR flip-flop (J=Set, K=Reset) by interpreting the J = K = 1 condition as a "flip" or toggle command. Specifically, the combination J = 1, K = 0 is a command to set the flip-flop; the combination J = 0, K = 1 is a command to reset the flip-flop; and the combination J = K = 1 is a command to toggle the flip-flop, i.e., change its output to the logical complement of its current value. Setting J = K = 0 maintains the current state. The characteristic equation of the JK flip-flop is $Q_{n+1} = J\overline{Q}_n + \overline{K}Q_n$

JK Flip-flop into D and T Flip-flops:

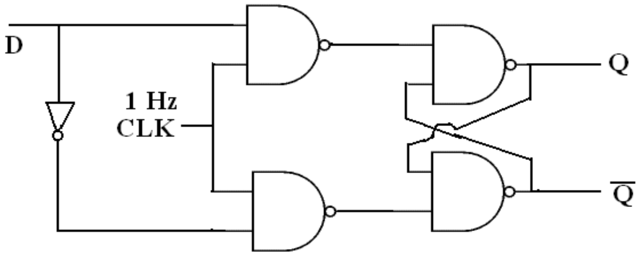
To synthesize a D flip-flop, simply set K equal to the complement of J. Similarly, to synthesize a T flip-flop, set K equal to J. The JK flip-flop is therefore a universal flip-flop, because it can be configured to work as an SR flip-flop, a D flip-flop, or a T flip-flop.

D Flip-flop:

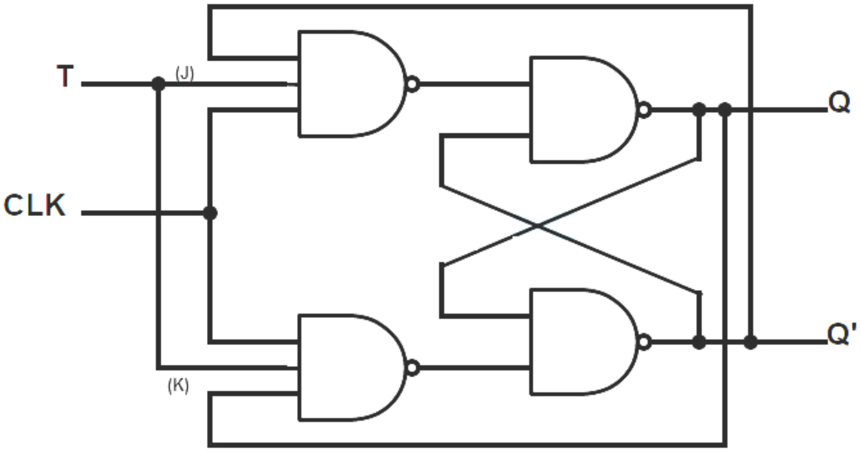
A D-type flip-flop is a clocked flip-flop which has two stable states. A D-type flip-flop operates with a delay in input by one clock cycle. Thus, by cascading many D-type flip-flops delay circuits can be created, which are used in many applications such as in digital television systems. Characteristic Equation is Q_{n+1}= D.

Circuit Diagrams:

D Flip-flop:



T Flip-flop:



Truth tables:

D Flip-flop:

D	Q_{n+1}
0	0
1	1

T Flip-flop:

T	Q_n	Q_{n+1}
0	0	0
0	1	1
1	0	1
1	1	0

Procedure:

- Connections are made as per the circuit diagram
- Switch on the power supply
- Apply different combinations of inputs and observe the out puts

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- T, D flip-flops are constructed by modifying JK flip-flop and their truth tables are verified

Viva-Voce Questions:

1. What is a Flip-flop?
2. What is a Latch?
3. Distinguish Flip-flop and Latch
4. What is a Characteristic equation of a Flip-flop?
5. What is meant by Race around condition in JK Flip-flop?
6. Give the characteristic equations for T and D flip-flops

13. Design of Decade Counter using Flip-flops

Aim:

- To construct a Decade counter
- To verify the truth tables of Decade counter

Apparatus Required:

- Digital trainer board
- ICs- 7476,7400
- Connecting wires

Theory:

Counter:

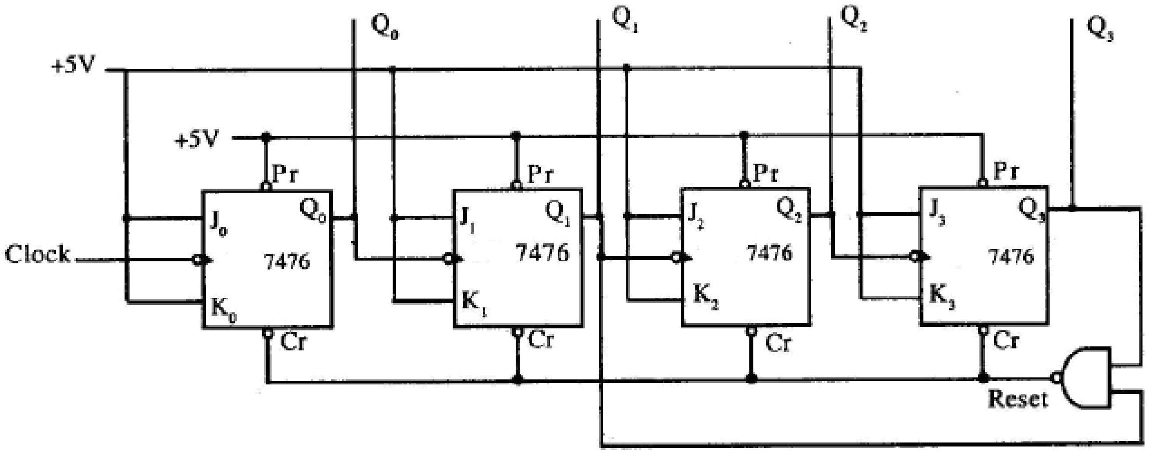
A counter is a device for recording the number of events, operations, or pulses that have occurred. The counter must keep the total and provide an indication of the total number, or count that has been handled. Counters may also be used for dividing frequencies, for addressing information in storage, or for temporary storage. Counters are a series of flip-flops wired together to perform the type of counting desired. The total number of counts or stable states a counter can indicate is called modulus. For instance, the modulus of a four-stage counter would be 16, since it is capable of indicating 0000 to 1111. The term modulo is used to describe the count capability of counters; that is, modulo-16 for a four-stage binary counter, modulo-10 for a decade counter, modulo-8 for a three-stage binary counter, and so forth.

Decade Counter:

A decade counter is a binary counter that is designed to count from 0 to 9 or 0000 to 1001. An ordinary four-stage counter can be easily modified to a decade counter by adding a NAND gate as shown in the figure below. Notice that FF2 and FF4 provide the inputs to the NAND gate. The NAND gate outputs are connected to the CLR input of each of the flip-flops. The counter operates as a normal counter until it reaches a count of 1010 or 10. At that time, both inputs to the NAND gate are HIGH, and the output goes LOW. This LOW applied to the CLR input of the flip-flops causes them to reset to 0. Once the flip-flops are reset, the count may begin again.

Circuit Diagram:

Decade Counter



Truth table:

Decade Counter:

Clock Pulse	Q_3	Q_2	Q_1	Q_0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	0	0	0	0

Procedure:

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

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- Constructed Decade Ripple counter and verified its truth table.

Viva-Voce Questions:

1. What is a counter?
2. Distinguish Asynchronous &Synchronous counters
3. What is Modulus of a counter?
4. Determine the number of Flip-flops required to design a Mod-12 counter
5. What is an up counter?
6. What is a down counter?
7.Compare binary and non-binary counters.

14. Design of Asynchronous Counter using Flip-flops

Aim:

- To construct an Asynchronous counter
- To verify the truth table of constructed counter

Apparatus Required:

- Digital trainer board
- ICs- 7476
- Connecting wires

Theory:

Counter:

A counter is a device for recording the number of events, operations, or pulses that have occurred. The counter must keep the total and provide an indication of the total number, or count that has been handled. Counters may also be used for dividing frequencies, for addressing information in storage, or for temporary storage. Counters are a series of flip-flops wired together to perform the type of counting desired. The total number of counts or stable states a counter can indicate is called modulus. For instance, the modulus of a four-stage counter would be 16, since it is capable of indicating 0000 to 1111. The term modulo is used to describe the count capability of counters; that is, modulo-16 for a four-stage binary counter, modulo-10 for a decade counter, modulo-8 for a three-stage binary counter, and so forth.

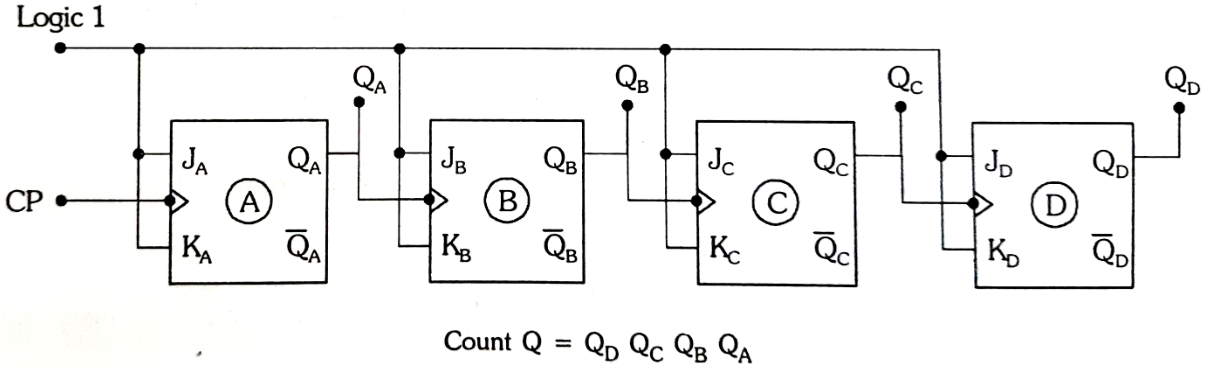
Asynchronous Counter:

These are the counters in which we do not use universal clock, main clock is only applied to the first flip flop and then for rest of flip flops the output of previous flip flop is taken as a clock. These are also called ripple counters. Ripple counter is a cascaded arrangement of flip-flops where the output of one flip-flop drives the clock input of the following flip-flop. The number of flip flops in the cascaded arrangement depends upon the number of different logic states that it goes through before it repeats the sequence a parameter known as the modulus of the counter. An n-bit ripple counter can count up to 2^n states. It is known as ripple counter because of the way the clock pulse ripples its way through the flip-flops. Some of the features of ripple counter are:

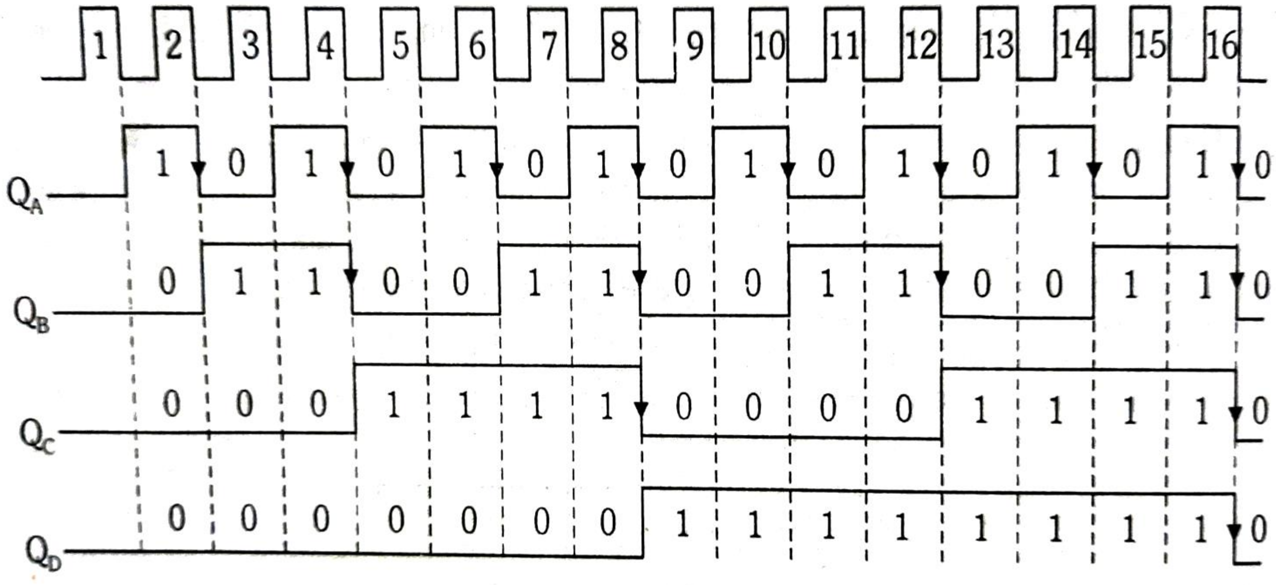
- Different flip-flops are used with a different clock pulse.
- All the flip-flops are used in toggle mode.
- Only one flip-flop is applied with an external clock pulse and another flip-flop clock is obtained from the output of the previous flip-flop.
- The flip-flop applied with an external clock pulse act as LSB (Least Significant Bit) in the counting sequence.

Circuit Diagram:

4-bit Ripple Counter



Timing diagram:


1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Q_A 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
Q_B 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0
Q_C 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0
Q_D 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0

Truth table:

4-bit Ripple Counter:

Clock Pulse	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0

Procedure:

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

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- Constructed 4-bit Ripple counter and verified its truth table.

Viva-Voce Questions:

1. What is a counter?
2. Distinguish Asynchronous &Synchronous counters
3. What is Modulus of a counter?
4. Determine the number of Flip-flops required to design a Mod-12 counter
5. What is an up counter?
6. What is a down counter?
7.Compare binary and non-binary counters.

15. Design of Synchronous Counter using Flip-flops

Aim:

- To construct a 4-bit Synchronous counter
- To verify the truth tables of 4-bit Synchronous counter

Apparatus Required:

- Digital trainer board
- ICs- 7476,7400
- Connecting wires

Theory:

Counter:

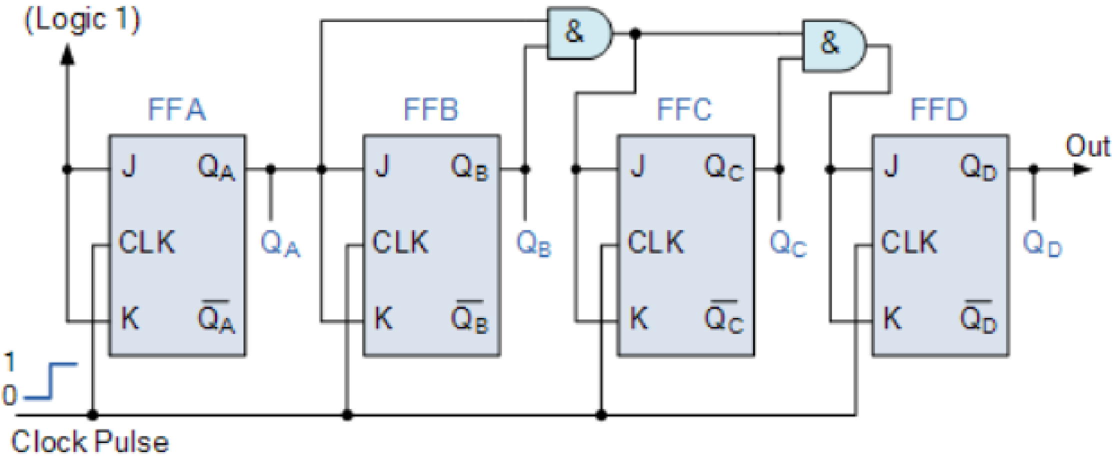
A counter is a device for recording the number of events, operations, or pulses that have occurred. The counter must keep the total and provide an indication of the total number, or count that has been handled. Counters may also be used for dividing frequencies, for addressing information in storage, or for temporary storage. Counters are a series of flip-flops wired together to perform the type of counting desired. The total number of counts or stable states a counter can indicate is called modulus. For instance, the modulus of a four-stage counter would be 16, since it is capable of indicating 0000 to 1111. The term modulo is used to describe the count capability of counters; that is, modulo-16 for a four-stage binary counter, modulo-10 for a decade counter, modulo-8 for a three-stage binary counter, and so forth.

Synchronous Counter:

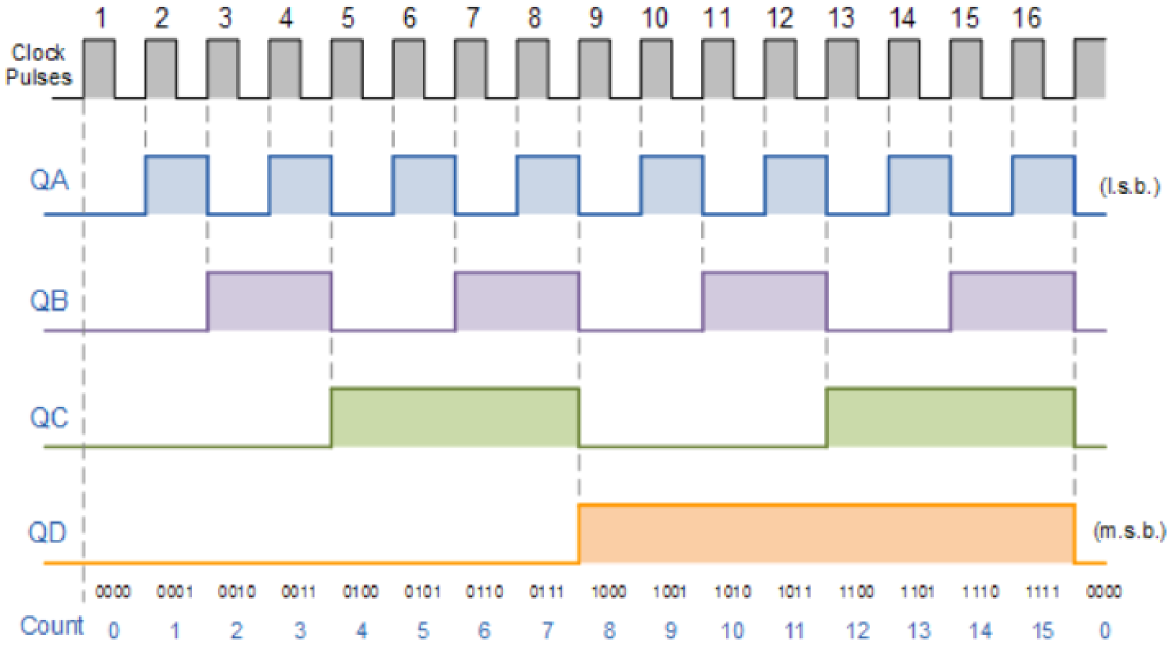
Unlike the asynchronous counter, synchronous counter has one global clock which drives each flip flop so output changes in parallel. The one advantage of synchronous counter over asynchronous counter is, it can operate on higher frequency than asynchronous counter as it does not have cumulative delay because of same clock is given to each flip flop. It is also called as parallel counter.

Circuit Diagram:

4-bit Synchronous Counter



Timing Diagram:



Truth table:

4-Bit Synchronous Counter:

Clock Pulse	Q_D	Q_C	Q_B	Q_A
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1
16	0	0	0	0

Procedure:

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

Precautions:

- Check the connections before giving the power supply.
- Place the ICs on the board properly

Result:

- Constructed 4-bit synchronous counter and verified its truth table.

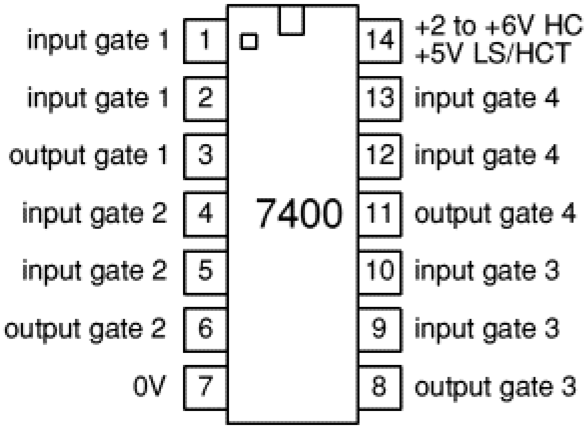
Viva-Voce Questions:

1. What is a counter?
2. Distinguish Asynchronous &Synchronous counters
3. What is Modulus of a counter?
4. Determine the number of Flip-flops required to design a Mod-12 counter
5. What is an up counter?
6. What is a down counter?

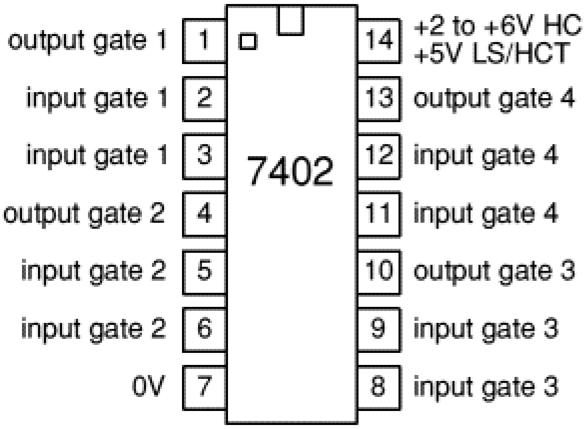
APPENDIX

Useful IC Pin details

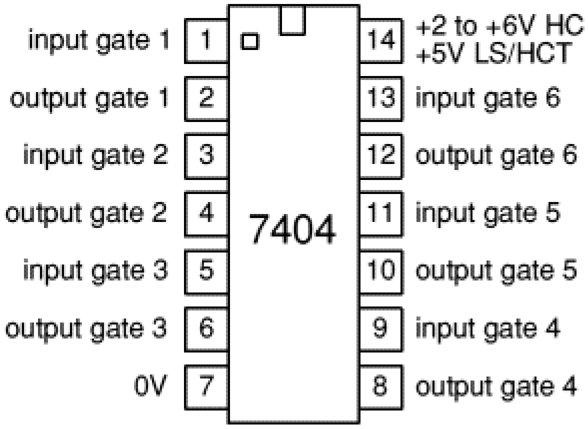
- **7400(NAND)**



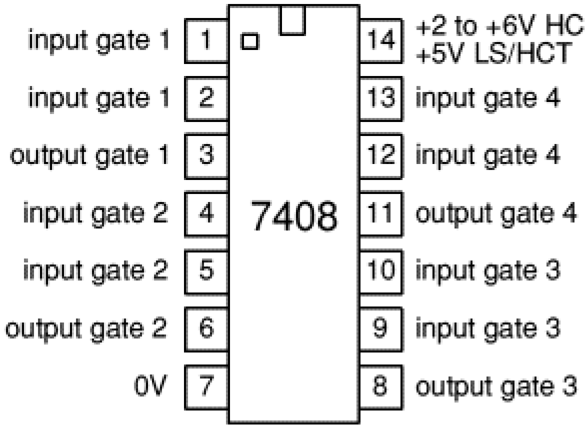
- **7402(NOR)**



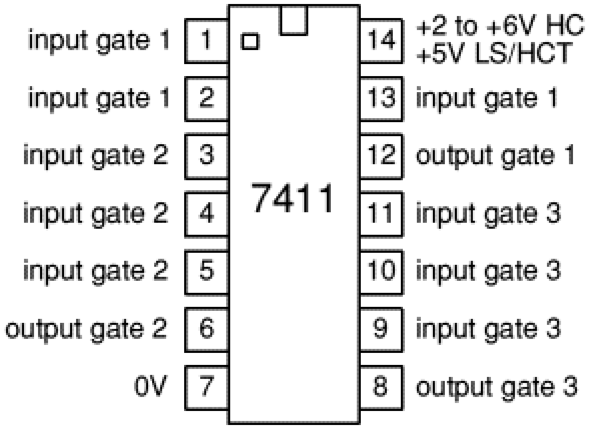
- **7404(NOT)**



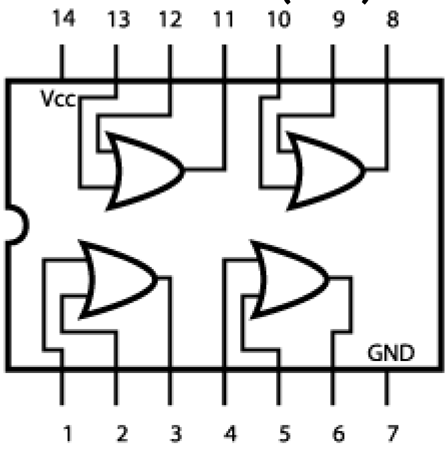
- **7408(AND)**



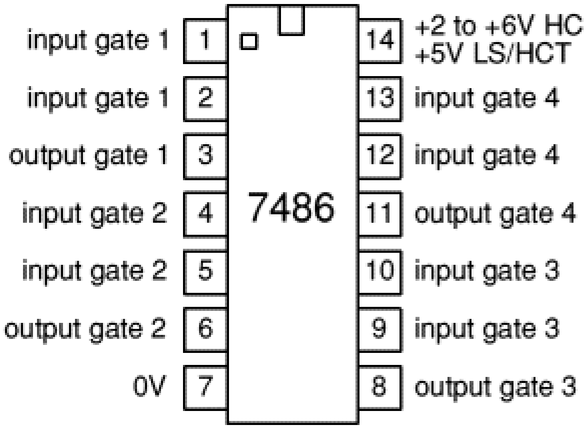
- **7411(3-i/p AND)**



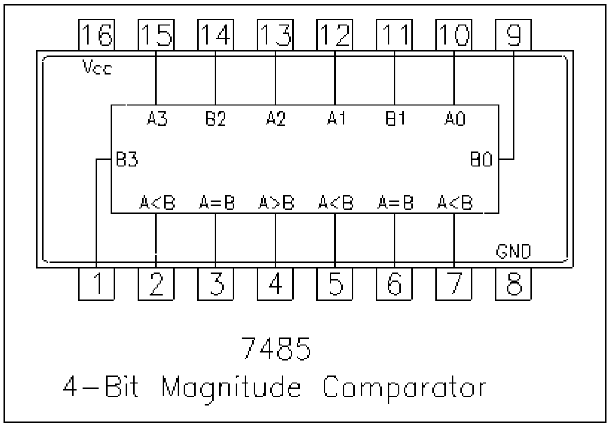
- **7432(OR)**



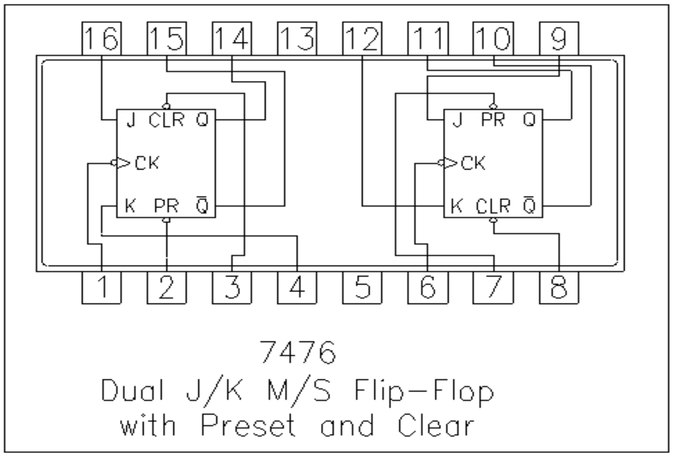
- **7486(EX-OR)**



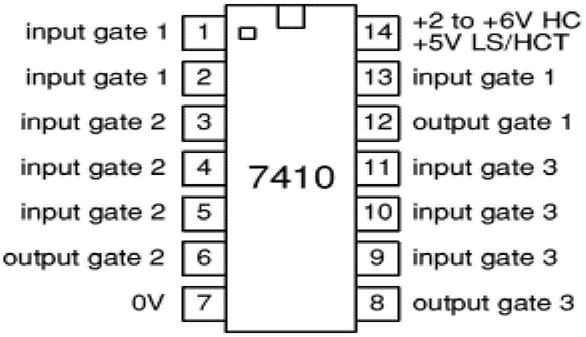
- **7485(Comparator)**



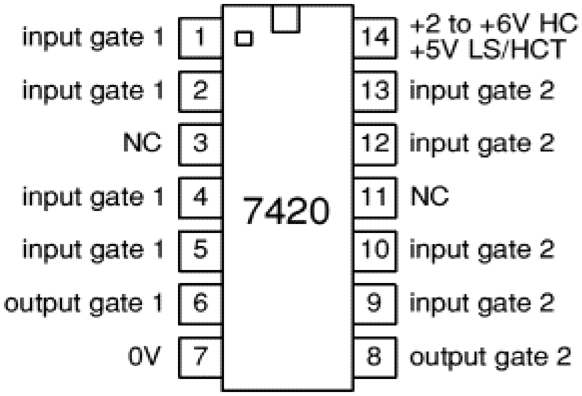
- **7476(JK Flip-flop)**



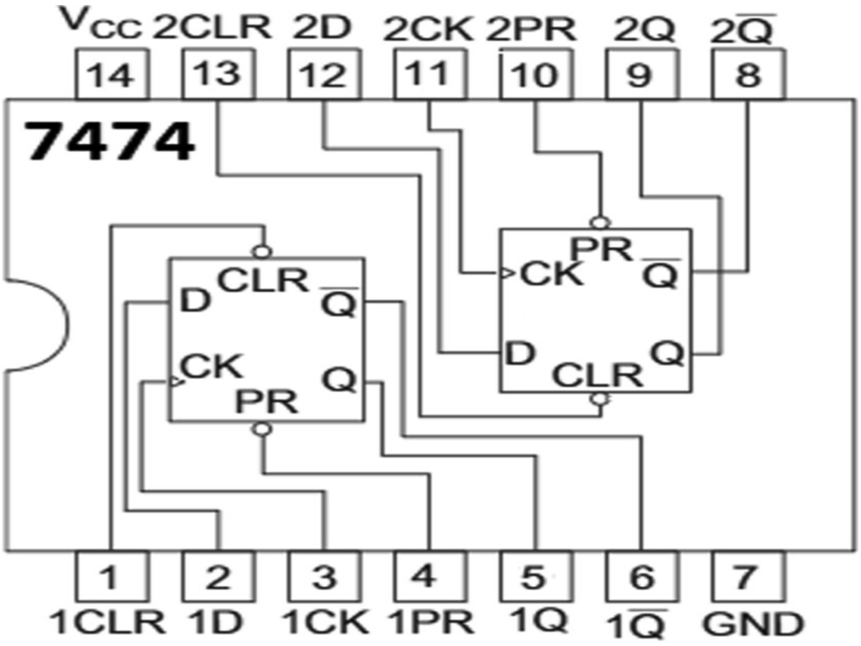
- **7410(3-i/p NAND)**



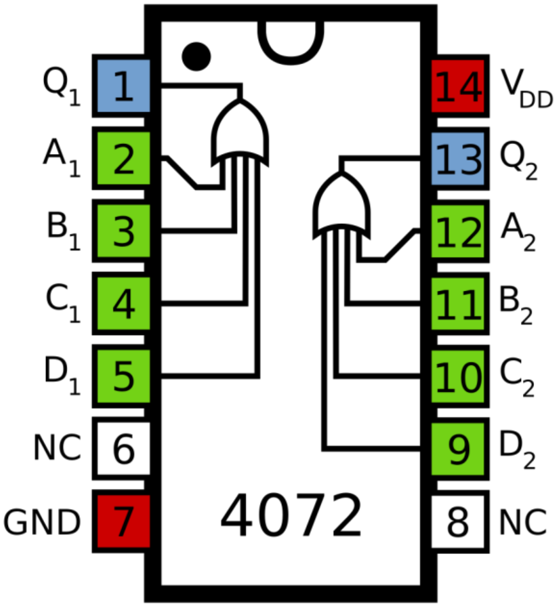
- **7420(4-i/p NAND)**



- **7474(D Flip-flop))**


Vcc 2CLR 2D 2CK 2PR 2Q 2$\overline{Q}$
14
13
12
11
10
9
8
7474
PR
CK
$\overline{Q}$
D
CLR
Q
CLR
D
$\overline{Q}$
CK
Q
PR
1
2
3
4
5
6
7
1CLR 1D 1CK 1PR 1Q 1$\overline{Q}$ GND

- 4072 (4-i/p OR)


Q1
1
A1
2
B1
3
C1
4
D1
5
NC
6
GND
7
14
VDD
13
Q2
12
A2
11
B2
10
C2
9
D2
8
NC
4072