**ANGELES UNIVERSITY FOUNDATION**

*College of Engineering and Architecture*

Computer Engineering Department

| **NAME OF STUDENT:** |
| --- |
| **GROUP NUMBER: DATE PERFORMED:** |
| **COURSE CODE:** *AEC 41* **DATE SUBMITTED:** |
| **COURSE TITLE:** *Logic Circuits and Switching Theory***YEAR AND SECTION:** |
| **LAB. INSTRUCTOR: GRADE:** |

EXPERIMENT NO. 7

**COMBINATIONAL CIRCUIT (CODE CONVERTERS)**

1. ***OBJECTIVES:***
2. To design code converters and verify their truth tables.
3. To familiarize students with the functionality of code converters.
4. ***MATERIALS AND EQUIPMENT:***

1 - Digital Trainer

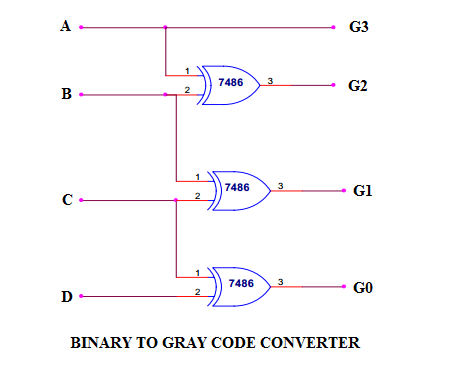
1 - Logic Probe

1 - TTL 74LS86

1 - IC Remover

1 - Set Connecting Copper Wires

1. ***INTRODUCTION:***

Code converter is acombinational circuit that translates the input code word into a new corresponding word. In this experiment, it will show the circuit design of binary to gray code and vice versa using Exclusive-OR gate. An interesting application for theexclusive-OR gate is a logic gate to change abinary number to its equivalent in Gray Code. The logic circuit shown in Figure 1 can beused to convert a 4-bit binary number ABCD into its Gray-code equivalent, G3, G2, G1 and G0.As an example, the binary number0011 will be converted into its Gray-Code equivalent of 0010 by the circuit in Figure 1. Note: Ais the most significant bit and D is the least significant bit.

**Figure 1**

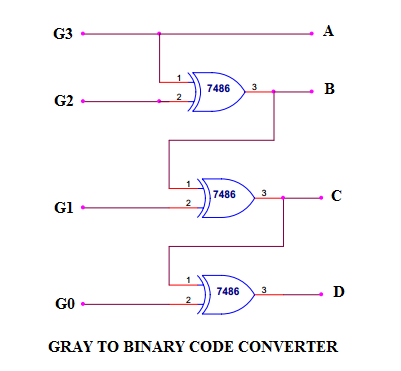
1. ***PROCEDURE:***
   1. Using the TTL 74LS86, constructthe equivalent circuit shown in Figure 1.
   2. Connect the binary input A, B, C, and D to the data switches SW1, SW2, SW3, and SW4 respectively. Then connect the output G3, G2, G1 and G0 to the logic indicators L4, L3, L2 and L1 respectively.
   3. Turn on the trainer. Observe the Output L4, L3, L2 and L1 based from the given conditions in Table 1. Record your readings in Table 1.

**Table 1**

| **SW1 (A)** | **SW2 (B)** | **SW3 (C)** | **SW4 (D)** | **L4 (G3)** | **L3 (G2)** | **L2 (G1)** | **L1 (G0)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 |  |  |  |  |
| 0 | 0 | 0 | 1 |  |  |  |  |
| 0 | 0 | 1 | 0 |  |  |  |  |
| 0 | 0 | 1 | 1 |  |  |  |  |
| 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 |  |  |  |  |

* 1. Using the same procedure, construct the circuit shown in Figure 2 to evaluate experimentally the output of Gray Code to Binary Converter.

***Note:*** *G3, G2, G1 and G0 will be the input and A,B,C,D will be the output.*



**Figure 2**

* 1. Connect the binary input G3, G2, G1, and G0 to the data switches SW1, SW2, SW3, and SW4 respectively. Then connect the output A, B, C and D to the logic indicators L4, L3, L2 and L1 respectively.
  2. Turn on the trainer. Observe the Output L4, L3, L2 and L1 based from the given conditions in Table 2. Record your readings in Table 2.

**Table 2**

| **SW1 (G3)** | **SW2 (G2)** | **SW3 (G1)** | **SW4 (G0)** | **L4 (A)** | **L3 (B)** | **L2 (C)** | **L1 (D)** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 |  |  |  |  |
| 0 | 0 | 0 | 1 |  |  |  |  |
| 0 | 0 | 1 | 0 |  |  |  |  |
| 0 | 0 | 1 | 1 |  |  |  |  |
| 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 |  |  |  |  |

1. ***QUESTIONS:***
2. Convert binary 100100 to gray code.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. What are the applications of code conversion?

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. ***PROBLEM:***
2. Using the basic logic gates AND, OR and INVERTER, Design a combinational circuit that will convert Binary Numbers to Excess-Three Codes. Show the truth table and draw the equivalent circuit inside the box.



1. ***CONCLUSION:***

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_