MINI PROJECT

DIGITAL DESIGN

EC4050

PROJECT INITIAL DESIGN REPORT

BY:-

GROUP H4

2021/E/171 : DULANYA W.D.A.

2021/E/172 : GUNASEKARA W.M.P

2021/E/175 : PERERA W.A.C.D.

DESIGN A DIGITAL LOGIC CIRCUIT TO DEMONSTRATE DIFFERENT BINARY CODES

Project Objectives

- To get knowledge about 8421, 5211, 2421 and excess-3 binary code.
- To gain experience in making digital circuits.
- To understand a basic idea of how our digital design subject is used in applications.
- Design a digital logic circuit to demonstrate different binary Codes.
- Change the 8421 code to,
    1. 2421
    2. 5211
    3. Excess-3 using logic circuits.

8421 BCD CODE

The 8421 BCD code represents decimal digits in binary form. BCD is a binary encoding of decimal numbers where each decimal digit is represented by a fixed number of binary bits. The name "8421" refers to the weights assigned to these bits.

The 8421 code, commonly known as Binary-Coded Decimal (BCD), employs a weighted coding scheme assigning four bits to represent each decimal digit from 0 to 9. The leftmost bit carries a weight of 8, followed by weights of 4, 2, and 1 for the remaining bits. This encoding method facilitates the direct translation of decimal numbers into their corresponding four-bit binary codes, offering a straightforward means of representation within a binary system. Moreover, BCD enables simplified arithmetic operations through the utilization of binary logic gates, contributing to convenience and efficiency in calculations involving BCD numbers.

However, BCD exhibits a limitation when compared to pure binary representation, particularly in expressing larger numbers. For instance, the seemingly uncomplicated decimal number 100 necessitates eight bits in BCD (0100 1000), whereas its pure binary counterpart can be represented in just seven bits. Despite its efficiency drawbacks, BCD continues to be a favored choice for specific applications, particularly in contexts where human comprehensibility is of utmost importance, such as displays and input/output devices.

The key advantage lies in the ease with which BCD numbers can be directly displayed on seven-segment displays and entered from keyboards without the need for additional conversions. This user-friendly feature makes BCD a practical and widely adopted choice in scenarios where human interaction and readability take precedence over the potential space inefficiencies associated with larger numerical representations.

In the 8421 BCD code, each decimal digit is represented by a 4-bit binary code, and the weights for each bit are as follows:

- 8: The most significant bit (MSB)
- 4: The second most significant bit
- 2: The third most significant bit
- 1: The least significant bit (LSB)

So, the name "8421" comes from the weights 8, 4, 2, and 1. Each decimal digit is represented by a unique combination of these weights. The BCD representation for decimal digits 0 through 9 is as follows:

Decimal number	8421 BCD code			
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

Working principle:

The working principle of the 8421 BCD code involves assigning binary weights to each decimal digit in a way that allows easy conversion between binary and decimal representations. The name "8421" comes from the weights assigned to the four bits representing each decimal digit: 8, 4, 2, and 1.

- 8: The most significant bit (MSB)
- 4: The second most significant bit
- 2: The third most significant bit
- 1: The least significant bit (LSB)

the working principle of the 8421 BCD code:

1. Decimal to BCD Conversion:

- To represent a decimal digit in BCD, the decimal digit is divided into its tens and unit places.
- Each place (tens and units) is then separately converted to its 4-bit binary representation using the weights 8, 4, 2, and 1.
- The BCD representation is obtained by concatenating the binary representations of the tens and units places.

2. BCD to Decimal Conversion:

- To convert a BCD number back to decimal, each group of four bits is considered separately.
- The weights 8, 4, 2, and 1 are assigned to each bit in the group.
- The decimal value is obtained by summing the products of the bit values and their corresponding weights.

3. Arithmetic Operations:

- BCD arithmetic operations (addition, subtraction, etc.) are performed similarly to binary arithmetic, but with additional steps to handle carry between decimal digits.
- In addition, if the sum in a digit is greater than 9, a "carry" is generated and added to the next higher-order digit.
- The 8421 BCD code simplifies the process of converting between decimal and binary representations, making it particularly useful in digital systems where decimal numbers are manipulated and displayed.

<u>Advantages of BCD</u>

- Easy for People to Read: BCD is designed to be readable for us, making it great for things like displays and devices where humans interact directly.
- No Fuss Display: BCD numbers can go straight onto seven-segment displays without any extra steps. This makes it simple to show BCD numbers on things like digital clocks or calculators.
- Simple Math: BCD makes math easier when you're working with decimal numbers in a digital world, thanks to its straightforward use of binary logic gates.

Disadvantages of BCD

- Wastes a Bit: For bigger numbers, BCD might use more bits than if we just spoke in pure binary. This can be inefficient for storage and data transfer.
- Limits: BCD has its limits compared to pure binary, especially when dealing with larger numbers. It might not be the best choice for really big numerical values.
- Tricky Conversions: While it's easy to turn BCD into regular numbers us, going back and forth between BCD and pure binary can be a bit tricky. It involves some bit of juggling and grouping.

So, BCD is like a special language that computers use when they want to talk about numbers, especially when humans are in a conversation. It's good for showing numbers on displays and doing math, but it's not always the most efficient when we're dealing with really big numbers.

Applications:

The 8421 BCD code finds applications in various digital systems and electronic devices where decimal representation is important.

1. Calculators:

BCD is commonly used in calculators to represent and perform arithmetic operations on decimal numbers. The 8421 code simplifies the conversion between binary and decimal representations, making it suitable for calculator applications.

2. Digital Displays:

8421 BCD is frequently used in digital displays, such as seven-segment displays, where each decimal digit is represented by a combination of 4 bits. This is common in devices like digital clocks, calculators, and other numerical displays.

3. Data Input Devices:

In systems where decimal data is input using binary devices, BCD is used to represent and process the input. For example, numeric keypads on keyboards or other data input devices may utilize BCD encoding.

4. Control Systems:
   In certain control systems, where numeric values need to be processed and displayed, the 8421 BCD code can be employed. This is particularly useful in applications where human-readable numeric information is essential.

5. Timekeeping Systems:

Digital clocks and timers often use the 8421 BCD code to represent and process time information. Each digit of the time (hours, minutes, and seconds) can be encoded separately.

6. Digital Signal Processors (DSPs):

DSPs that deal with signal processing in applications like audio processing, where decimal representation is common, may use BCD internally or in communication with external devices.

7. Communication Protocols:

In certain communication protocols, BCD encoding can be used to transmit decimal information efficiently. This is seen in applications where serial communication involves the exchange of numeric data.

8. Embedded Systems:

In various embedded systems, especially those involving human-machine interfaces or numerical data processing, the 8421 BCD code can be a suitable representation.

2421 BCD CODE

The 2421 BCD code represents decimal digits in a binary form. The 2421 code uses weights of 2, 4, 2, and 1 for its four bits, distinguishing it from the 8421 code, which uses weights of 8, 4, 2, and 1.

In the 2421 BCD code, the name "2421" comes from the binary weights assigned to each bit:

- 2: The most significant bit (MSB)
- 4: The second most significant bit
- 2: The third most significant bit
- 1: The least significant bit (LSB)

The breakdown of the 2421 BCD representation for decimal digits 0 through 9:

Decimal number	2421 code			
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	1	0	1	1
6	1	1	0	0
7	1	1	0	1
8	1	1	1	0
9	1	1	1	1

Comparison with 8421 BCD:

The primary difference between the 2421 and 8421 BCD codes is in the weights assigned to the bits. While the 8421 code uses weights of 8, 4, 2, and 1, the 2421 code uses weights of 2, 4, 2, and 1. This difference affects the way decimal digits are encoded in binary form.

Both the 2421 and 8421 BCD codes aim to represent decimal digits in a binary format, but they differ in their binary weights and, consequently, in the bit patterns used for encoding.

Working principle:

The working principle of a 2421 BCD code involves the representation of decimal digits using four bits with specific binary weights: 2, 4, 2, and 1. The code aims to provide a binary representation that simplifies the conversion between decimal and binary forms.

- 2: The most significant bit (MSB)
- 4: The second most significant bit
- 2: The third most significant bit
- 1: The least significant bit (LSB)

The working principle:

1. Binary Weights:

- Each of the four bits in the 2421 BCD code is assigned a specific weight: 2, 4, 2, and 1 from the most significant bit (MSB) to the least significant bit (LSB).

2. Decimal to 2421 BCD Conversion:

- To represent a decimal digit in the 2421 BCD code, the decimal digit is divided into its tens and unit places.
- Each place is then separately converted to its 4-bit binary representation using the weights 2, 4, 2, and 1.
- The BCD representation is obtained by concatenating the binary representations of the tens and units places.

3. 2421 BCD to Decimal Conversion:

- To convert a 2421 BCD number back to decimal, each group of four bits (representing a decimal digit) is considered separately.
- The weights 2, 4, 2, and 1 are assigned to each bit in the group.
- The decimal value is obtained by summing the products of the bit values and their corresponding weights.

4. Arithmetic Operations:

- Arithmetic operations (addition, subtraction, etc.) in 2421 BCD are performed in a manner similar to binary arithmetic.
- If the sum in a digit is greater than 9, a "carry" is generated and added to the next higher-order digit.

Advantages of 2421 Code

- Built-In Error Checker: The self-complementing property makes the 2421 code great for catching errors in digital systems. The code itself can serve as a built-in error checker, ensuring the accuracy of the data.
- Reliable Error Detection: The ability to easily determine the 9's complement through the self-complementing property enhances the reliability of error detection in digital circuits.

Disadvantages of 2421 Code

- Not the Best for Big Numbers: Like some other BCD codes, the 2421 code might not be the most efficient when dealing with larger numbers. It could end up using more bits compared to other coding schemes.
- Less Common: The 2421 code isn't as widely used as the more popular 8421 BCD. Its relative inefficiency for larger numbers means it has a more specific use case, and you won't find it everywhere.

K-Map

For W

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	1	1	1
11	X	X	X	X
10	1	1	X	X

For X

AB \ CD	00	01	11	10
00	0	0	0	0
01	1	0	1	1
11	X	X	X	X
10	1	1	X	X

For Y

AB \ CD	00	01	11	10
00	0	0	1	1
01	0	1	0	0
11	X	X	X	X
10	1	1	X	X

For Z

AB \ CD	00	01	11	10
00	0	1	1	0
01	0	1	1	0
11	X	X	X	X
10	0	1	X	X

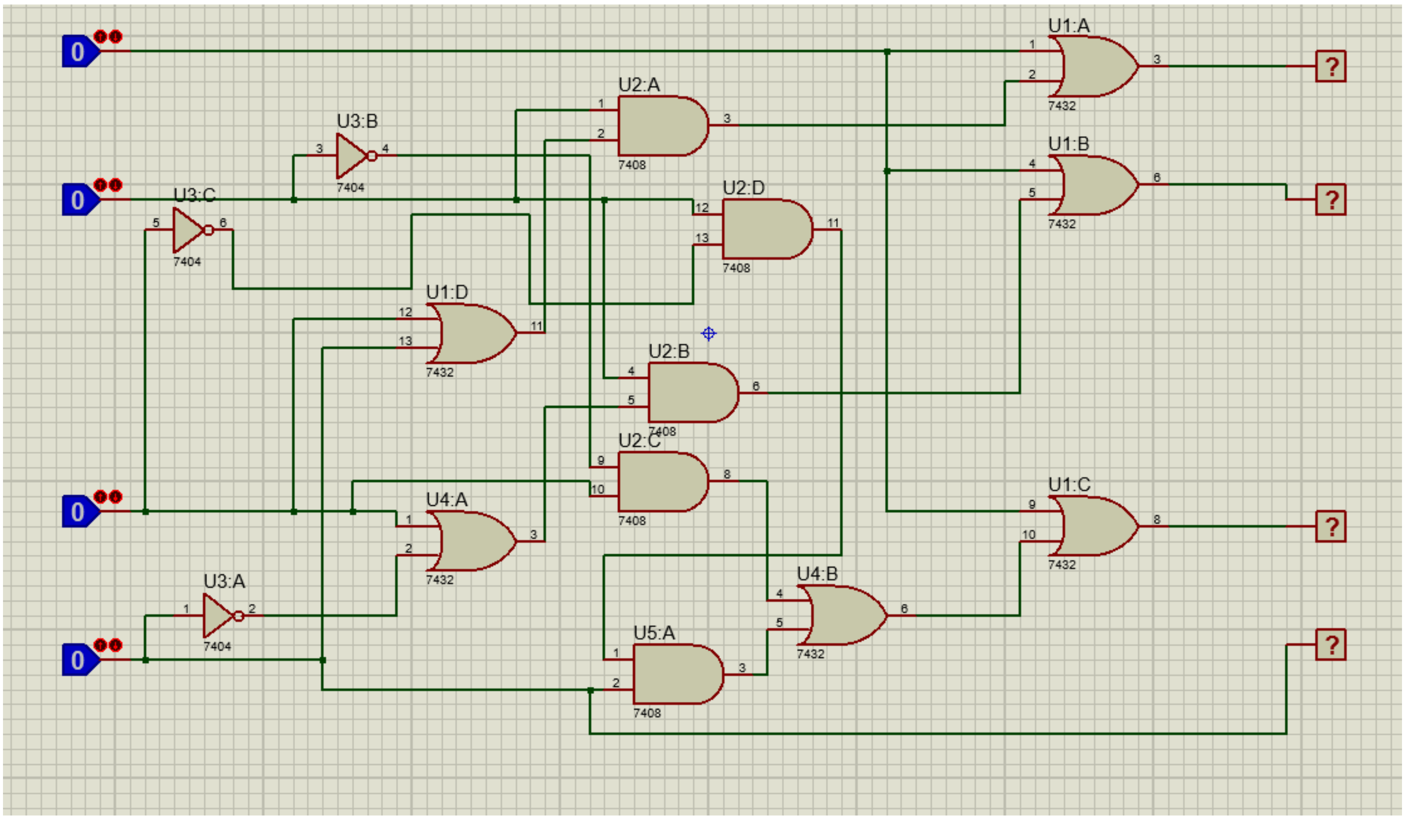
W= A+B(C+D)

X= A+B(C+D')

Y= A+BC'D+ B'C

Z= D

Logic Diagram



Applications:

The 2421 BCD code is used in various applications where decimal representation is required, and a binary format is employed for internal processing or communication.

1. Digital Displays:

2421 BCD is used in digital displays, especially in seven-segment displays, to represent decimal digits. Devices like digital clocks, calculators, and numeric displays may utilize the 2421 BCD code.

2. Calculators:

BCD is commonly used in calculators to represent and perform arithmetic operations on decimal numbers. The 2421 code simplifies the conversion between binary and decimal representations.

3. Data Input Devices:

In systems where decimal data is input using binary devices, the 2421 BCD code can be used to represent and process the input. This is applicable in numeric keypads and other input devices.

4. Timekeeping Systems:

Similar to the 8421 BCD code, the 2421 BCD code is used in digital clocks, timers, and other timekeeping systems where decimal time needs to be represented and processed.

5. Communication Protocols:

In certain communication protocols, the 2421 BCD code can be employed to transmit decimal information efficiently. This is especially useful in applications where serial communication involves the exchange of numeric data.

6. Embedded Systems:

In various embedded systems, especially those involving human-machine interfaces or numerical data processing, the 2421 BCD code may be utilized.

7. Control Systems:

In certain control systems, where numeric values need to be processed and displayed, the 2421 BCD code can be used. This is relevant in applications where human-readable numeric information is essential.

8. Consumer Electronics:

Devices such as digital watches, digital thermometers, and other consumer electronics that involve numeric displays may use the 2421 BCD code for representing decimal information.

5211 code

The 5211 code, also referred to as a weighted code, is a form of binary-coded decimal (BCD) that represents decimal digits from 0 to 9 using a four-bit binary format. The weights assigned to the bits are arranged from left to right as 5, 2, 1, and 1, corresponding to the powers of 2 raised to the 2nd, 1st, 0th, and 0th positions, respectively.

While the 5211 code isn't as commonly employed as the more prevalent 8421 BCD or 2421 code, it possesses intriguing properties. Similar to the 2421 code, it is self-complementing, and additionally, it features a sequential code nature. This sequential property implies that two consecutive codes, when interpreted as binary numbers, differ by only one bit. This unique attribute enhances its utility in error detection and correction within digital circuits.

Despite its lesser popularity, the 5211 code's self-complementing and sequential characteristics make it valuable in specific applications where error detection and correction are priorities.

Working Principle of 5211 Code

The 5211 code, a variant of binary-coded decimal (BCD), operates on the principle of representing decimal digits from 0 to 9 using a four-bit binary format. In this coding scheme, each bit is assigned a specific weight: 5, 2, 1, and 1, corresponding to the powers of 2 raised to the 2nd, 1st, 0th, and 0th positions, respectively. The binary representation of each decimal digit is determined by the combination of these weighted bits.

Advantages of 5211 Code

- Self-Complementing Property: Similar to other BCD codes like 2421, the 5211 code is self-complementing. The one's complement of a 5211 code is equivalent to the 9's complement of the corresponding decimal digit. This self-complementing nature can be advantageous in certain arithmetic and error detection scenarios.
- Sequential Code Nature: The 5211 code possesses a sequential code property, meaning that two consecutive codes, when interpreted as binary numbers, differ by only one bit. This sequential characteristic enhances its utility for error detection and correction in digital circuits.

Disadvantages of 5211 Code

- Limited Adoption: The 5211 code is not as widely adopted as other BCD codes like the 8421 or 2421. Its use is relatively niche, and it may not be as prevalent in modern digital systems.
- Efficiency for Larger Numbers: Similar to other BCD codes, the 5211 code may be less efficient when representing larger numbers. It might require more bits compared to alternative coding schemes for expressing numerical values beyond the single-digit range.

Truth Table for- 5211 Code

Decimal	5211 Code
0	0000
1	0001
2	0011
3	0101
4	0111
5	1000
6	1010
7	1100
8	1110
9	1111

K-Map

For W

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	1	1	1
11	X	X	X	X
10	1	1	X	X

For X

AB \ CD	00	01	11	10
00	0	0	1	0
01	1	0	1	0
11	X	X	X	X
10	1	1	X	X

For Z

AB \ CD	00	01	11	10
00	0	1	1	1
01	1	0	0	0
11	X	X	X	X
10	0	1	X	X

For Y

AB \ CD	00	01	11	10
00	0	0	0	1
01	1	0	0	1
11	X	X	X	X
10	1	1	X	X

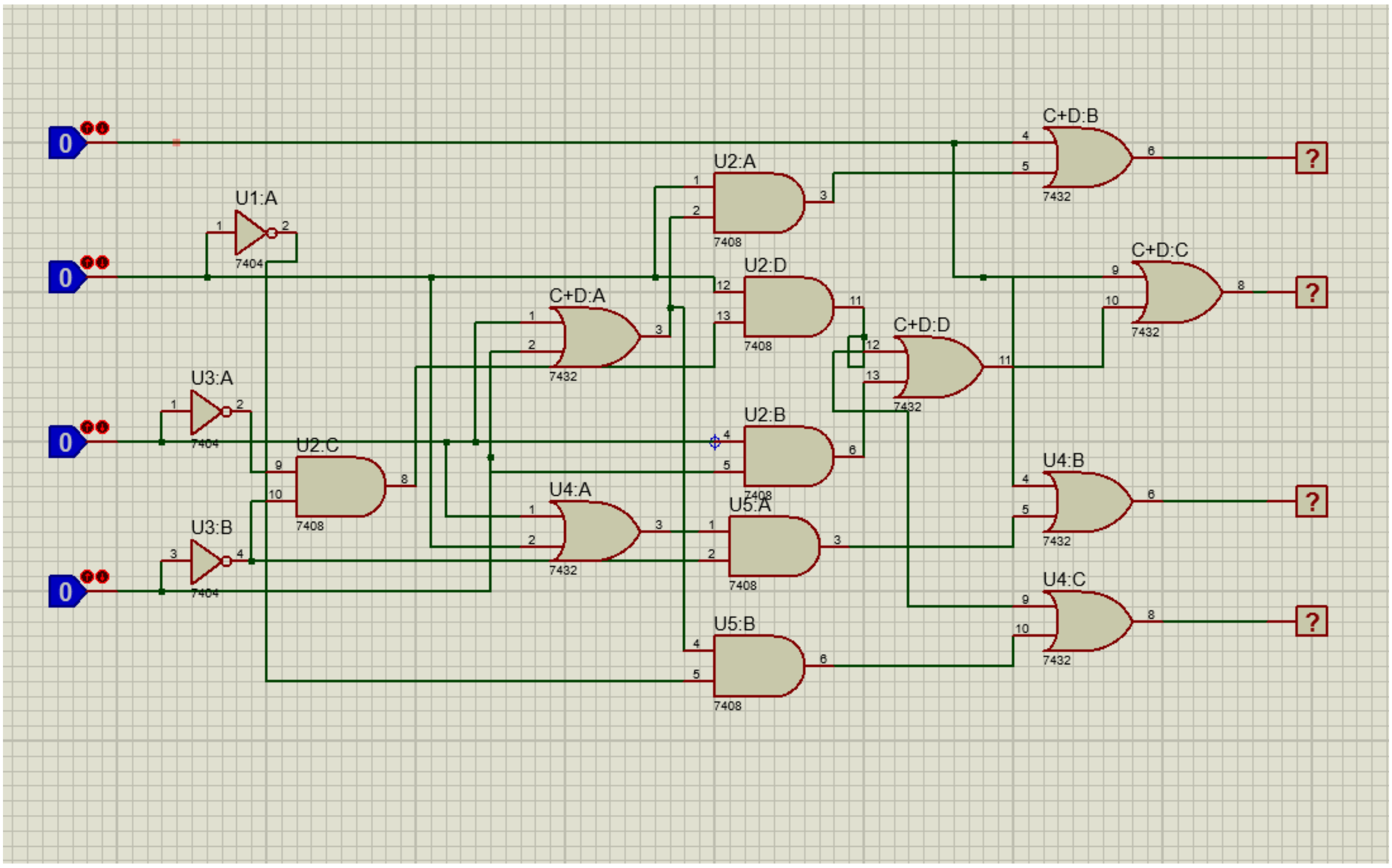
W= A+B(C+D)

X= A+CD+BC'D'

Y= A+D'(C+B)

Z= BC'D'+B'(C+D)

Logical Diagram



Excess-3 CODE

Excess-3 is a binary-coded decimal (BCD) code used to represent decimal digits in a binary form. It is called "excess-3" because it is three units greater than the corresponding pure binary-coded decimal.

Representation:

In Excess-3 code, each decimal digit is represented by a 4-bit binary code. The binary value for each decimal digit is obtained by adding 3 to its corresponding 4-bit binary representation. This results in a unique code for each decimal digit.

Decimal number	2421 code			
0	0	0	1	1
1	0	1	0	0
2	0	1	0	1
3	0	1	1	0
4	0	1	1	1
5	1	0	0	0
6	1	0	0	1
7	1	0	1	0
8	1	0	1	1
9	1	1	0	0

<u>Working principle:</u>

The working principle of Excess-3 encoding involves representing decimal digits in binary form by adding an excess of 3 to their corresponding 4-bit Binary-Coded Decimal (BCD) representations.

The working principle:

1. BCD to Excess-3 Conversion:

- Start with a BCD representation of a decimal digit. BCD uses a 4-bit code to represent each decimal digit.
- To obtain the Excess-3 code, add 3 to each of the 4-bit binary groups of the BCD representation.

2. Decimal-to-Excess-3 Conversion:

- To convert a decimal digit to Excess-3, first convert the decimal digit to its BCD representation.
- Add 3 to each 4-bit group of the BCD representation to obtain the corresponding Excess-3 code.

3. Arithmetic Operations:

- Excess-3 simplifies binary-coded decimal arithmetic. Arithmetic operations, such as addition and subtraction, can be performed directly in Excess-3. The result can then be converted back to BCD if necessary.

4. Error Detection:

- Due to its unique properties, Excess-3 can be used for error detection in digital systems. Since each Excess-3 code corresponds to a unique decimal digit, errors in transmission or processing can be detected by checking if the received Excess-3 code corresponds to a valid decimal digit.

Applications:

1. Decimal-to-BCD Conversion: Excess-3 is often used in applications where decimal input needs to be converted to BCD. By subtracting 3 from each digit in Excess-3, you obtain the corresponding BCD representation.

2. Arithmetic Operations: Excess-3 simplifies binary-coded decimal arithmetic. Operations such as addition and subtraction can be performed directly in Excess-3, and the result can then be converted back to BCD.

3. Error Detection: Due to its unique properties, Excess-3 can be used for error detection in digital systems.

4. Automation and Control Systems: In certain applications, Excess-3 is used for numeric representation, especially when interfacing with devices that use BCD.

5. Excess-3 is part of the family of Gray codes, which are binary codes with specific properties that make them useful in various applications, including error detection and correction.

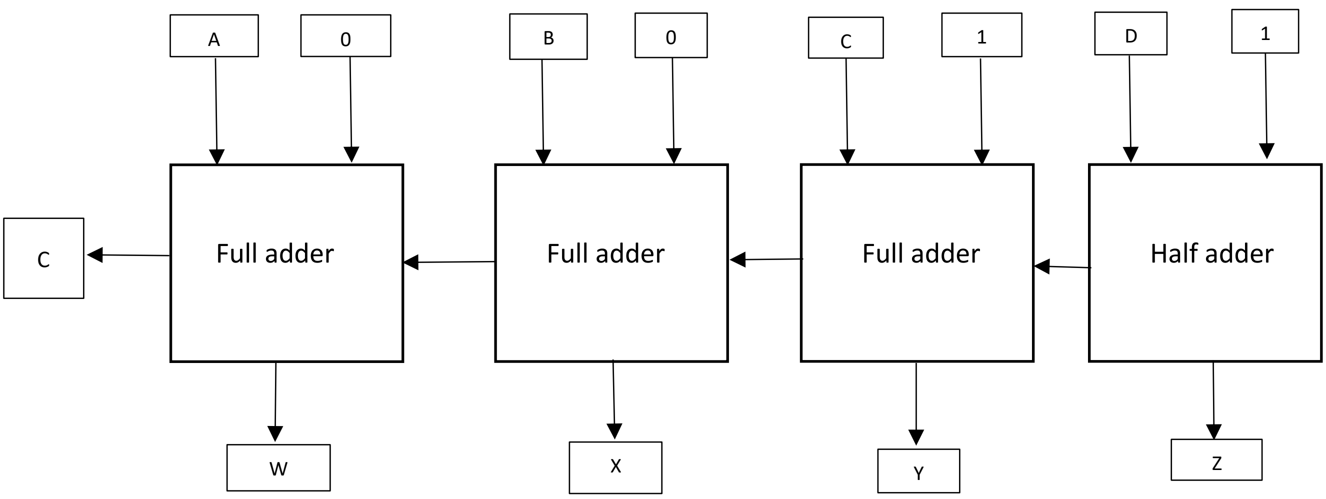
Advantages of Excess-3 Code

- Self-Complementary Nature: One of the significant advantages of Excess-3 code is its self-complementary property. The complement of an Excess-3 code is its own inverse. This property simplifies arithmetic operations, especially in scenarios where finding the complement or inverse is beneficial.
- Carry-Free Addition: Excess-3 code eliminates the need for carry propagation when adding two decimal digits whose sum exceeds 9. In standard BCD, adding digits greater than 9 requires carry generation and propagation, a complex and time-consuming process. In the Excess-3 code, adding two digits simply results in a larger Excess-3 code, and no carry is required.

Disadvantages of Excess-3 Code

- Limited Efficiency for Large Numbers: While Excess-3 code excels in simplifying arithmetic operations, it may not be the most efficient for representing larger numerical values. The code's bias of adding 3 to each digit can result in larger bit sequences compared to other BCD codes, impacting space efficiency.
- Historical Obsolescence: Excess-3 code, once prevalent in older computers, cash registers, and calculators during the 1970s, has gradually fallen out of widespread use. The rise of pure binary representation and other coding schemes has contributed to its historical obsolescence.

Block Diagram



K-Map

For W

AB \ CD	00	01	11	10
00	0	0	0	0
01	0	0	1	1
11	X	X	X	X
10	1	1	X	X

For X

AB \ CD	00	01	11	10
00	0	1	1	1
01	1	0	0	0
11	X	X	X	X
10	0	1	X	X

For Z

AB \ CD	00	01	11	10
00	1	0	1	0
01	1	0	1	0
11	X	X	X	X
10	1	0	X	X

For Y

AB \ CD	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	X	X	X	X
10	1	0	X	X

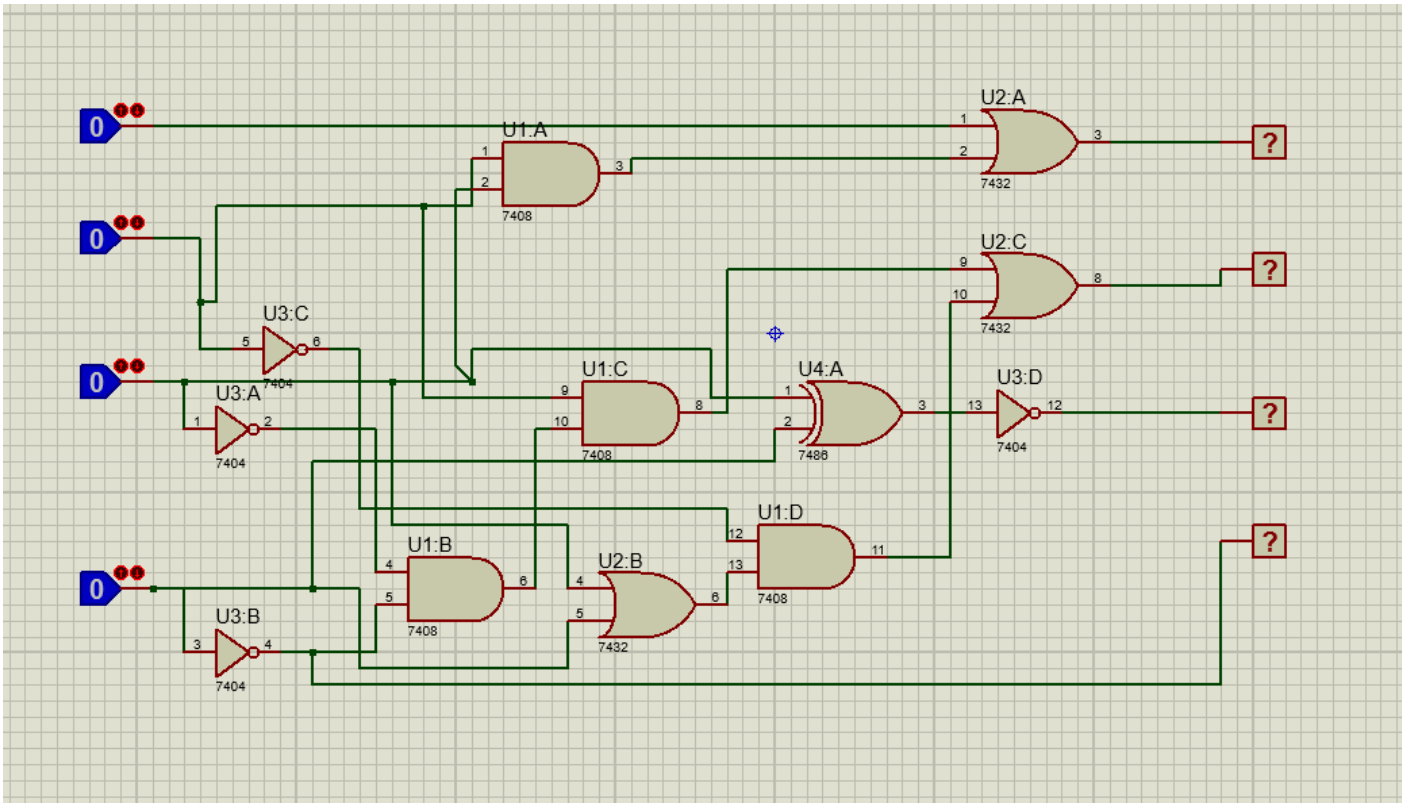
W= A+BC

X= BC'D'+B'(D+C)

Y= $\overline{(C \oplus D)}$

Z= D'

<u>Logical Diagram</u>



References:

"Digital Design" by M. Morris Mano and Michael D. Ciletti: This textbook covers digital design principles, including binary codes and number systems.

https://www.naics.com/sic-industry-description/?code=2421

https://www.tutorialspoint.com/digital_circuits/digital_circuits_codes.htm

"Digital Electronics: Principles and Applications" by Roger L. Tokheim: This textbook is focused on digital electronics and covers topics like binary systems, codes, and digital circuits.

https://www.electronicshub.org/binary-codes/

Group Members:

Group H4

2021/E/171 : DULANYA W.D.A.

2021/E/172 : GUNASEKARA W.M.P

2021/E/175 : PERERA W.A.C.D.