Excess-3 Code

The excess-3 code is also treated as **XS-3 code**. The excess-3 code is a non-weighted and self-complementary BCD code used to represent the decimal numbers. This code has a biased representation. This code plays an important role in arithmetic operations because it resolves deficiencies encountered when we use the 8421 BCD code for adding two decimal digits whose sum is greater than 9. The Excess-3 code uses a special type of algorithm, which differs from the binary positional number system or normal non-biased BCD.

We can easily get an excess-3 code of a decimal number by simply adding 3 to each decimal digit. And then we write the 4-bit binary number for each digit of the decimal number. We can find the excess-3 code of the given binary number by using the following steps:

1. We find the decimal number of the given binary number.
2. Then we add 3 in each digit of the decimal number.
3. Now, we find the binary code of each digit of the newly generated decimal number.

We can also add 0011 in each 4-bit BCD code of the decimal number for getting excess-3 code.

The Excess-3 code for the decimal number is as follows:

Decimal Digit	BCD Code	Excess-3 Code
0	0000	0011
1	0001	0100
2	0010	0101
3	0011	0110
4	0100	0111
5	0101	1000
6	0110	1001

7	0111	1010
8	1000	1011
9	1001	1100

In excess-3 code, the codes 1111 and 0000 are never used for any decimal digit. Let's take some examples of Excess-3 code.

Example 1: Decimal number 31

1. We find the BCD code of each digit of the decimal number.

Digit	BCD
3	0011
1	0001

2) Then, we add 0011 in both of the BCD code.

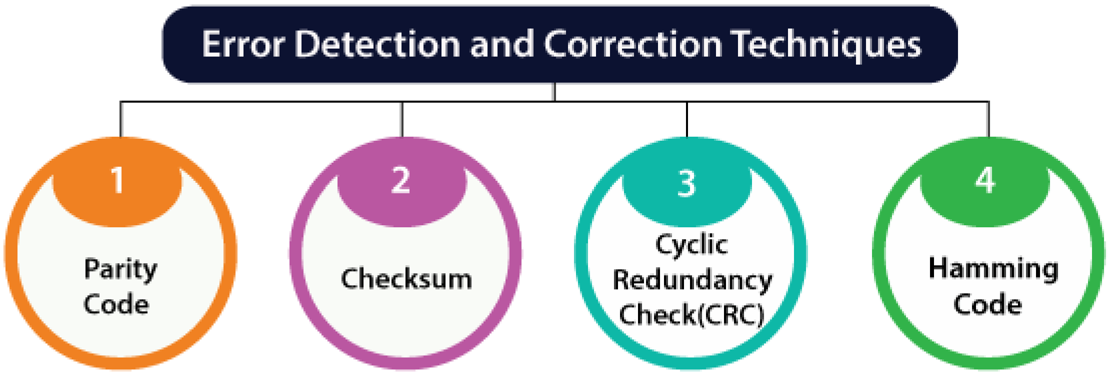
Decimal	BCD	Excess-3
3	0011+0011	0110
1	0001+0011	0100

4. So, the excess-3 code of the decimal number 31 is **0110 0100**

Error Detection and Correction Code

Error detection and correction code plays an important role in the transmission of data from one source to another. The noise also gets added into the data when it transmits from one system to another, which causes errors in the received binary data at other systems. The bits of the data may change(either 0 to 1 or 1 to 0) during transmission.

It is impossible to avoid the interference of noise, but it is possible to get back the original data. For this purpose, we first need to detect either an error **z** is present or not using error detection codes. If the error is present in the code, then we will correct it with the help of error correction codes.


Error Detection and Correction Techniques
1
Parity Code
2
Checksum
3
Cyclic Redundancy Check(CRC)
4
Hamming Code

Error detection code

The error detection codes are the code used for detecting the error in the received data **bitstream**. In these codes, some bits are included appended to the original bitstream.

Error detecting codes encode the message before sending it over the noisy channels. The encoding scheme is performed in such a way that the decoder at the receiving can find the errors easily in the receiving data with a higher chance of success.

Parity Code

In parity code, we add one parity bit either to the right of the LSB or left to the MSB to the original bitstream. On the basis of the type of parity being chosen, two types of parity codes are possible, i.e., even parity code and odd parity code.

Features of Error detection codes

These are the following features of error detection codes:

- These codes are used when we use message backward error correction techniques for reliable data transmission. A feedback message is sent by the receiver to inform the sender whether the message is received without any error or not at the receiver side. If the message contains errors, the sender retransmits the message.
- In error detection codes, in fixed-size blocks of bits, the message is contained. In this, the redundant bits are added for correcting and detecting errors.
- These codes involve checking of the error. No matter how many error bits are there and the type of error.
- Parity check, Checksum, and CRC are the error detection technique.