SİVAS UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF ENGINEERING AND NATURAL SCIENCES

DIGITAL SYSTEM DESIGN

EXPERIMENTS MANUAL REPORT

Supervision : **Asst. Prof. Nurbanu GÜZEY**
Prepared by : **Research Asst. Hatice AKTAŞ AYDIN**
Research Asst. İlhan ERDOĞAN
Research Asst. İremnur DURU

2025

SİVAS

TABLE OF CONTENT

Titles	Page Number
Experiment 1: Logic Gates and Verify Their Truth Tables	
Experiment 2: Half/Full Adder and Half/Full Subtractor	
Experiment 3: IC7483 as a Parallel Adder / Subtractor.	
Experiment 4: BCD to Seven-Segment Display	
Experiment 5: Multiplexers (MUX) and Demultiplexers (DEMUX)	
Experiment 6: Comparators	
Experiment 7: Flip-Flops	
Experiment 8: 3-Bit Counters	

SİVAS BİLİM VE TEKNOLOJİ ÜNİVERSİTESİ

Electrical- Electronics Engineering

Computer Engineering

Digital System Design

Experiment-1

Prepared By:

Research Asst. Hatice AKTAŞ AYDIN

Research Asst. İlhan ERDOĞAN

Research Asst. İremnur DURU

2025

Name-Surname	Number	Sign.

EXPERIMENT-1

Objective-1: To study about logic gates and verify their truth tables.

Apparatus Required: 7400, 7402, 7404, 7408, 7410, 7432, 7486.

Theoretical Background:

Circuit that takes the logical decision and the process are called logic gates. Each gate has one or more input and only one output. OR, AND & NOT are basic gates. NAND, NOR and X-OR are known as universal gates.

The AND gate performs a logical multiplication commonly known as AND function. The output is high when both the inputs are high. The output is low level when any one of the inputs is low.

The OR gate performs a logical addition commonly known as OR function. The output is high when any one of the inputs is high. The output is low level when both the inputs are low.

The NOT gate is called an inverter. The output is high when the input is low. The output is low when the input is high.

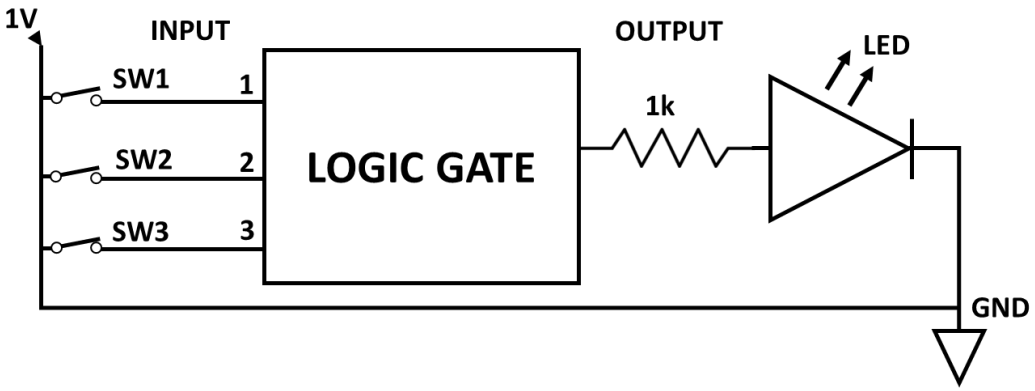
The NAND gate is a contraction of AND-NOT. The output is high when both inputs are low and any one of the inputs is low. The output is low level when both inputs are high.

The NOR gate is a contraction of OR-NOT. The output is high when both inputs are low. The output is low when one or both inputs are high.

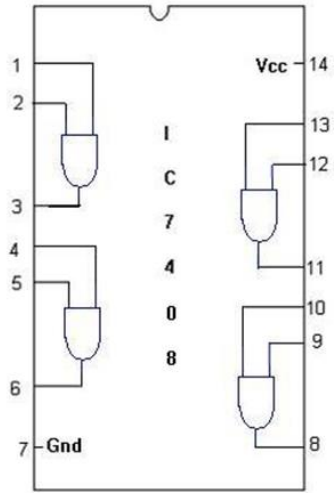
The XOR gate output is high when any one of the inputs is high. The output is low when both the inputs are low, and both the inputs are high.

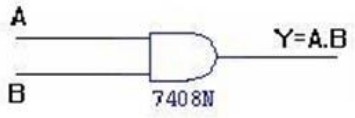
Prelab:

1- **Connections and Logical inputs are given as per circuit diagram. Observe the outputs and verify the truth tables on simulating the Proteus software** ***(The number of inputs in the circuit varies according to the logic gate. For example, the number of inputs for the AND gate is 2. The number of inputs for the NOT gate is 1).***



A. AND GATE

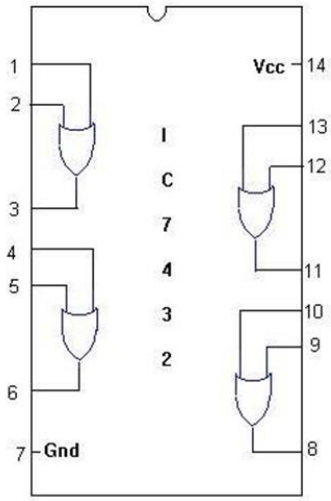


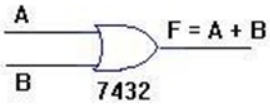


A	B	A.B
0	0	
0	1	
1	0	
1	1	

B. OR GATE

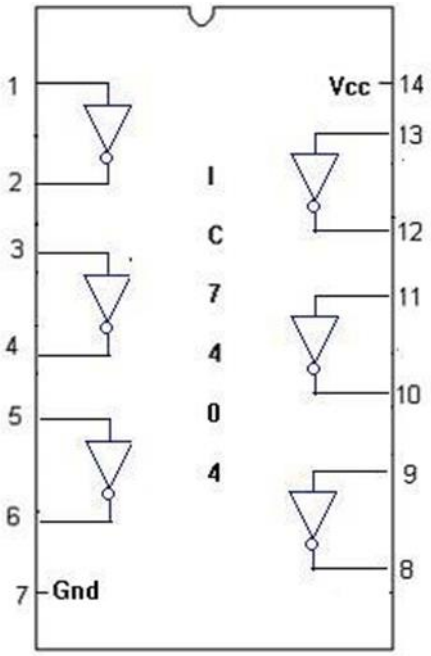
PIN DIAGRAM :

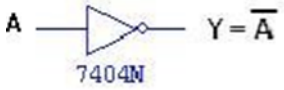




A	B	A+B
0	0	
0	1	
1	0	
1	1	

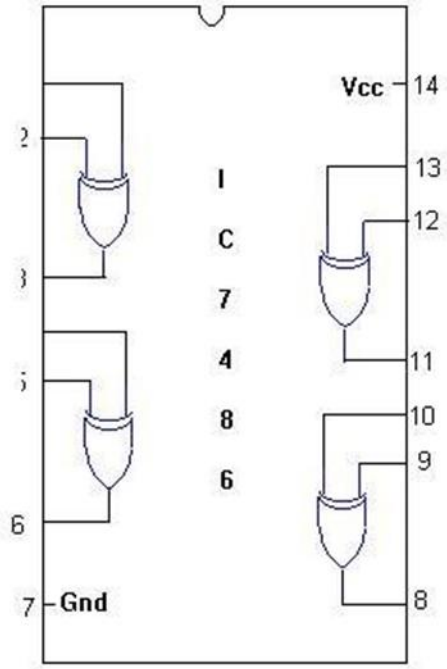
C. NOT GATE

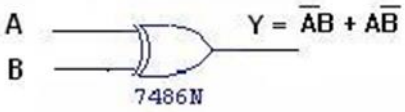




A	$\bar{A}$
0	
1	

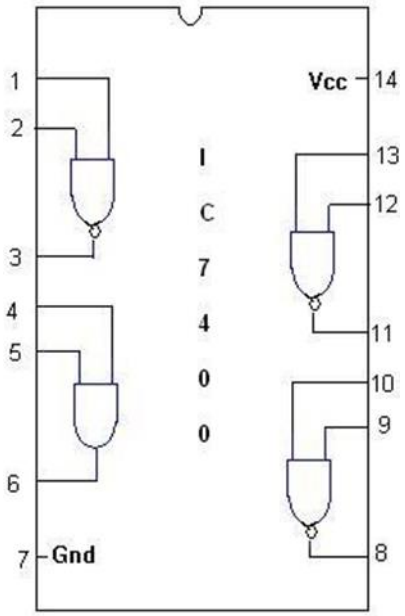
D. XOR GATE

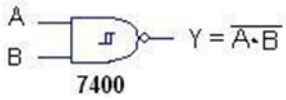




A	B	$\bar{A}B+A\bar{B}$
0	0	
0	1	
1	0	
1	1	

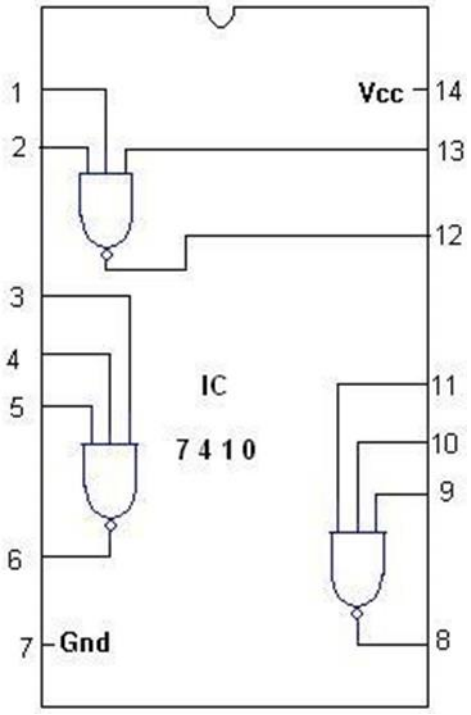
E. 2 & 3 INPUTS NAND GATES



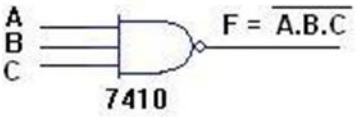


A	B	$\overline{A.B}$
0	0	
0	1	
1	0	
1	1	

PIN DIAGRAM :



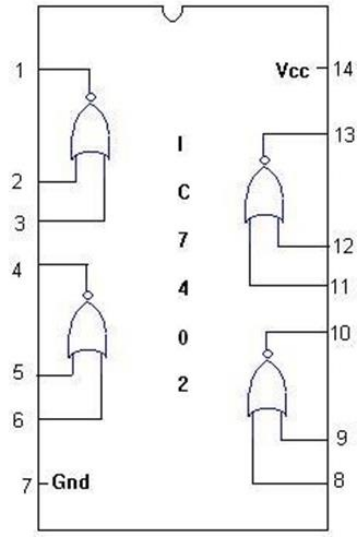
SYMBOL :

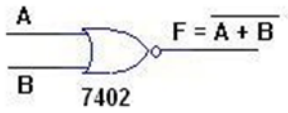


A	B	C	$\overline{A.B.C}$
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

F. NOR GATE

PIN DIAGRAM :





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

Objective-2: To verify the Boolean Theorems using logic gates.

Theoretical Background: The basic operations of Boolean algebra are as follows:

Commutative Law: The binary operator OR, AND is said to be commutative if,

A+B = B+A	A.B=B.A

Associative Law: The binary operator OR, AND is said to be associative if,

A+(B+C) = (A+B)+C	A.(B.C) = (A.B).C

Distributive Law: The binary operator OR, AND is said to be distributive if,

A+(B.C) = (A+B).(A+C)	A.(B+C) = (A.B)+(A.C)

Absorption Law:

A+AB = A	A.(A+B) = A

Involution or Double Complement Law

A = A

Idempotent Law:

A+A = A	A.A = A

Complementary Law:

A+A' = 1	A.A' = 0

De Morgan's Theorem: The complement of the sum is equal to the sum of the product of the individual complements.

$\overline{A + B} = \overline{A}.\overline{B}$

The complement of the product is equal to the sum of the individual complements.

$\overline{A.B} = \overline{A} + \overline{B}$

We will also use the following set of postulates to prove all other theorems in Boolean algebra.

P1: Boolean algebra is closed under the AND, OR, and NOT operations.

P2: The identity element with respect to AND (•) is one (1) and OR (+) is zero (0). There is no identity element with respect to logical NOT.

P3: The AND (•) and OR (+) operators are commutative.

P4: AND (•) & OR (+) operators are distributive with respect to one another.

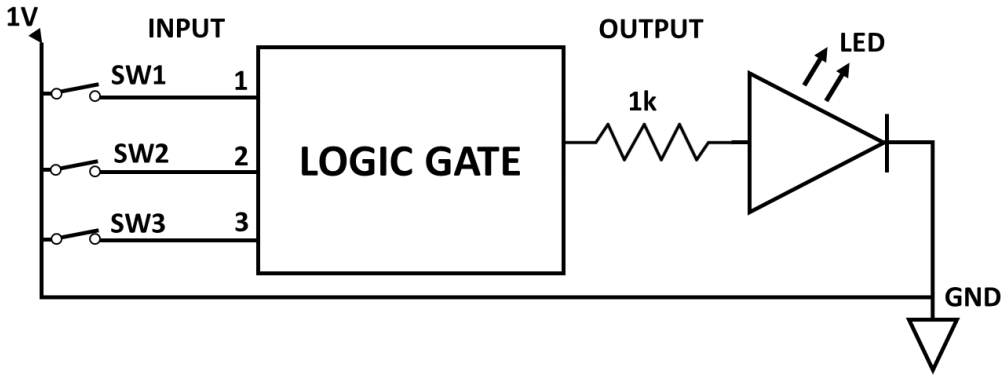
P5: For every value A there exists a value A' such that A•A' = 0 and A+A' = 1. This value is the logical complement (or NOT) of A.

P6: AND (•) & OR (+) are both associative.

Experimental Procedure for Lab.:

1- Connections and Logical inputs are given as per circuit diagram. Observe the outputs and verify the truth tables. (7400, 7402, 7404, 7408, 7410, 7432, 7486)

***(The number of inputs in the circuit varies according to the logic gate. For example, the number of inputs for the AND gate is 2. The number of inputs for the NOT gate is 1).*



AND-7408

A	B	A.B
0	0	
0	1	
1	0	
1	1	

OR-7432

A	B	A+B
0	0	
0	1	
1	0	
1	1	

NOT-7404

A	$\bar{A}$
0	
1	

XOR-7486

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

2 & 3 INPUTS NAND GATES

A	B	$\overline{A.B}$
0	0	
0	1	

Triple 3-input NAND gate-7410

A	B	C	$\overline{A.B.C}$
0	0	0	
0	0	1	
0	1	0	

1	0	
1	1	

0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

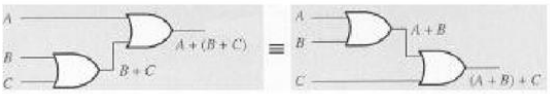
NOR-7402

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

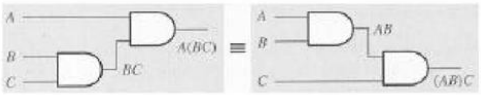
A. ASSOCIATIVE LAW OF BOOLEAN ALGEBRA

Associative Laws of Boolean Algebra

$$A + (B + C) = (A + B) + C$$



$$A \bullet (B \bullet C) = (A \bullet B) \bullet C$$



Proof of the Associative Property for the OR operation: (A+B)+C=A+(B+C)

A	B	C	A+B	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				

Proof of the Associative Property for the AND operation: (A.B).C=A.(B.C)

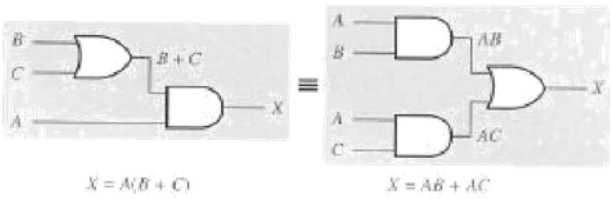
A	B	C	A.B	B.C	A.(B.C)	(A.B).C
0	0	0				
0	0	1				
0	1	0				
0	1	1				
1	0	0				
1	0	1				
1	1	0				
1	1	1				

B. DISTRIBUTIVE LAW OF BOOLEAN ALGEBRA

Distributive Laws of Boolean Algebra

$$A \bullet (B + C) = A \bullet B + A \bullet C$$

$$A\,(B + C) = A\,B + A\,C$$



Proof of the distributive rule

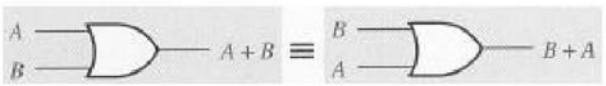
A	B	C	A.B	A.C	(A.B)+(A.C)	B+C	A.(B+C)
0	0	0					
0	0	1					
0	1	0					
0	1	1					
1	0	0					
1	0	1					
1	1	0					
1	1	1					

A	B	C	A+B	A+C	(A+B). (A+C)	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					

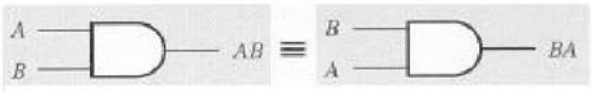
C. COMMUTATIVE LAW OF BOOLEAN ALGEBRA

Commutative Laws of Boolean Algebra

$$A + B = B + A$$



$$A \bullet B = B \bullet A$$

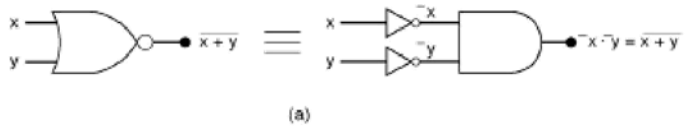


Proof commutative law of Boolean algebra via completing the below given truth table:

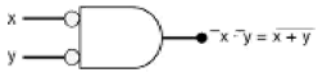
A	B	A+B	B+A	A.B	B.A

D. DE MORGAN'S THEOREM

Construct the two circuits corresponding to the functions A'. B'and (A+B)' respectively. Show that for all combinations of A and B, the two circuits give identical results. Connect these circuits and verify their operations.



(a)

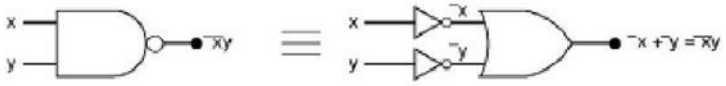


(b)

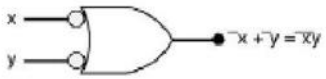
Proof (via Truth Table) of DeMorgan's Theorem $\overline{A \cdot B} = \overline{A} + \overline{B}$

A	B	A.B	$\overline{A.B}$	$\overline{A}$	$\overline{B}$	$\overline{A} + \overline{B}$
0	0					
0	1					
1	0					
1	1					

Construct two circuits corresponding to the functions A'+B'and (A.B)' A.B, respectively. Show that, for all combinations of A and B, the two circuits give identical results. Connect these circuits and verify their operations.



(a)



(b)

Proof (via Truth Table) of DeMorgan's Theorem $\overline{A + B} = \overline{A} \cdot \overline{B}$

A	B	A+B	$\overline{A + B}$	$\overline{A}$	$\overline{B}$	$\overline{A}.\overline{B}$
0	0					
0	1					
1	0					
1	1					

Report: Please send the following outputs to the relevant research assistants via e-mail (including Proteus outputs)

1- Simulating the assignments in the Prelab section in the Proteus software programme and processing the outputs in the truth table

2- Establishing the circuits in the Procedure section in the laboratory and filling the relevant tables

Electrical- Electronics Engineering

Computer Engineering

Digital System Design

Experiment-2

Prepared By:

Research Asst. Hatice AKTAŞ AYDIN

Research Asst. İlhan ERDOĞAN

Research Asst. İremnur DURU

2025

EXPERIMENT-2

Objective-1: To realize half/full adder and half/full subtractor. Using only X-OR gates and using only NAND gates.

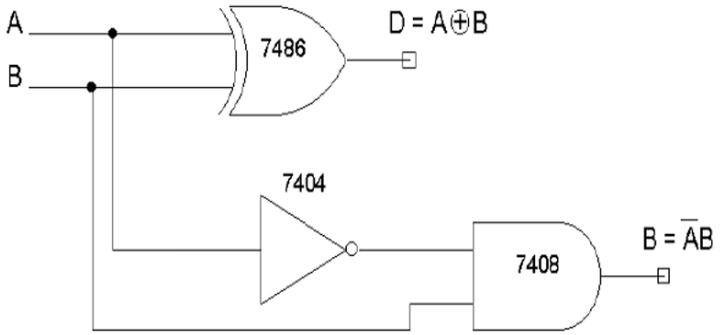
Apparatus Required: 7400, 7408, 7432, 7486, etc.

Prelab:

Simulate the "A", "B" and "C"section's electronic circuits in this section using Proteus software, verify their truth tables, and report the results.

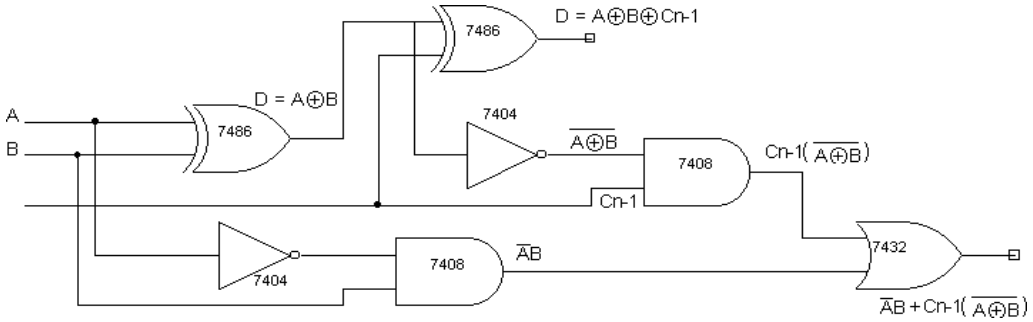
A. HALF SUBTRACTOR WITH USING ONLY X-OR AND BASIC GATES

D(v) and B(v) means verified datas.



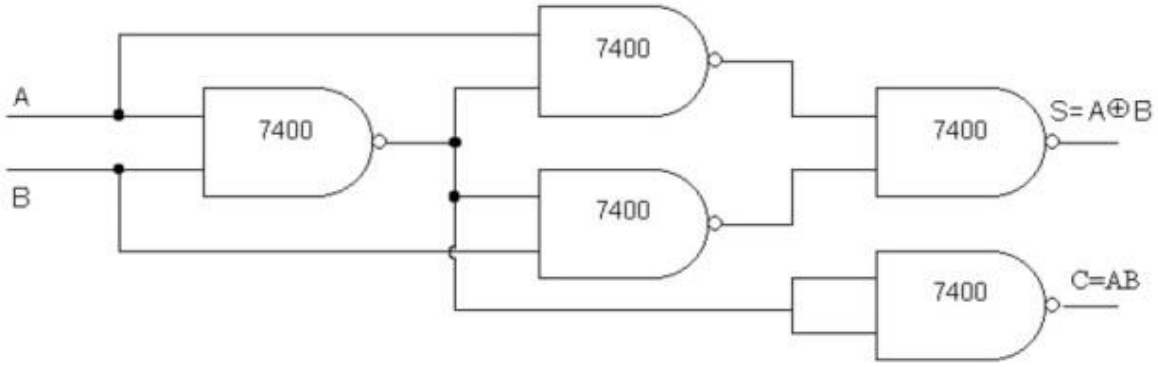
Half Subtractor					
A	**B**	**D**	**B**	**D(v)**	**B(v)**
0	**0**	**0**	**0**		
0	**1**	**1**	**1**		
1	**0**	**1**	**0**		
1	**1**	**0**	**0**		

B. FULL SUBTRACTOR WITH USING ONLY X-OR AND BASIC GATES



Full Subtractor						
A	**B**	**C_{n-1}**	**D**	**B**	**D(v)**	**B(v)**
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**		

C. HALF ADDER USING NAND GATES



A	B	S(sum)	C(carry)	S(V)	C(V)
0	**0**	0	0		
0	**1**	1	0		
1	**0**	1	0		
1	**1**	0	1		

Name-Surname	Number	Sign.

Experimental Procedure for Lab.:

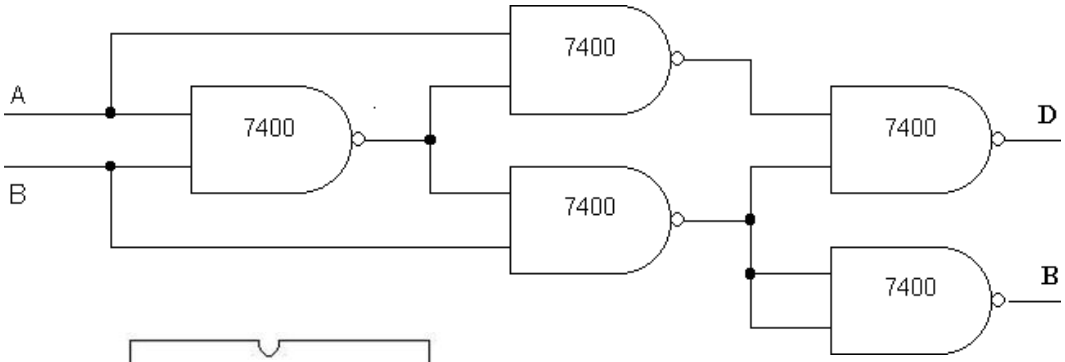
Procedure:

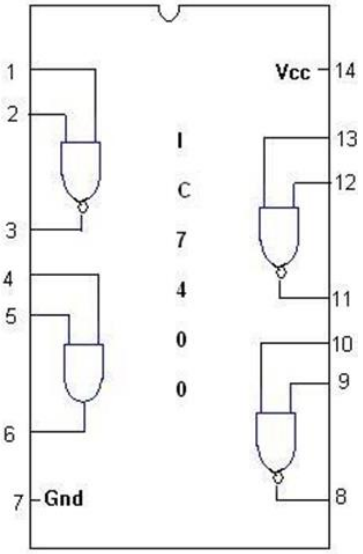
2- Connections and Logical inputs are given as per the circuit diagram. Observe the outputs and verify the truth tables. *D(v), B(v), S(v), and C(v) means verified datas.*

3- Switch on V_{CC} (5V) and apply various combinations of input according to the truth table.

4- Note down the output readings for half/full adder and half/full subtractor sum/difference and the carry/borrow bit for different combinations of inputs.

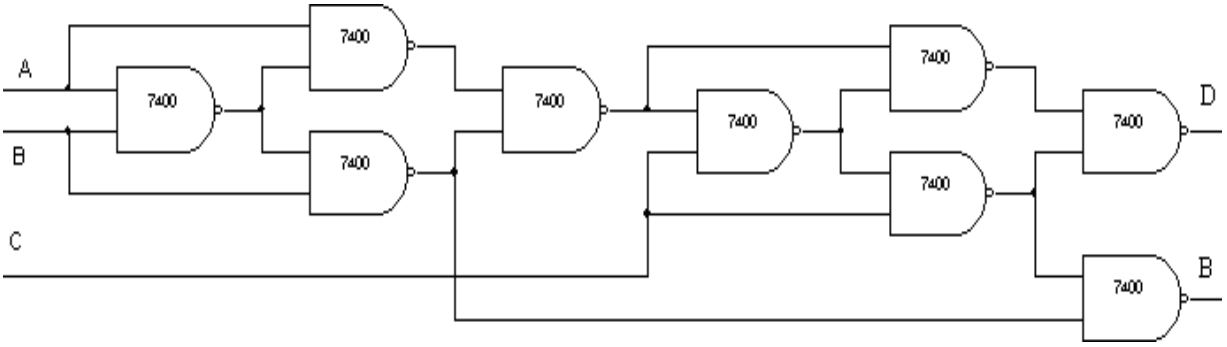
A. HALF SUBTRACTOR WITH USING ONLY NAND GATES





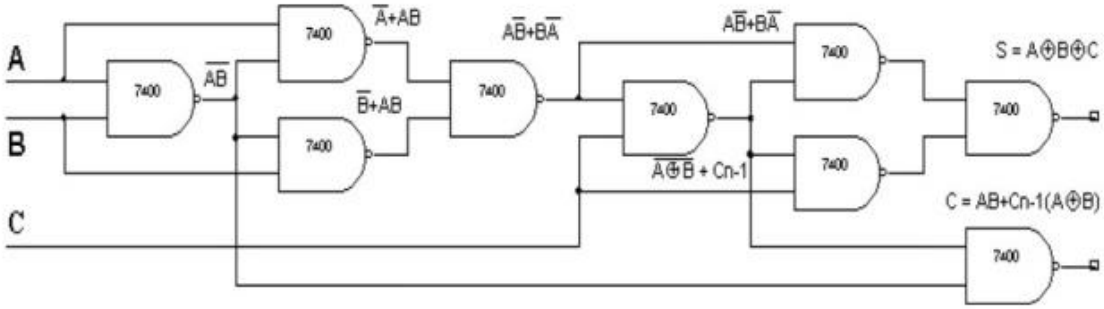
Half Subtractor					
A	**B**	**D**	**B**	**D(V)**	**B(V)**
0	**0**	**0**	**0**		
0	**1**	**1**	**1**		
1	**0**	**1**	**0**		
1	**1**	**0**	**0**		

B. FULL SUBSTRACTOR WITH USING ONLY NAND GATES



Full Subtractor						
A	**B**	**C_{n-1}**	**D**	**B**	**D(v)**	**B(v)**
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**		

C. FULL ADDER USING NAND GATES



A	B	C	S	C	S(v)	C(v)
0	**0**	**0**	**0**	**0**		
0	**0**	**1**	**1**	**0**		
0	**1**	**0**	**1**	**0**		
0	**1**	**1**	**0**	**1**		
1	**0**	**0**	**1**	**1**		
1	**0**	**1**	**0**	**1**		
1	**1**	**0**	**0**	**1**		
1	**1**	**1**	**1**	**1**		

Electrical- Electronics Engineering

Computer Engineering

Digital System Design

Experiment-3

Prepared By:

Research Asst. Hatice AKTAŞ AYDIN

Research Asst. İlhan ERDOĞAN

Research Asst. İremnur DURU

2025

Name-Surname	Number	Sign.

EXPERIMENT-3

Objective-1: To realize IC7483 as a parallel adder / Subtractor.

Apparatus Required: 7404, 7483, etc.

Procedure:

1. Apply the inputs to A0 to A3 and B0 to B3.

2. Connect C0 to the Ground.

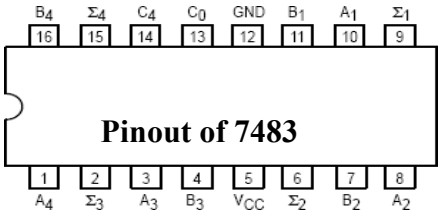
3. Check the output sum on the S0 to S3 and C4.

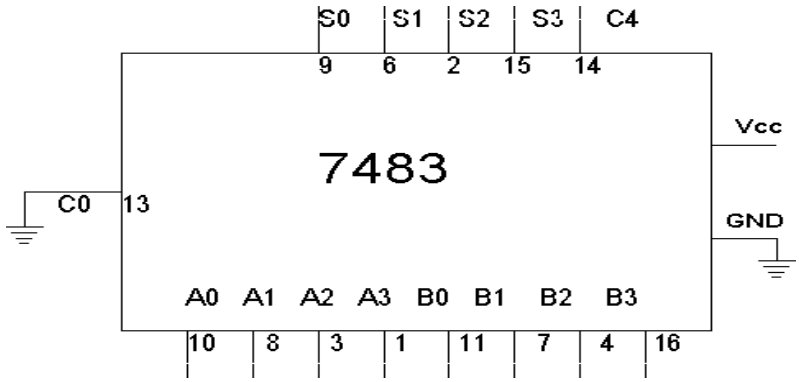
4. For subtraction connect C0 to Vcc, Apply the B input through NOT gate, which gives the complement of B.

5. The truth table of adder and subtractor is noted down.

Experimental Procedure for Lab and Prelab

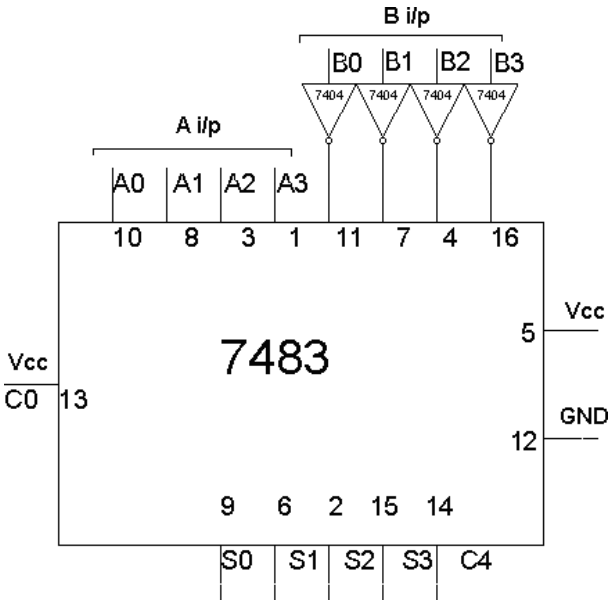
A. ADDER





A3	A2	A1	A0	B3	B2	B1	B0	C4 (V)	S3(V)	S2(V)	S1(V)	S0(V)
0	0	0	1	0	0	1	0					
0	1	0	1	1	0	1	1					
1	0	1	0	1	0	1	0					
1	1	1	1	1	1	1	1					
0	1	1	1	0	0	1	1					

B. SUBTRACTOR



A3	A2	A1	A0	B3	B2	B1	B0	C4 (V)	S3(V)	S2(V)	S1(V)	S0(V)
0	**0**	**1**	**0**	**0**	**0**	**0**	**1**					
0	**1**	**0**	**1**	**0**	**0**	**1**	**1**					
0	**0**	**1**	**1**	**0**	**1**	**0**	**1**					
1	**0**	**1**	**0**	**0**	**1**	**1**	**0**					
1	**0**	**0**	**0**	**1**	**1**	**1**	**1**					

Report: Please send the following outputs to the relevant research assistants via e-mail (including Proteus outputs)

1- Simulating the assignments in the Prelab section in the Proteus software programme and processing the outputs in the truth table

2- Establishing the circuits in the Procedure section in the laboratory and filling the relevant tables

Electrical- Electronics Engineering

Computer Engineering

Digital System Design

Experiment-4

Prepared By:

Research Asst. Hatice AKTAŞ AYDIN

Research Asst. İlhan ERDOĞAN

Research Asst. İremnur DURU

2025

Name-Surname	Number	Sign.

EXPERIMENT-4

Objective-1: BCD to Seven-Segment Display

Apparatus Required: 7447, 7483, etc.

Procedure for Prelab:

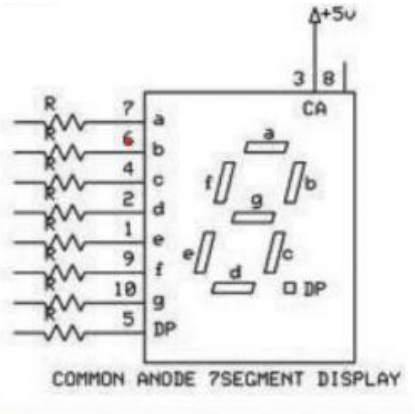
A. BCD to SEVEN SEGMENT WITH LOGIC GATES

First design a combinational circuit, which would implement the decoder function for only the segment "a", of the display. This can be done in the following steps:

a) Write down the truth table with 4 inputs and 7 outputs (Table).

b) For only the output "a", obtain a minimum logic function. Realize this function using NAND gates and inverters only. For example, if decimal 9 is to be displayed a, b, c, d, f, g must be 0 and the others must be 1 (For common anode type display units), if decimal 5 is to be displayed then a, f, g, c, d must be 0 and the others must be 1.

c) Connect the output "a" of your circuit to the appropriate input of the 7-segment display unit. By applying BCD codes verify the displayed decimal digits for that segment for "a" of the display.

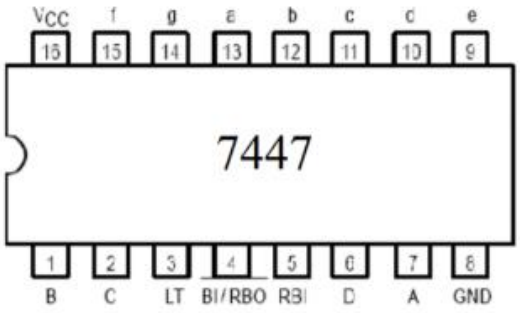


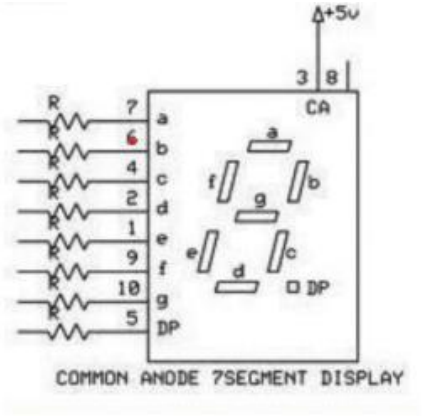
Dec.	BCD				Outputs						
	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0							
1	0	0	0	1							
2	0	0	0	0							
3	0	0	1	1							
4	0	1	0	0							
5	0	1	0	1							
6	0	1	1	0							
7	0	1	1	1							
8	1	0	0	0							
9	1	0	0	1							

Procedure for Lab Experiment-1:

BCD to SEVEN SEGMENT WITH DECODER

Replace your circuit with a decoder IC 7447 for all of the seven segments. Observe the display and record the segments that will light up for invalid input sequences. Comment on the design if you don't want to see any digit for an invalid input sequence.





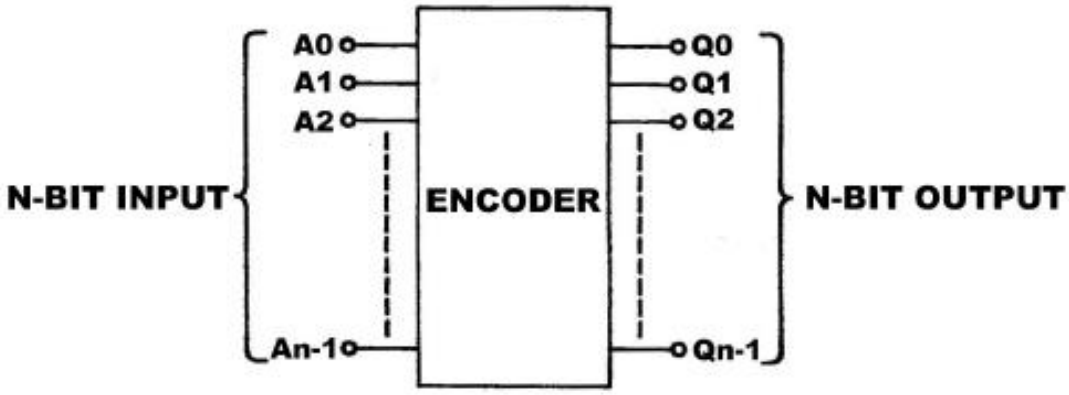
Dec.	BCD				Outputs						
	A	B	C	D	a	b	c	d	e	f	g
0	0	0	0	0							
1	0	0	0	1							
2	0	0	0	0							
3	0	0	1	1							
4	0	1	0	0							
5	0	1	0	1							
6	0	1	1	0							
7	0	1	1	1							
8	1	0	0	0							
9	1	0	0	1							

Objective-2: Understanding the working principles of encoders.

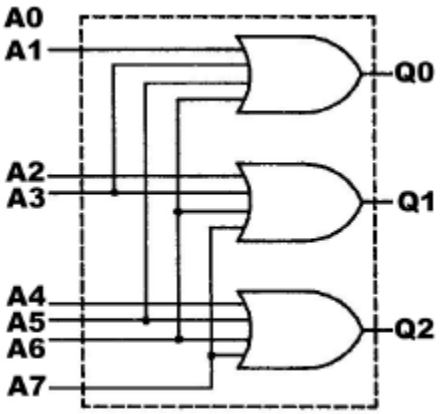
Apparatus Required: 74LS148

Theoretical Background:

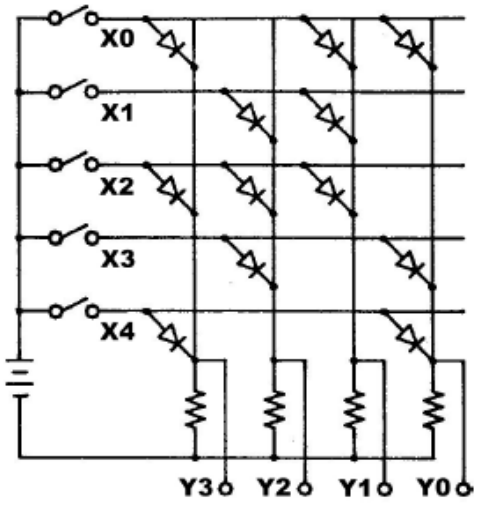
Encoder: Encoder An encoder, a combinational logic circuit, generates a specific output code from one or more inputs. Only one input can be triggered at a time. Figure shows an encoder with n-bit input and n-bit output. When one of the inputs is triggered, an n-bit output code will be generated at the output.



The figure below shows an 8:3 encoder. The circuit has eight inputs (A1~A7) (0~7) and three outputs (Q0, Q1) (000~111). If input A0 is "1", the corresponding output Q2Q1Q0 will be equal to "000". Actually the input A0 is not connected to the input of the gate. If input A1 is "1", Q2Q1Q0=001, if input A2 is "1", Q2Q1Q0=010. There cannot be more than one "1" value between the inputs; for example, if A2 and A3 inputs are "1" at the same time, Q2Q1Q0=011, if A3 and A4 inputs are "1" at the same time, Q2Q1Q0=111 and both outputs are false.



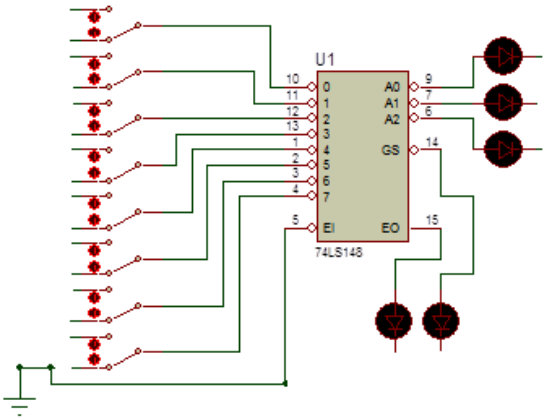
Matrix Encoder: If the encoders on the market do not meet the desired specifications, a matrix encoder can be built using diodes. Figure shows a simple matrix encoder made with diodes.



In some digital applications it may be necessary to process various input signals according to a certain priority. A special type of encoder, called a priority encoder, fulfills this function. When a high priority input is triggered, the output receives the value corresponding to that input regardless of lower priority

inputs. The 74148 is an 8:1 priority encoder with Binary output. Input priority is in ascending order, i.e. input 1 has the lowest priority and input 8 has the highest priority. The outputs are in Binary code.

Procedure for both Prelab and Lab Experiment-2:



10 (S1)	11 (S2)	12 (S3)	13 (S4)	1 (S5)	2 (S6)	3 (S7)	4 (S8)	A0	A1	A2	GS	Eo
0	0	0	0	0	0	0	0					
0	0	0	0	0	0	0	1					
0	0	0	0	0	0	1	1					
0	0	0	0	0	1	1	1					
0	0	0	0	1	1	1	1					
0	0	0	1	1	1	1	1					
0	0	1	1	1	1	1	1					
0	1	1	1	1	1	1	1					
1	1	1	1	1	1	1	1					

Report: Please send the following outputs to the relevant research assistants via e-mail (including Proteus outputs)

1- Simulating the assignments in the Prelab section in the Proteus software programme and processing the outputs in the truth table

2- Establishing the circuits in the Procedure section in the laboratory and filling the relevant tables

Electrical- Electronics Engineering

Computer Engineering

Digital System Design

Experiment-5

Prepared By:

Research Asst. Hatice AKTAŞ AYDIN

Research Asst. İlhan ERDOĞAN

Research Asst. İremnur DURU

2025

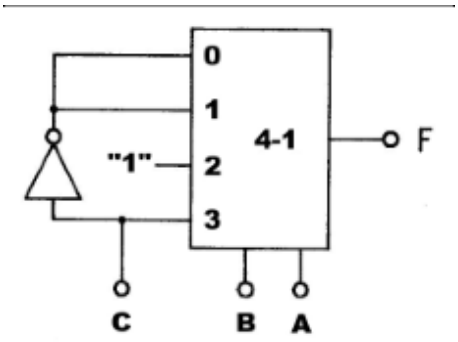
EXPERIMENT-5

Objective-1: Understanding the working principles of Multiplexers (MUX) and Demultiplexers (DEMUX).

Apparatus Required: 74LS151

Theoretical Background:

Multiplexers (MUX): a logic circuit that selects one of its inputs and sends it to the output. One of its inputs is selected by the select inputs and this input is sent to the output. The output is unique. The number of selected inputs determines the capacity of the multiplexer. For example, if the multiplexer has a single select input, the circuit is called a 2:1 (two-to-one) multiplexer, because a single select input can select between two inputs. A multiplexer with three inputs is called an 8:3 (three by eight) multiplexer because it can select from eight inputs (2^3=8). Using a multiplexer, logic functions such as F(CBA)= Ç (0, 1, 2, 6, 7) can be easily implemented. The function F is the sum of the products of the states 0, 1, 2, 6, 7. If we look at the 4:1 multiplexer below, we can see that the output is determined by the inputs A, B, C. When CBA=000, 001, 010, 110, 111 the output F is "1". In all other cases F=0.

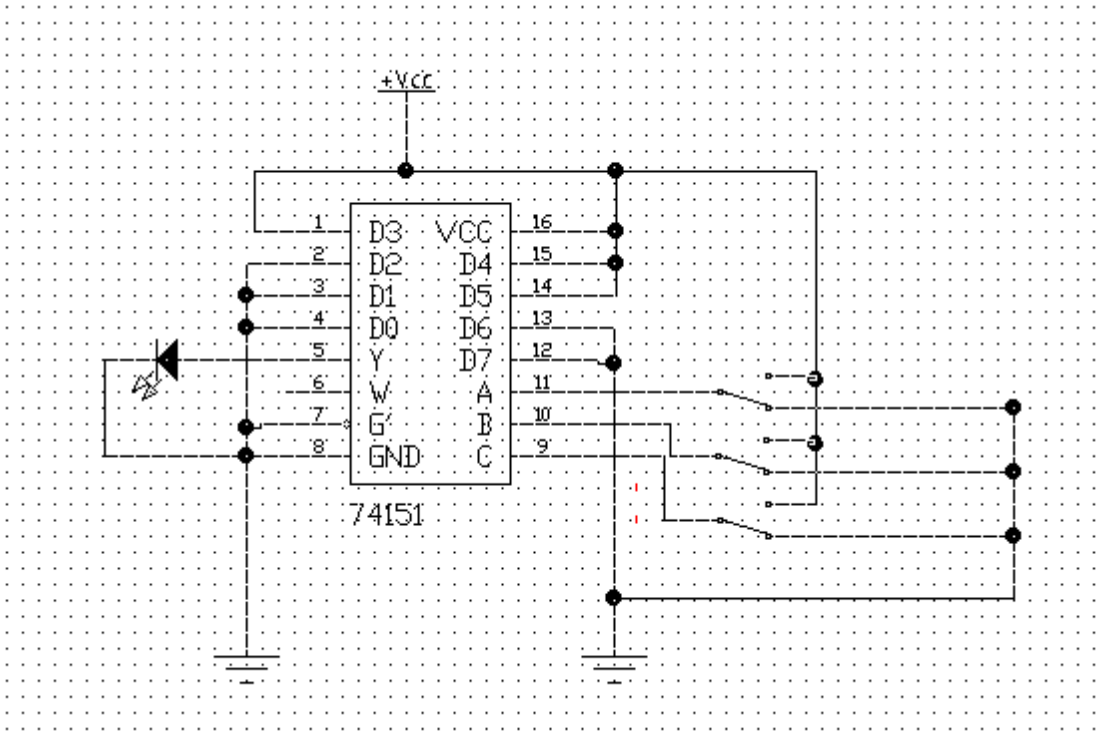


Şekil 2.7

Demultiplexers operate in the opposite way to multiplexers. The 4051 demultiplexer is most commonly used (C input terminal).

Procedure for PreLab:

1. Build the circuit presented in the figure below (input terminals D0, D1, D2,..., D7)
2. Connect input A to S1, input B to S2, input C to switch S3, output Y to the "logic indicators" LED.
3. Fill in the table by switching the switches from 0 to 1 in sequence.



C(S2)	B(S1)	A(S0)	Y
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

Objective-2: To verify the truth table of MUX and DEMUX using logic gates.

Apparatus Required: 7400, 7410, 7420, etc.

Procedure for PreLab and Lab Experiment:

1. Design both the 4x1 MUX and 1x4 DEMUX circuits using logic gates that you choose. Simulate them in Proteus for the pre-lab.

2. Tabulate the outputs of your simulated circuit in a truth table. Then implement it in the lab and test it.

Table of MUX

INPUT						OUPUT
A	**B**	**I0**	**I1**	**I2**	**I3**	**Y (V)**
0	**0**	**0**	**X**	**X**	**X**	
0	**0**	**1**	**X**	**X**	**X**	
0	**1**	**X**	**0**	**X**	**X**	
0	**1**	**X**	**1**	**X**	**X**	
1	**0**	**X**	**X**	**0**	**X**	
1	**0**	**X**	**X**	**1**	**X**	
1	**1**	**X**	**X**	**X**	**0**	
1	**1**	**X**	**X**	**X**	**1**	

Table of DEMUX

INPUT			OUPUT			
A	**B**	**D**	**Y0**	**Y1**	**Y2**	**Y3**
0	**0**	**0**				
0	**0**	**1**				
0	**1**	**0**				
0	**1**	**1**				
1	**0**	**0**				
1	**0**	**1**				
1	**1**	**0**				
1	**1**	**1**				

Report: Please send the following outputs to the relevant research assistants via e-mail (including Proteus outputs)

1- Simulating the assignments in the Prelab section in the Proteus software program and processing the outputs in the truth table

2- Establishing the circuits in the Procedure section in the laboratory and filling the relevant tables

Electrical- Electronics Engineering

Computer Engineering

Digital System Design

Experiment-6

Prepared By:

Research Asst. Hatice AKTAŞ AYDIN

Research Asst. İlhan ERDOĞAN

Research Asst. İremnur DURU

2025

EXPERIMENT-6

Objective-1: Understanding the working principles of Comparators

Apparatus Required: IC 7486, IC 7404, IC 7408

Theoretical Background:

Comparators are essential components in digital system design, used to compare two analog voltage levels and produce a binary output indicating which input is greater. When the voltage at the non-inverting input is higher than that at the inverting input, the output of the comparator goes high (logic 1); otherwise, it goes low (logic 0). This simple yet powerful functionality allows comparators to serve in applications such as threshold detection, zero-crossing identification, and analog-to-digital conversion. Unlike typical operational amplifiers, comparators operate in open-loop mode without feedback, enabling fast switching responses. In some cases, to avoid instability caused by small input variations, hysteresis is added—resulting in Schmitt trigger behavior. In this experiment, students will investigate the basic operating principles of comparators and explore their roles in digital circuit design.

Procedure for PreLab and Lab Experiment:

1. Verify the functionality of the logic gates to be used.
2. Connect the circuit components according to the provided circuit diagram.
3. Switch on the Vcc power supply.
4. Apply the input signals and observe the corresponding outputs.
5. Measure the output voltages using a voltmeter and record the readings in a table.
6. Verify that the observed outputs match the expected results.

2- bit Comparator

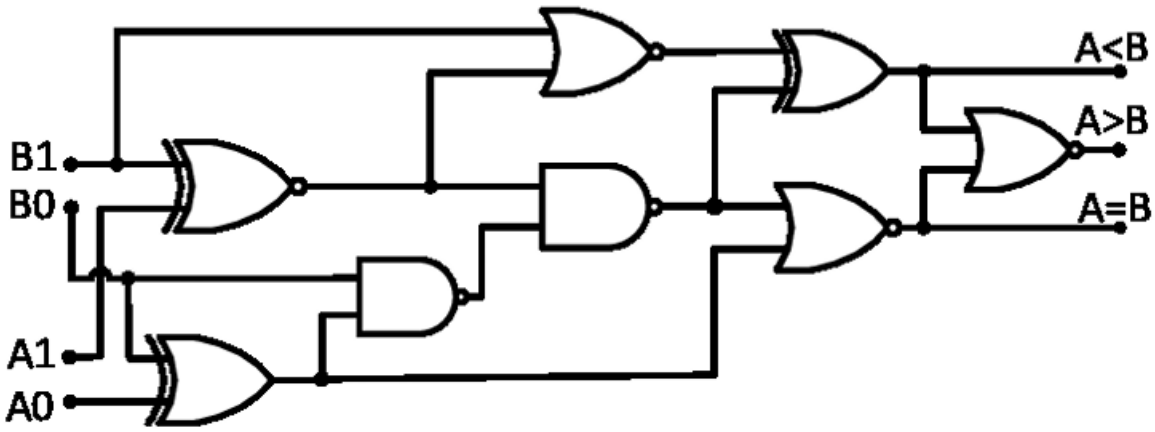


Table: Column For 4-Bit Comparator

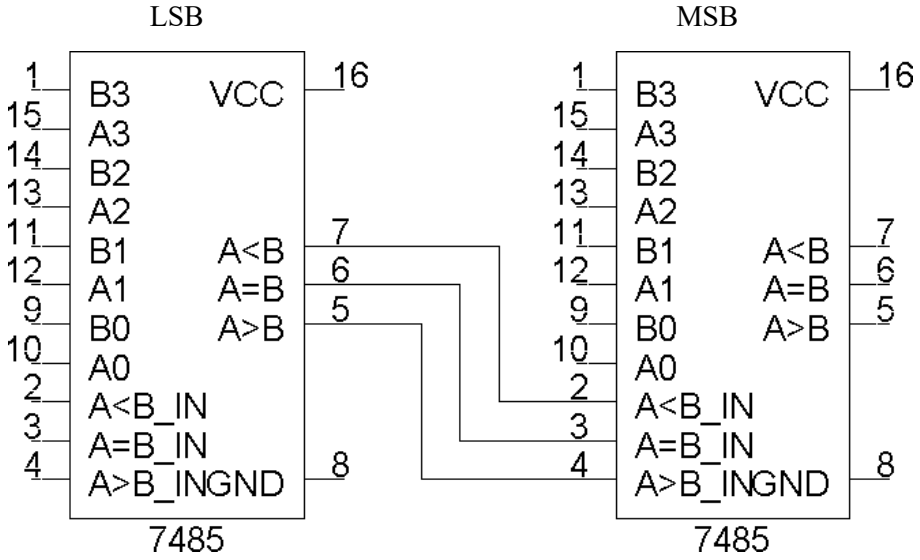
A1	A0	B1	B0	A < B	A = B	A > B
0	0	0	0			
0	0	0	1			
0	0	1	0			
0	0	1	1			

A1	A0	B1	B0	A < B	A = B	A > B
0	1	0	0			
0	1	0	1			
0	1	1	0			
0	1	1	1			
1	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			

Table: Column For 8-Bit Comparator

A_3 B_3	**A_2 B_2**	**A_1 B_1**	**A_0 B_0**	**A>B**	**A=B**	**A<B**	**A>B**	**A=B**	**A<B**
$A_3>B_3$	**X**	**X**	**X**	**X**	**X**	**X**			
$A_3<B_3$	**X**	**X**	**X**	**X**	**X**	**X**			
$A_3=B_3$	**$A_2>B_2$**	**X**	**X**	**X**	**X**	**X**			
$A_3=B_3$	**$A_2<B_2$**	**X**	**X**	**X**	**X**	**X**			
$A_3=B_3$	**$A_2=B_2$**	**$A_1>B_1$**	**X**	**X**	**X**	**X**			
$A_3=B_3$	**$A_2=B_2$**	**$A_1<B_1$**	**X**	**X**	**X**	**X**			
$A_3=B_3$	**$A_2=B_2$**	**$A_1=B_1$**	**$A_0>B_0$**	**X**	**X**	**X**			
$A_3=B_3$	**$A_2=B_2$**	**$A_1=B_1$**	**$A_0<B_0$**	**X**	**X**	**X**			
$A_3=B_3$	**$A_2=B_2$**	**$A_1=B_1$**	**$A_0=B_0$**	**1**	**0**	**0**			
$A_3=B_3$	**$A_2=B_2$**	**$A_1=B_1$**	**$A_0=B_0$**	**0**	**1**	**0**			
$A_3=B_3$	**$A_2=B_2$**	**$A_1=B_1$**	**$A_0=B_0$**	**0**	**0**	**1**			

8-Bit Comparator: -



Report: Please send the following outputs to the relevant research assistants via e-mail (including Proteus outputs)

1- Simulating the assignments in the Prelab section in the Proteus software program and processing the outputs in the truth table

2- Establishing the circuits in the Procedure section in the laboratory and filling the relevant tables

3- Write the truth table for 8-bit comparator and verify the same for the above circuit.

Electrical- Electronics Engineering

Computer Engineering

Digital System Design

Experiment-7

Prepared By:

Research Asst. Hatice AKTAŞ AYDIN

Research Asst. İlhan ERDOĞAN

Research Asst. İremnur DURU

2025

EXPERIMENT-7

Objective-1: Understanding the working principles of Flip-Flops, Truth table verification of Flip-Flops for JK Master Slave, D- Type and T- Type.

Apparatus Required: IC 7410, IC 7400, etc.

Theoretical Background:

Flip-flop circuits are bistable logic devices used as fundamental memory elements in digital electronic systems. These circuits can store a single bit (0 or 1) by changing their state in response to trigger (clock) signals applied to their inputs. Due to this characteristic, flip-flops are considered essential building blocks of digital systems such as counters, registers, and timers. Essentially, flip-flops are bistable circuits with two stable states. These two stable states are represented by logic 1 (HIGH) and logic 0 (LOW). Flip-flops are typically triggered by a clock signal and change their output based on the input data at specific times. This behavior is crucial for timing control in synchronous digital systems.

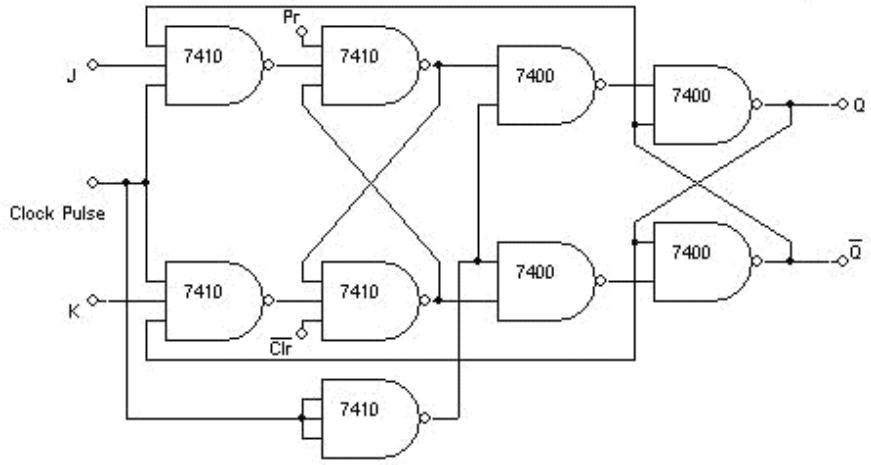


Figure 1: Circuit Diagram of Master Slave JK Flip-Flop

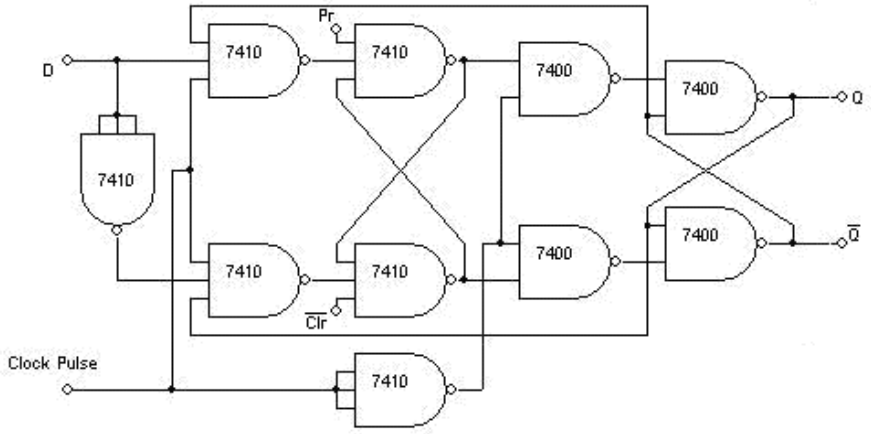


Figure 2: Circuit Diagram of D Flip-Flop

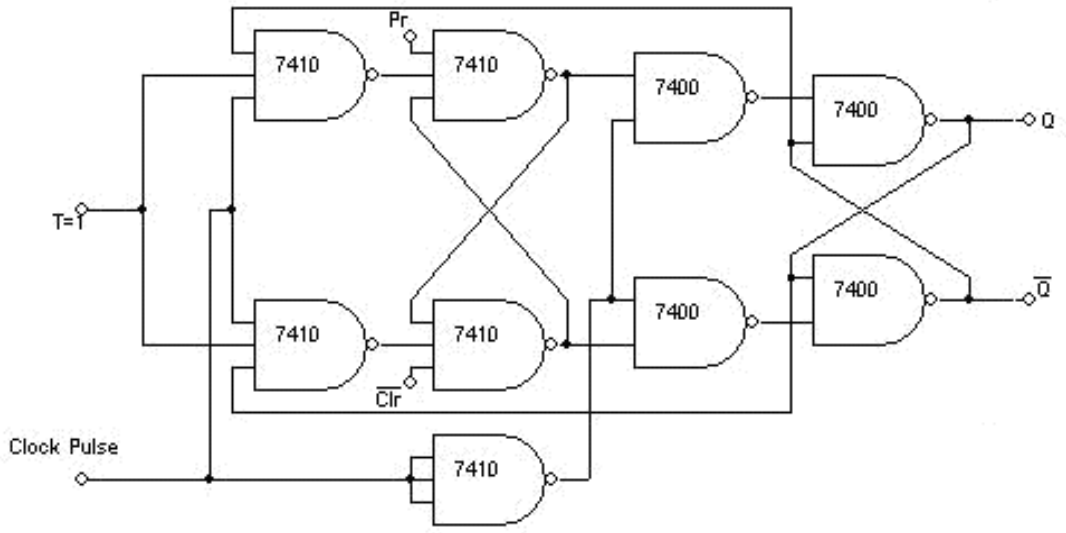


Figure 3: Circuit Diagram of T Flip-Flop

Experiment Stages

1. Connections are made as per circuit diagram.
2. The truth table is verified for various combinations of inputs.

Table 1: Truth Table for Master Slave JK Flip-Flop

Preset	Clear	J	K	Clock	Qn+1	$\overline{Qn+1}$	
0	1	X	X	X			
1	0	X	X	X			
1	1	0	0	⎍			
1	1	0	1	⎍			
1	1	1	0	⎍			
1	1	1	1	⎍			

Table 2: Truth Table for D Flip-Flop:

Preset	Clear	D	Clock	Qn+1	$\overline{Qn+1}$
1	1	0	⎍		
1	1	1	⎍		

Table 3: Truth Table for T Flip-Flop:

Preset	Clear	T	Clock	Qn+1	$\overline{Qn+1}$
1	1	0	⎍		
1	1	1	⎍		

Write the timing diagrams for all the above Flip-Flops

Report: Please send the following outputs to the relevant research assistants via e-mail (including Proteus outputs)

1- Simulating the assignments in the Prelab section in the Proteus software program and processing the outputs in the truth table

2- Establishing the circuits in the Procedure section in the laboratory and filling the relevant tables

SİVAS BİLİM VE TEKNOLOJİ ÜNİVERSİTESİ

Electrical- Electronics Engineering

Computer Engineering

Digital System Design

Experiment-8

Prepared By:

Research Asst. Hatice AKTAŞ AYDIN

Research Asst. İlhan ERDOĞAN

Research Asst. İremnur DURU

2025

EXPERIMENT-8

Objective-1: Realization of 3-bit counters as a-sequential circuit and Mod-N counter design

Apparatus Required: IC 7408, IC 7476, IC 7 490, IC 74192, IC 74193, IC 7400, IC 7416, IC 7432

Theoretical Background:

The IC 7476 is a dual JK flip-flop with preset and clear capabilities. A JK flip-flop is a type of sequential logic circuit used to store and toggle binary information. It is an improvement over the SR flip-flop, eliminating the undefined state when both inputs are high.

Each flip-flop in the 7476 has inputs for J, K, Clock (CLK), Preset (PRE), and Clear (CLR), as well as outputs Q and $\overline{Q}$ (not-Q). The state of the output changes in response to the clock signal based on the values of the J and K inputs according to the truth table:

J	K	CLK (rising edge)	Q(next state)
0	0	↑	No change
0	1	↑	Reset (0)
1	0	↑	Set (1)
1	1	↑	Toggle

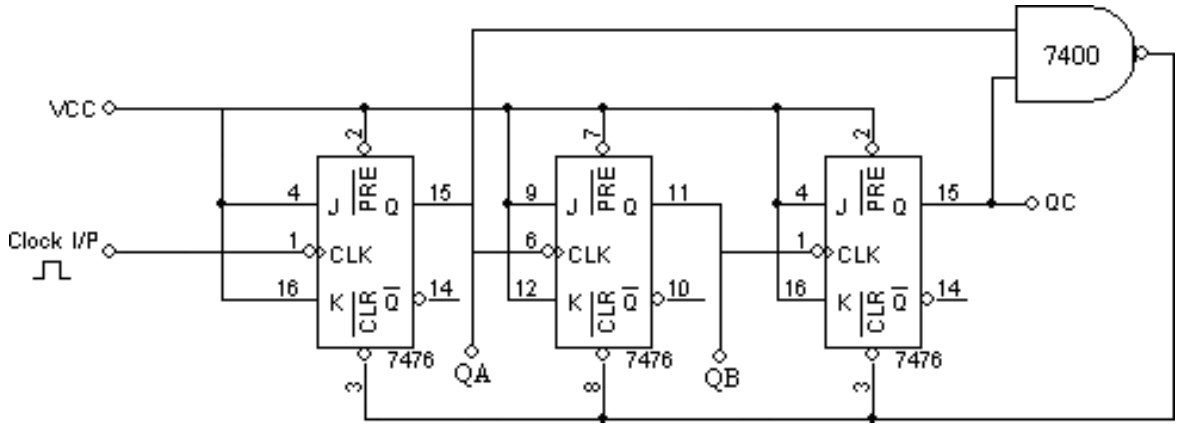
- **Preset (active low)** forces the output Q to 1 regardless of other inputs.
- **Clear (active low)** forces the output Q to 0.

The JK flip-flop operates on the edge of the clock signal, typically the rising edge, making it suitable for synchronous applications such as counters, shift registers, and memory devices.

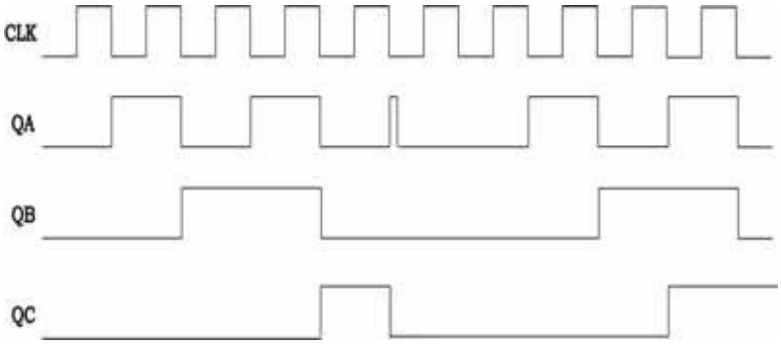
In this experiment, clock pulses are applied manually or from a pulse generator, and the resulting changes in output (QA, QB, QC) are observed to verify the operation of the JK flip-flop and confirm its truth table.

Prelab

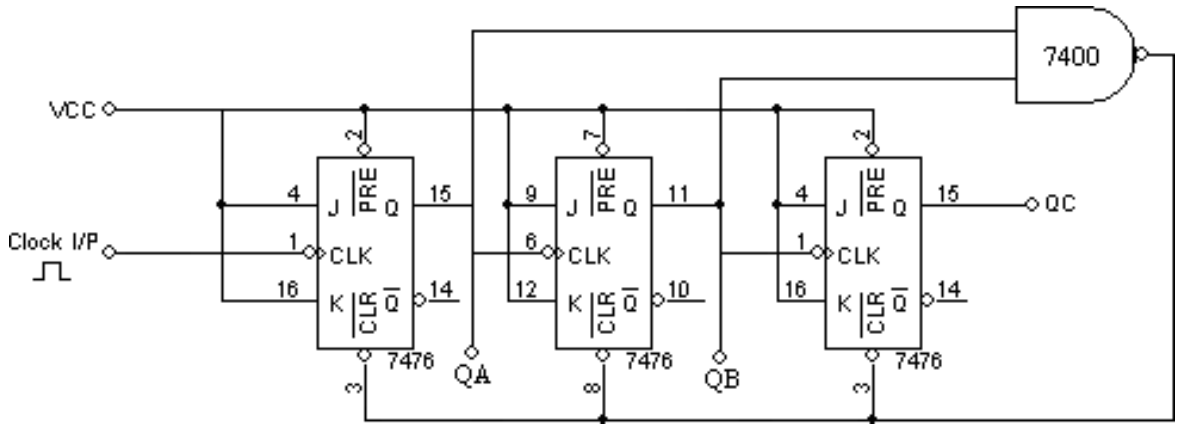
1. **Mod 5 Asynchronous Counter:**



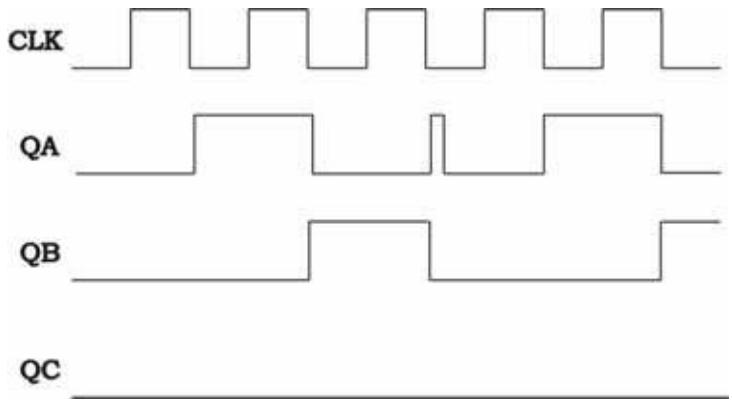
Mod 5 Asynchronous counter			
Clock	**QC**	**QB**	**QA**
0	**0**	**0**	**0**
1	**0**	**0**	**1**
2	**0**	**1**	**0**
3	**0**	**1**	**1**
4	**1**	**0**	**0**
5	**0**	**0**	**0**



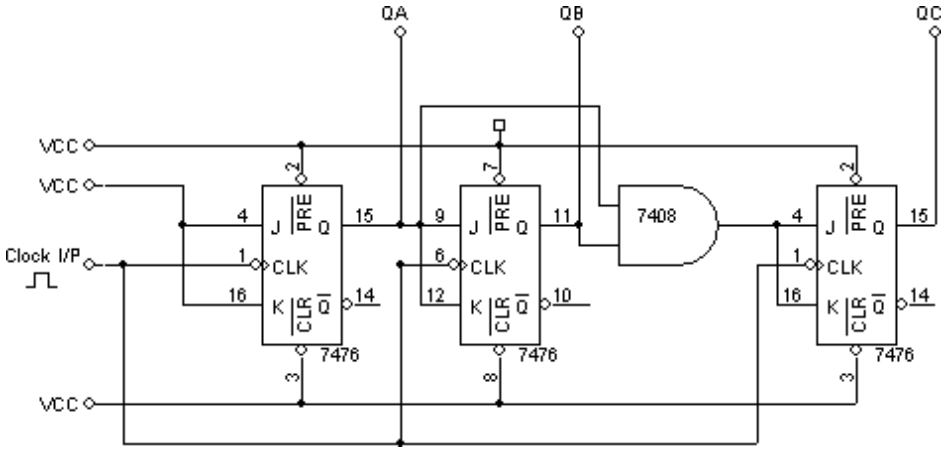
2. **Mod 3 Asynchronous Counter:**

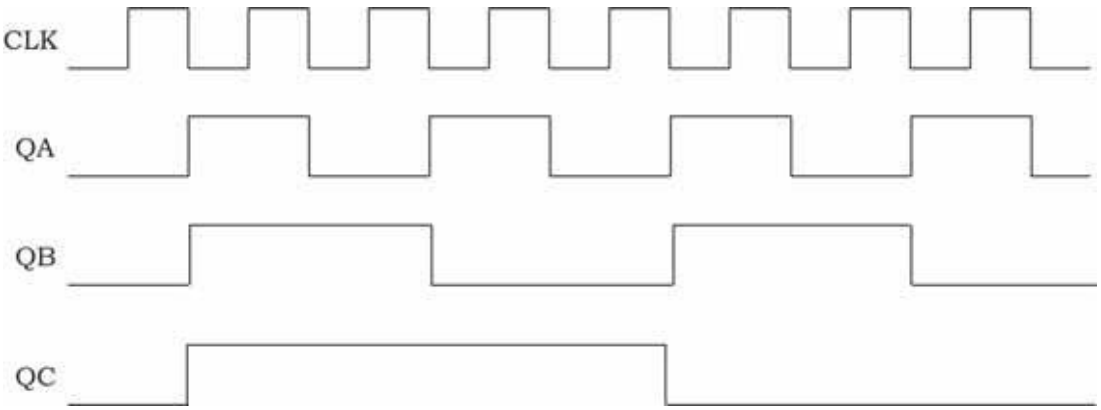


Mod 3 Asynchronous counter			
Clock	**QC**	**QB**	**QA**
0	**0**	**0**	**0**
1	**0**	**0**	**1**
2	**0**	**1**	**0**
3	**0**	**0**	**0**
4	**0**	**0**	**1**
5	**0**	**1**	**0**

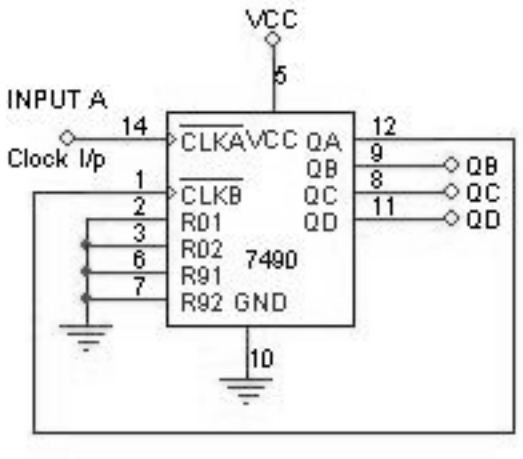


3. **3-bit Synchronous Counter:**



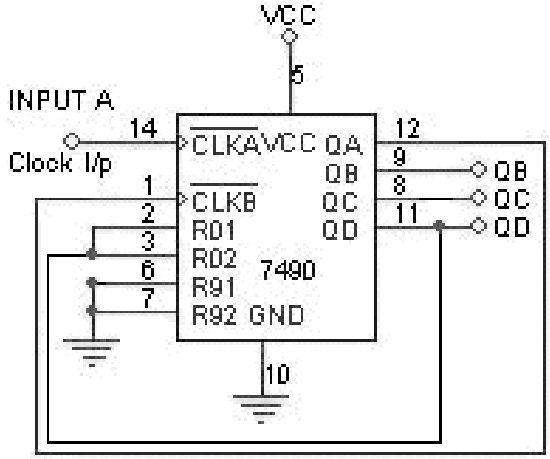


4. **IC 7490 (Decade Counter):**



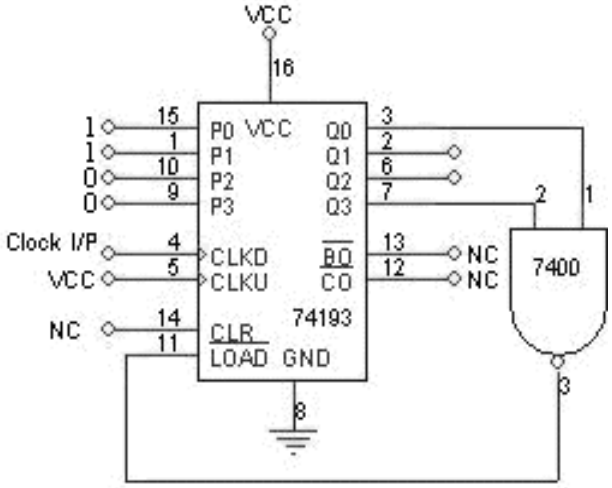
Clock	QD	QC	QB	QA
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	0	0	0	0

5. IC 7490 (MOD-8 Counter):



Clock	QD	QC	QB	QA
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	0	0	0	0
9	0	0	0	1

6. Circuit Diagram (IC 74193) To Count from 3 to 8:



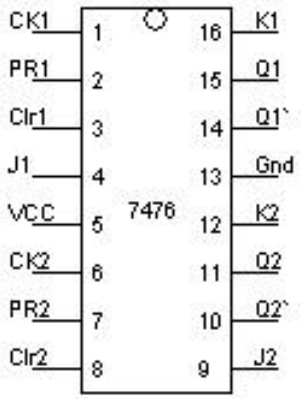
Clock	QD	QC	QB	QA	Count in Decimal
0	0	0	1	1	3
1	0	1	0	0	4
2	0	1	0	1	5
3	0	1	1	0	6
4	0	1	1	1	7
5	1	0	0	0	8
6	0	0	1	1	3
7	repeats				4

Experimental study

Experiment Stages

1. Connections are made as per circuit diagram.
2. Clock pulses are applied one by on e at the clock I/P and the O/P isobserved at QA, QB & QC for IC 7476.
3. Truth table is verified.

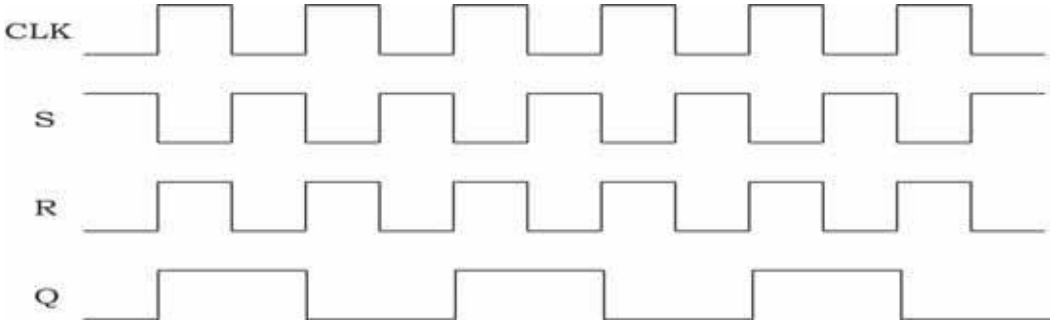
Pin Details:



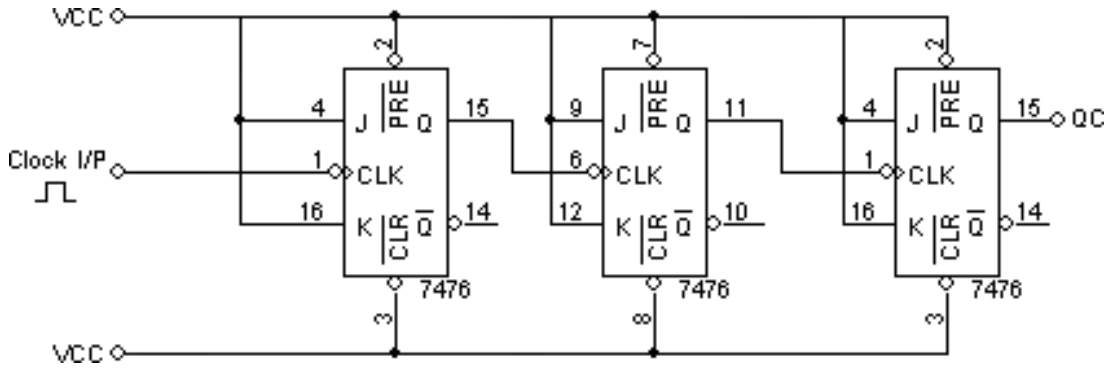
Truth Table

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

Timing Diagram:

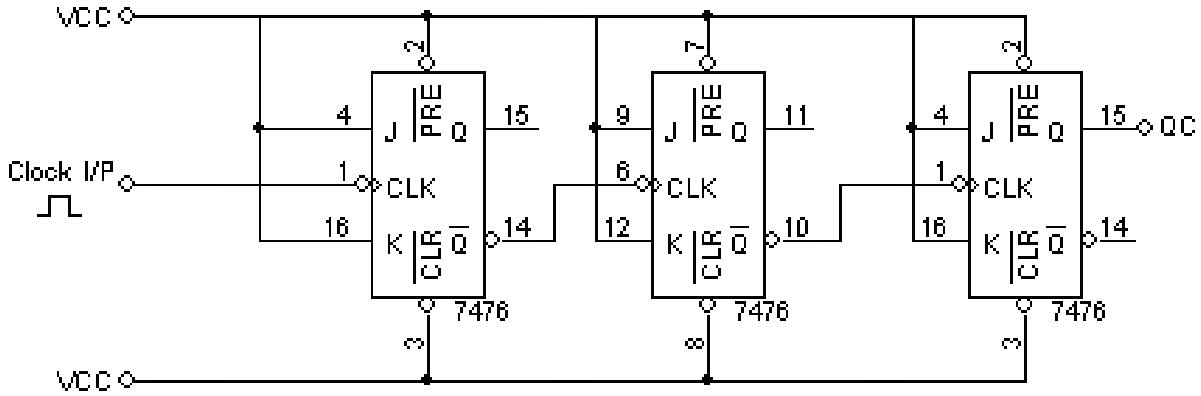


1. **Circuit Diagram: - 3-Bit Asynchronous Up Counter**



3-bit Asynchronous up counter			
Clock	**QC**	**QB**	**QA**
0	**0**	**0**	**0**
1	**0**	**0**	**1**
2	**0**	**1**	**0**
3	**0**	**1**	**1**
4	**1**	**0**	**0**
5	**1**	**0**	**1**
6	**1**	**1**	**0**
7	**1**	**1**	**1**
8	**0**	**0**	**0**

2. Circuit Diagram: - 3-Bit Asynchronous Down Counter



3-bit Asynchronous down counter			
Clock	**QC**	**QB**	**QA**
0	**1**	**1**	**1**
1	**1**	**1**	**0**
2	**1**	**0**	**1**
3	**1**	**0**	**0**
4	**0**	**1**	**1**
5	**0**	**1**	**0**
6	**0**	**0**	**1**
7	**0**	**0**	**0**
8	**1**	**1**	**1**
9	**1**	**1**	**0**