Subject Code CS 254	Digital Systems Laboratory	Credits: 2 (0-0-3) Total hours: 42
Course Objectives	The course provides practical knowledge in designing the digital systems	
List of Experiments		
(1) Simplification, realization of boolean expressions using logic gates/universal gates (2) Realization of half/full adder & half/full subtractors using logic gates (3) Realization of parallel adder/subtractors using 7483 chip, BCD to Excess-3code conversion & vice versa (4) Realization of binary to gray code conversion & vice versa (5) MUX/DEMUX – use of 74153,74139 for arithmetic circuits & code converter (6) Realization of one/two bit comparator and study of 7485 magnitude comparator (7) Use of a) Decoder chip to drive LED display & b) Priority encoder (8) Truth table verification of flip-flops: i) JK Master Slave ii) T type iii) D type (9) Realization of 3 bit counters as a sequential circuit & MOD-N counter design (7476,7490,74192,74193) (10) Writing & testing of sequence generator		
Reference books	(1) J. Bhasker, "A VHDL primer", 3rd edition, Addison Wesley Longmen, 1999. (2) Douglas Perry, "VHDL: Programming by example", 4th ed. McGraw Hill International, 2002. (3) Peter Ashenden, "The Designer Guide to VHDL",Morgan Kaufmann, 1998	

Experiment.1: To simplify and realize Boolean Expression using Logic Gates/Universal Gates.

AIM: To Simplify and Realize Boolean Expressions Using Logic Gates/Universal Gates.

APPARATUS REQUIRED: - IC Trainer Kit, patch chords, IC7408, IC7432, IC7400, IC7402, IC 7404, IC 7486.

Procedure:

1. Place the IC in the socket of the trainer kit.

2. Make the connections as shown in the circuit diagram.

3. Apply the different combinations of input according to the truth table and verify the output.

Simplification of expression: -

Y=(A,B,C,D)= Σ(5,7,9,11,13,15)

Write the expression using K-map

Y=A'BC'D + A'BCD +AB'C'D+AB'CD + ABC'D + ABCD

Y =A'BD (C+C') + AB'D(C+C') + ABD(C+C')

Y =A'BD + AB'D + ABD

Y =BD(A+A') + AB'D

Y=BD + AB'D

Y=D(B+AB')

Y=D(A+B)

Y =AD+BD

Exercise: Simplify and realize the following POS expn. and implement using NAND gates only:
Y(A,B,C,D) = π (5,7,9,11,13,15)

GIVEN EXPRESSION:

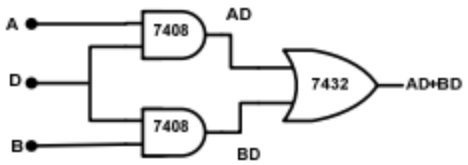
Y=(A,B,C,D)= Σ(5,7,9,11,13,15)

Truth-table:

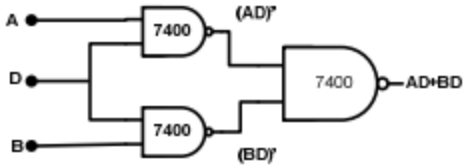
A	B	C	D	Y= AD+BD
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Implementation:

Using Basic Gates:



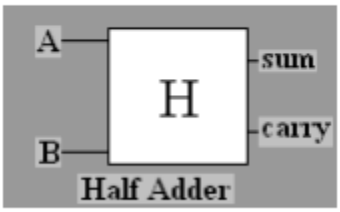
Using Universal Gates:



Exercise: Simplify and realize the following POS and implement using NAND gates only: $Y(A,B,C,D)=\pi(5,7,9,11,13,15)$

Half Adder:

Logic Diagram

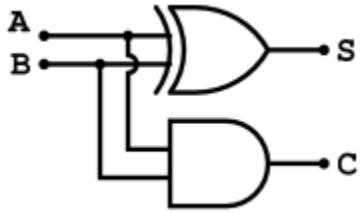


Truth Table

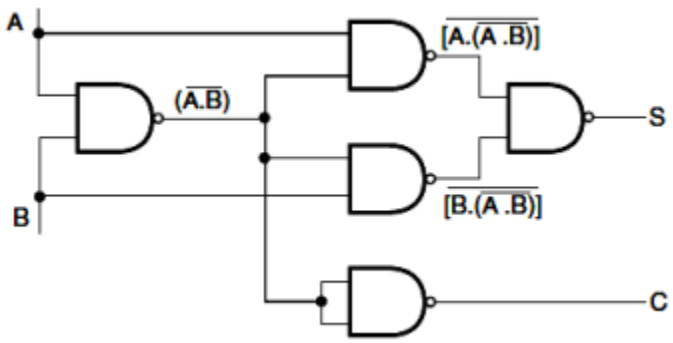
A	B	Sum (S)	Carry (C)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Circuit Diagram:

1. **USING BASIC AND XOR GATES**



2. **USING NAND GATES ONLY**



Experiment 2: HALF/FULL ADDER AND HALF/FULL SUBTRACTORS.

Aim: - To realize half/full adder and half/full subtractor.

i. Using X-OR and basic gates

ii. Using only NAND gates.

Apparatus Required: -

IC Trainer Kit, patch chords, IC 7486, IC 7432, IC 7408, IC 7400, etc.

Procedure: -

1. Verify the gates.

2. Make the connections as per the circuit diagram.

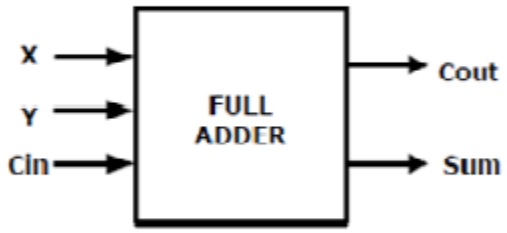
3. Switch on VCC and apply various combinations of input according to the truth table.

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.

Exercise: Implement half/full adder and half/full adder circuits using NOR gates only. Which is better, NAND or NOR? Why?

Full Adder

Logic Diagram

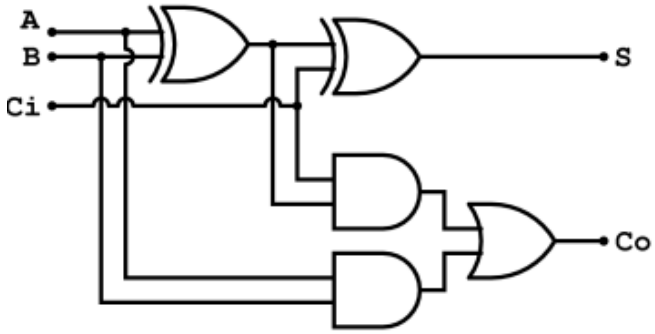


Truth Table

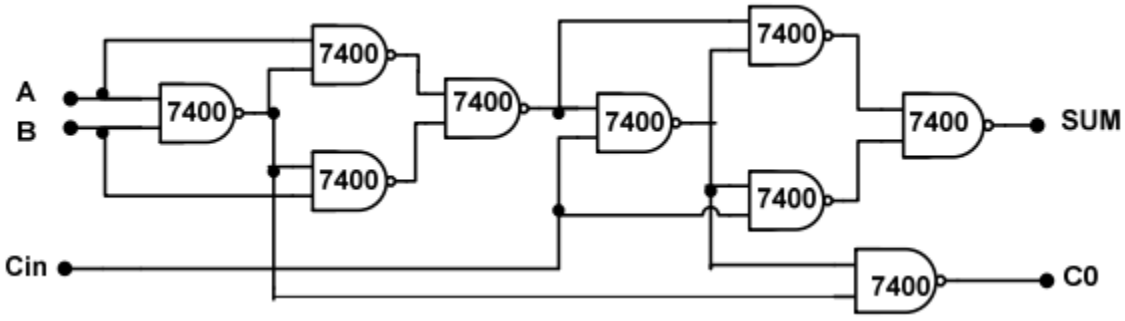
C_i	A	B	Sum (S)	Carry (C_o)
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Circuit Diagram

USING BASIC AND XOR GATES

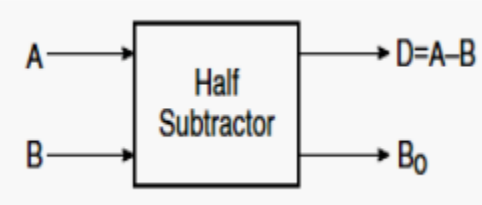


USING NAND GATES ONLY



Half subtractor

Logic Diagram

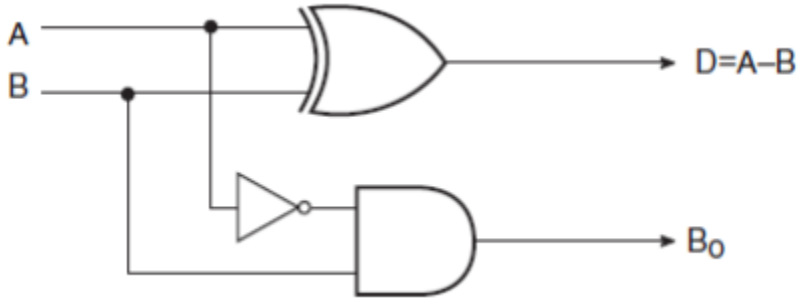


Truth Table

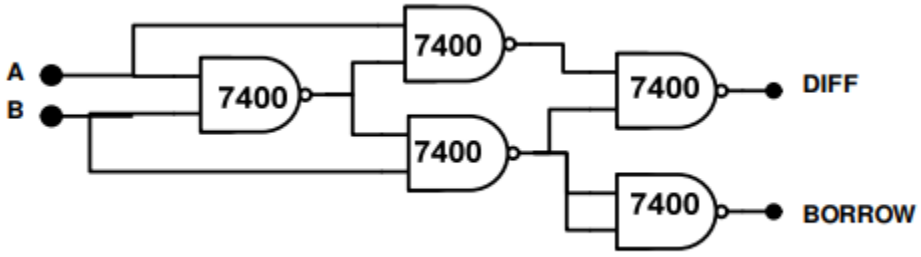
A	B	Diff (D)	Borrow (B_0)
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Circuit Diagram

1.USING BASIC AND XOR GATES

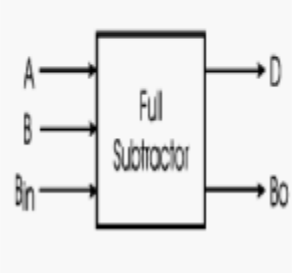


2. USING NAND GATES ONLY



Full subtractor

Logic Diagram:

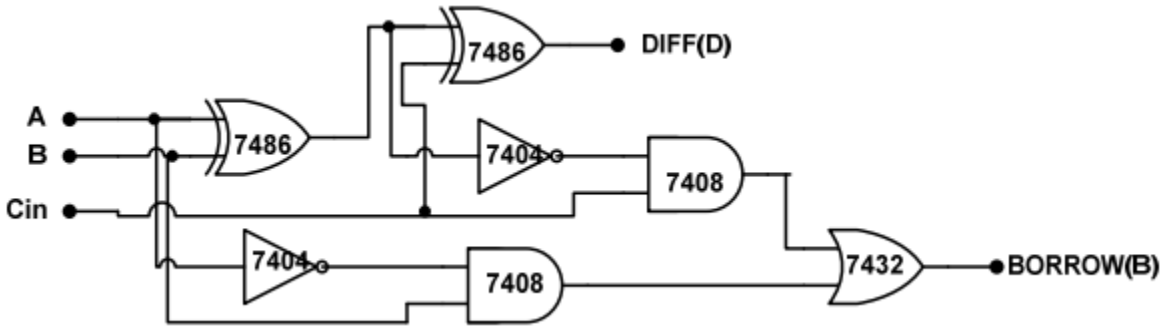


Truth Table

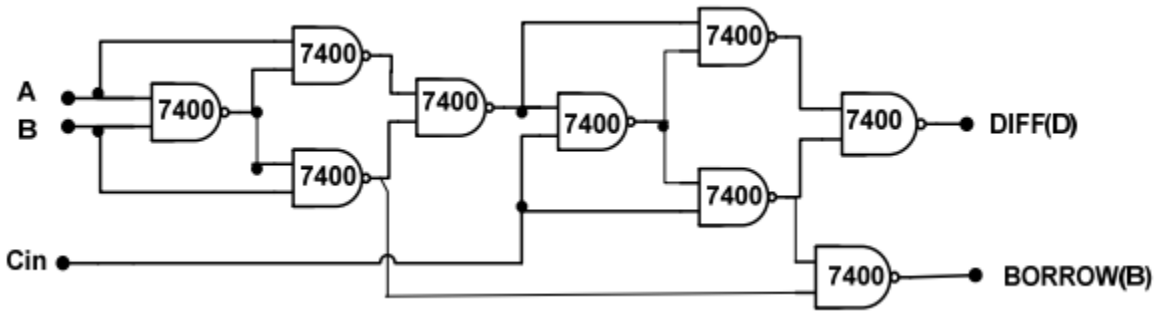
Minuend (A)	Subtrahend (B)	Borrow In (B_{in})	Difference (D)	Borrow Out (B_O)
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Circuit Diagram

1. USING BASIC AND XOR GATES



2. USING NAND GATES ONLY



I. Experiment: PARALLEL ADDER/SUBTRACTOR

AIM: - To realize IC 7483 as parallel adder / Subtractor.

Apparatus Required: - IC Trainer Kit, patch chords, IC 7483, IC 7404, etc.

Procedure: -

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

2. Connect C0 to the Ground.

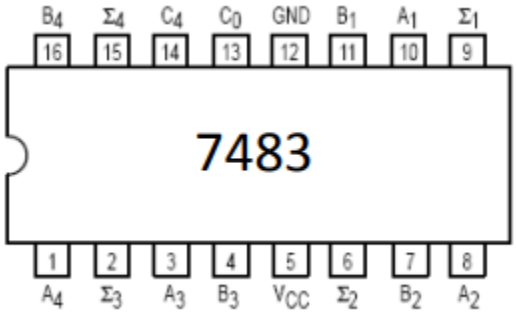
3. Check the output sum on the S0 to S3 and also C4.

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

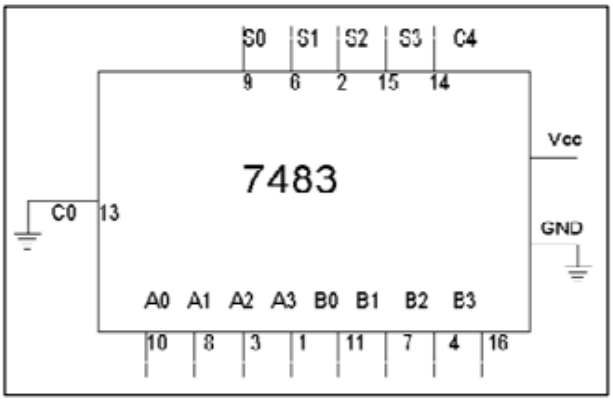
5. The truth table of adder and Subtractor are noted down.

Exercise: Implement a parallel adder/subtractor using IC 7483 and Xor gates.

Pin Detail: -



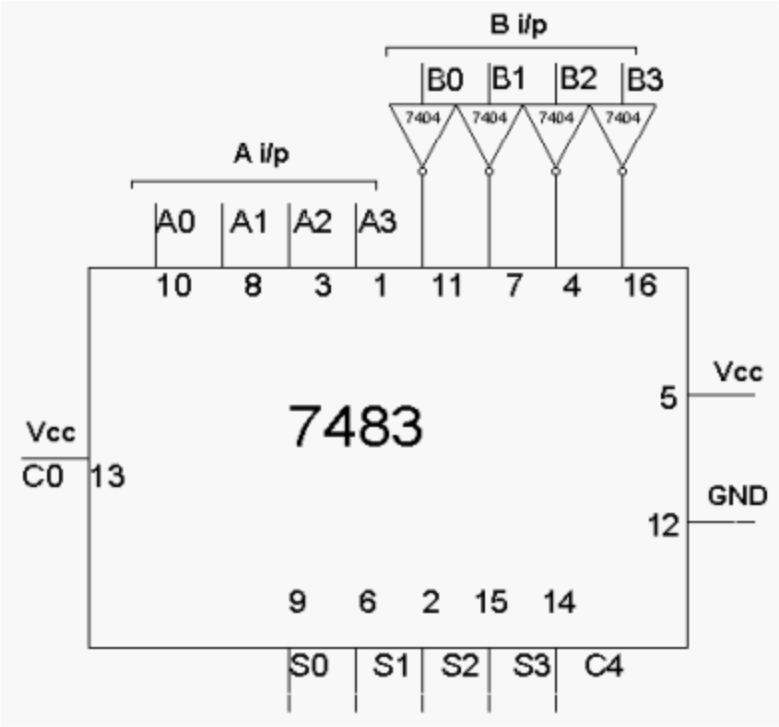
Adder: -



Truth Table: -

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

Subtractor:-



Truth Table for 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	1	0	0	0	1
0	1	0	1	0	0	1	1	1	0	0	1	0
0	0	1	1	0	1	0	1	0	1	1	1	0
1	0	1	0	0	1	1	0	1	0	1	0	0
1	0	0	0	1	1	1	1	0	1	0	0	1

II. BCD TO EXCESS-3 AND EXCESS-3 TO BCD CODE CONVERTER

AIM: - To study and verify BCD to excess –3 code and excess-3 to BCD code conversion using NAND gates.

APPARATUS REQUIRED: - IC Trainer Kit, patch chords, IC 7400, IC 7404, etc.

Procedure: - (BCD Excess 3 and Vice Versa)

1. Make the connections as shown in the fig.

2. Pin [14] of all IC'S are connected to +5V and pin [7] to the ground.

3. The inputs are applied at E3, E2, E1, and E0 and the corresponding outputs at B3, B2, B1, and B0 are taken for excess – 3 to BCD.

4. The inputs are applied at B3, B2, B1, and B0 and the corresponding outputs are E3, E2, E1 and E0 for BCD to excess – 3.

5. Truth tables are verified.

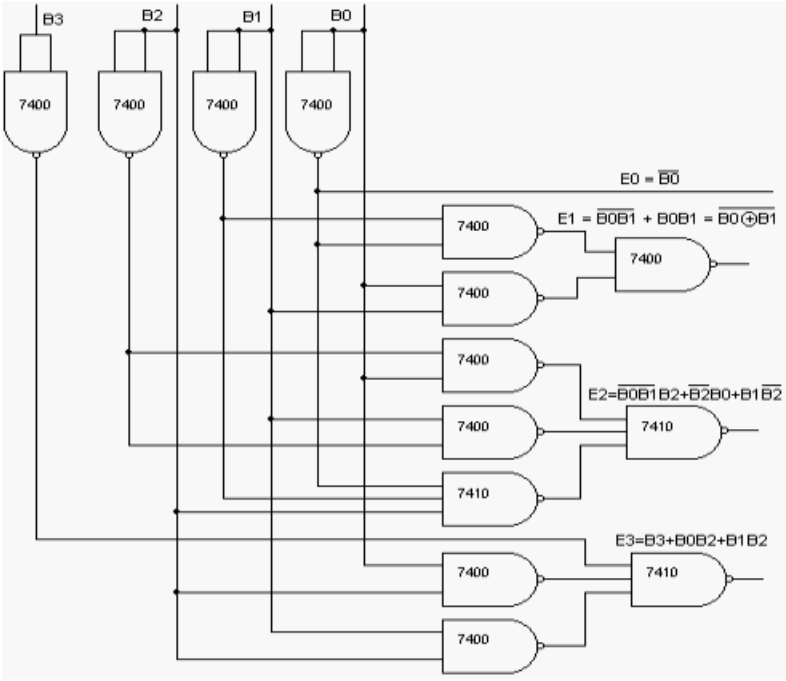
Exercise: Implement BCD to excess-3 and excess-3 to BCD code converter using parallel adder IC 7483.

BCD To Excess-3:

Truth Table For Code Conversion: -

Inputs				Outputs			
B3	B2	B1	B0	E3 (v)	E2 (v)	E1 (v)	E0 (v)
0	0	0	0	0	0	1	1
0	0	0	1	0	1	0	0
0	0	1	0	0	1	0	1
0	0	1	1	0	1	1	0
0	1	0	0	0	1	1	1
0	1	0	1	1	0	0	0
0	1	1	0	1	0	0	1
0	1	1	1	1	0	1	0
1	0	0	0	1	0	1	1
1	0	0	1	1	1	0	0

Circuit diagram using NAND gates

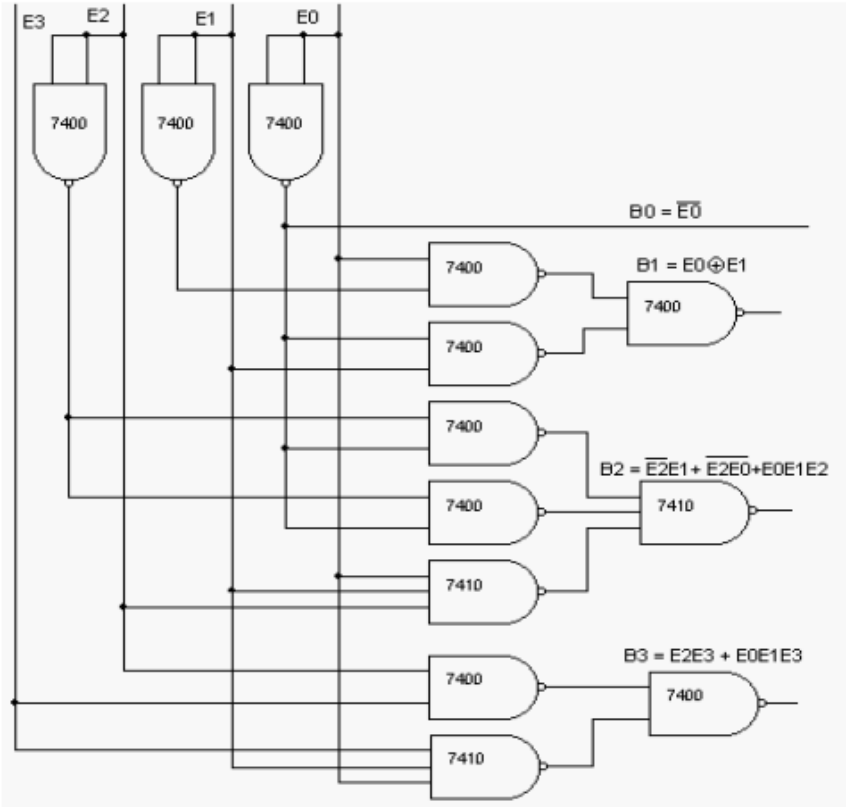


Excess-3 To BCD :-

Truth Table For Code Conversion: -

Inputs				Outputs			
E3	E2	E1	E0	B3 (v)	B2 (v)	B1 (v)	B0(v)
0	0	1	1	0	0	0	0
0	1	0	0	0	0	0	1
0	1	0	1	0	0	1	0
0	1	1	0	0	0	1	1
0	1	1	1	0	1	0	0
1	0	0	0	0	1	0	1
1	0	0	1	0	1	1	0
1	0	1	0	0	1	1	1
1	0	1	1	1	0	0	0
1	1	0	0	1	0	0	1

Circuit diagram using NAND gates only:



Experience : BINARY-TO-GRAY AND GRAY-TO-BINARY CODE CONVERTER

AIM: - To convert given binary numbers to gray codes.

APPARATUS REQUIRED: - IC Trainer Kit, patch chords, IC 7486, etc

PROCEDURE: -

1. The circuit connections are made as shown in fig.

2. Pin (14) is connected to +Vcc and Pin (7) to ground.

3. In the case of binary to gray conversion, the inputs B0, B1, B2 and B3 are given at respective pins and outputs G0, G1, G2, G3 are taken for all the 16 combinations of the input.

4. In the case of gray to binary conversion, the inputs G0, G1, G2 and G3 are given at respective pins and outputs B0, B1, B2, and B3 are taken for all the 16 combinations of inputs.

5. The values of the outputs are tabulated.

Exercise:

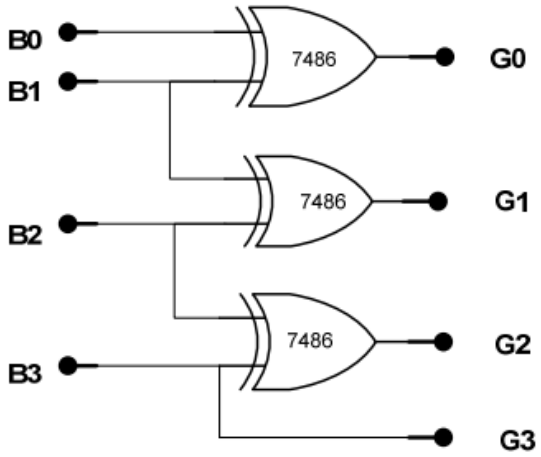
Implement binary to gray and gray to binary code converter using NAND gates only.

Binary to Gray:

Truth-table:

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

Circuit diagram using EX-OR Gates:



Gray to binary:

Truth-table:

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

Circuit Diagram using EX-OR Gates