Experiment No. 2

Aim: To study, design, implement various code converters.

# Objectives:

1. To study various types of codes and its specialities.
2. To analyse various codes, code converters and its applications.
3. To understand the process of code conversion such as binary codes to gray codes and vice versa.
4. To understand simplification of circuits using K- map and Boolean algebra.
5. To design, realize and implement various code converters using gate ICs.
6. To understand various commercial code converters ICs available.

(4-bit binary to gray code converter and vice versa, BCD to Excess 3 code converters, Commercial code converters ICs available)

# Equipment:

Gate ICs (Ex-OR), 5VDC regulated power supply, bread board, connecting wires, LED, DMM, etc.

# Theory:

## Code conversion

A code is a rule for converting a piece of information into another form or representation, not necessarily of the same type. In communications and information processing, encoding is the process by which information from a source is converted into symbols to be communicated. Decoding is the reverse process, converting these code symbols back into information understandable by a receiver.

## Necessity of code conversion:

Computers, microprocessors, microcontrollers etc. are digital systems. There are different number systems available like binary, hexadecimal, octal etc. All integers in decimal system can be converted into 1s and 0s in binary system (digital system). This process of conversion from one system to another is called as code conversion. Code conversion is a process by which all the decimal numbers and alphabets can be specified using other symbols (usually by 1s and 0s). Code conversion is necessary in order to interface the digital circuits for signal processing.

## ASCII Code:

The American Standard Code Information Interchange, or ASCII, uses a 7 bit binary code to represent text within a computer, communications equipment, and other devices that use text. Each letter or symbol is assigned to a number from 0 to 127. For example, in the 8-bit ASCII code, a lowercase "a" is represented by the bit string 01100001.

## Natural BCD Code:

A code in which a string of four binary digits represents each decimal number 0 through 9 as a means of preventing calculation errors due to rounding and conversion. For example, since the binary equivalent of 3 is 0011 and the binary equivalent of 6 is 0110, then 36 would be represented as 0011 0110.

## Excess 3 Code:

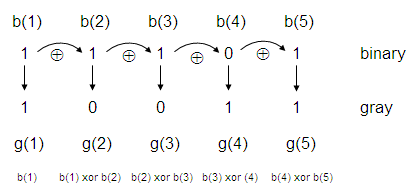
A binary code for which the weighted sum of the four bits in each code-word is three greater than the decimal digit represented by that code-word. For example, 9 is represented by 1100, the weighted sum of which is 8×1 + 4×1 + 2×0 + 1×0 = 12.

## Gray Code:

It is a form of binary that uses a different method of incrementing from one number to the next. With Gray Code, only one bit changes state from one position to another. This feature allows a system designer to perform some error checking (i.e. if more than one bit changes, the data must be incorrect). These codes are also called cyclic codes or reflected codes.

## Converting binary to gray code:

The most significant bit (MSB) in Gray is taken directly from the MSB in binary. The rest of the Gray bits comes from anEx-OR operation between the precedent binary bit(b(i-1)) and the current binary bit (b(i)). In the case shown in the figure above:



g(1) = b(1)

g(2) = b(1) XOR b(2)

g(3) = b(2) XOR b(3)

g(4) = b(3) XOR b(4)

g(5) = b(4) XOR b(5)

\*Considering b (1) to be the MSB.

The Ex-OR operation produces a 1 if the bits are different and produces a 0 if the bits are equal. So, a binary ‘11101’ becomes a ‘10011’ in Gray.

## Binary to Gray code conversion: truth table

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Decimal | Binary | | | | Gray | | | |
|  | B3 | B2 | B1 | B0 | G3 | G2 | G1 | G0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| 3 | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 0 |
| 4 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 |
| 5 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
| 6 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| 7 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 0 |
| 8 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 |
| 11 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 12 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 |
| 13 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 14 | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 1 |
| 15 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

## K-map simplification: Binary to Gray

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **0** | **0** | **0** |
| **0** | **0** | **0** | **0** |
| **1** | **1** | **1** | **1** |
| **1** | **1** | **1** | **1** |

**G3=**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **0** | **0** | **0** |
| **1** | **1** | **1** | **1** |
| **0** | **0** | **0** | **0** |
| **1** | **1** | **1** | **1** |

**G2=**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **0** | **1** | **1** |
| **1** | **1** | **0** | **0** |
| **1** | **1** | **0** | **0** |
| **0** | **0** | **1** | **1** |

**G1=**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **1** | **0** | **1** |
| **0** | **1** | **0** | **1** |
| **0** | **1** | **0** | **1** |
| **0** | **1** | **0** | **1** |

**G0=**

## K-map simplification: Gray to Binary

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **0** | **0** | **0** |
| **0** | **0** | **0** | **0** |
| **1** | **1** | **1** | **1** |
| **1** | **1** | **1** | **1** |

**B3=**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **0** | **0** | **0** |
| **1** | **1** | **1** | **1** |
| **0** | **0** | **0** | **0** |
| **1** | **1** | **1** | **1** |

**B2=**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **0** | **1** | **1** |
| **1** | **1** | **0** | **0** |
| **0** | **0** | **1** | **1** |
| **1** | **1** | **0** | **0** |

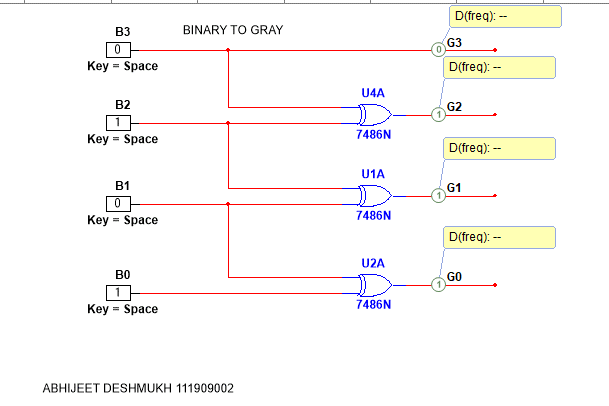
**B1=**

|  |  |  |  |
| --- | --- | --- | --- |
| **0** | **1** | **0** | **1** |
| **1** | **0** | **1** | **0** |
| **0** | **1** | **0** | **1** |
| **1** | **0** | **1** | **0** |

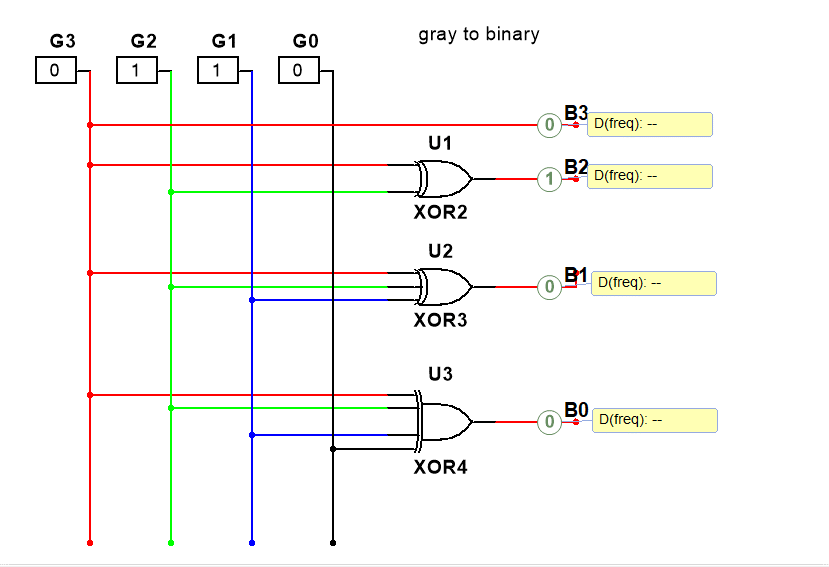
**B0=**

# Logic Diagram (Design):

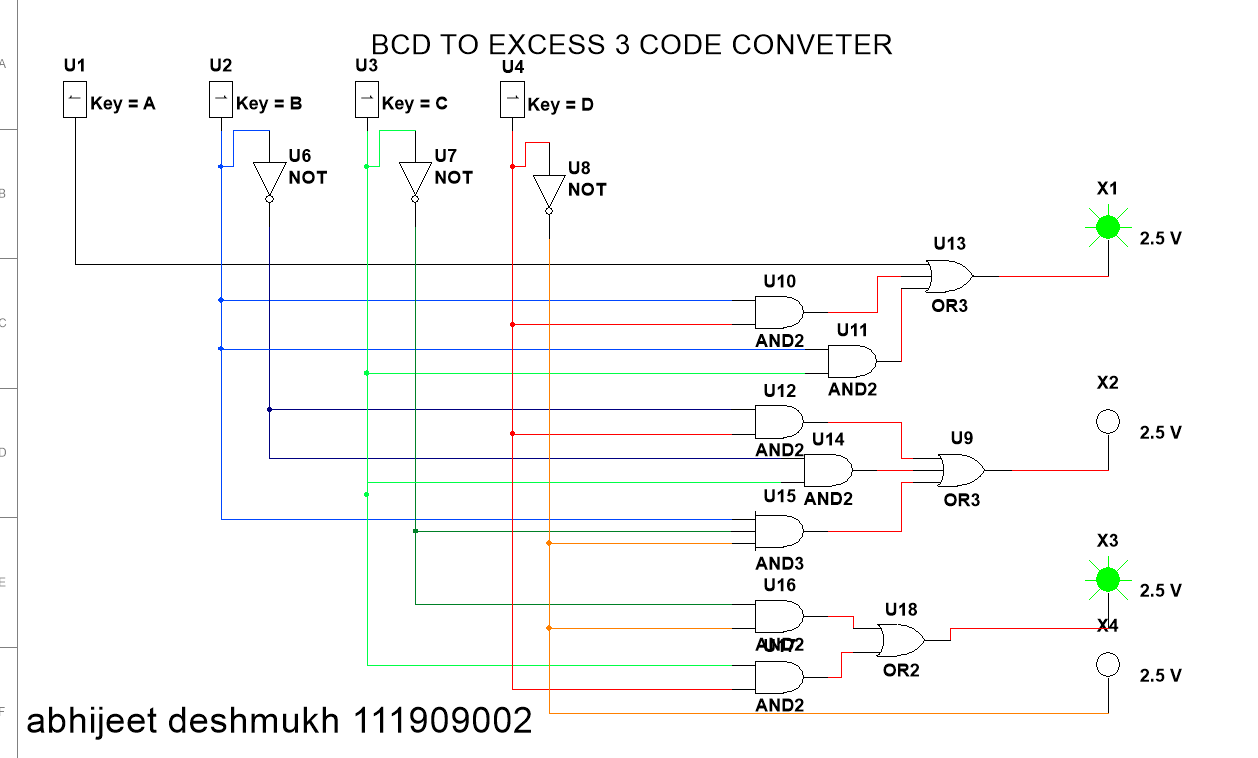
## Binary to Gray

****

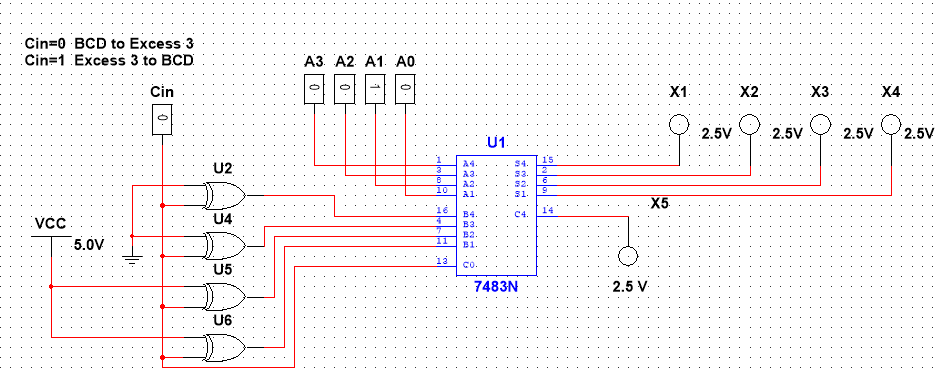
## Gray to Binary

****

## BCD to Excess 3 and vice versa



Circuit using IC 7483.

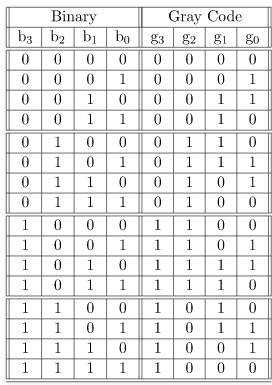


# Procedure:

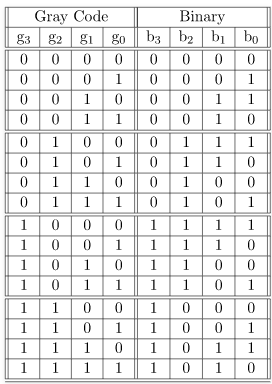
1. Connect the components as shown in the circuit diagram.
2. Give +5V supply to the IC’s.
3. From the LED observe the outputs.
4. Make a note of the truth tables in the observations.
5. Verify the truth tables accordingly.

# Observations:

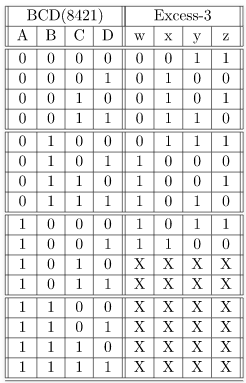
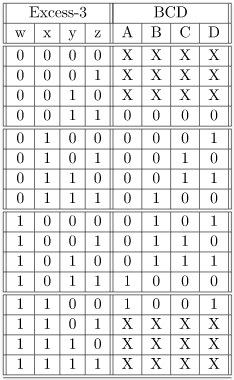
## Binary to Gray

****

## Gray to Binary

****

## BCD to Excess 3 and vice versa

** **

# Result:

Above mentioned circuits were created and run in multisim software**.**

# Conclusion:

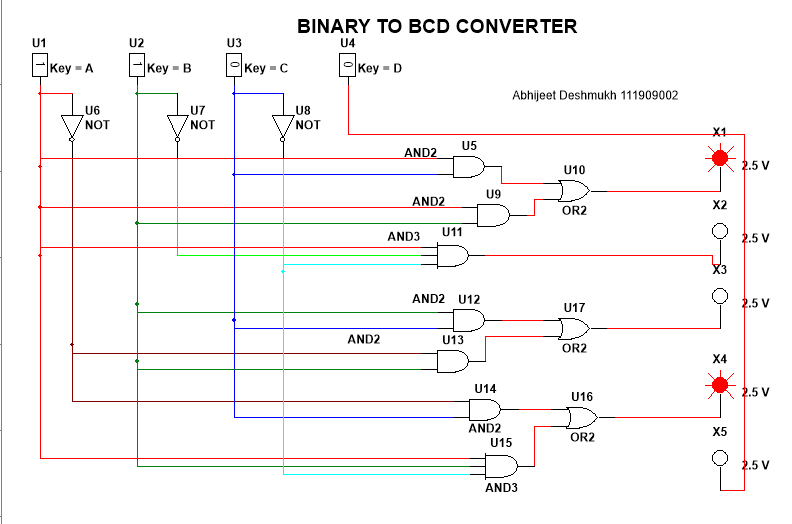
Truth tables were verified by simulating the created circuit.

# What did you learn?

I learnt how binary to gray, gray to binary, octal to binary, bcd to excess3 and excess3 to bcd code converter circuits are made, and I understood their working by simulating them and verifying the truth table. **I also learnt TTL IC 7483 .**

# Assignment:

We often use BCD rather than binary code, so as an assignment to this experiment I have done binary to BCD code convertor.



THANK YOU!