# ERROR DETECTION

**Experiment No : 6 DATE:-18-03-2024**

**Aim**: To implement error detection mechanism using.

1. Checksum
2. CRC

# Theory:

Error is a condition when the receiver’s information does not match the sender’s information. During transmission, digital signals suffer from noise that can introduce errors in the binary bits traveling from sender to receiver. That means a 0 bit may change to 1 or a 1 bit may change to 0.

Data may get scrambled by noise or get corrupted whenever a message is transmitted. To prevent such errors, error-detection codes are added as extra data to digital messages. This helps in detecting any errors that may have occurred during message transmission.

Various techniques for error detection include:

* 1. Checksum

Checksum is a simple error detection method used to validate the integrity of data packets. It involves summing up the bytes in a data packet and appending this sum to the packet. Upon reception, the checksum is recalculated, and if the recalculated checksum matches the transmitted checksum, the data is assumed to be error-free. However, checksums have limitations; they may fail to detect certain types of errors, such as transpositions.

Data : 1011 and 0100

Checksum : 0000

* 1. Cyclic Redundancy Check (CRC)

CRC is a more sophisticated error detection technique compared to checksum. It involves generating a polynomial code based on the data packet and appending it to the packet before transmission. At the receiver's end, the same polynomial is applied to the received data, and if the generated code matches the appended one, the data is considered intact. CRC offers a higher degree of error detection capability compared to checksums, making it widely used in various communication protocols like Ethernet and Wi-Fi.

data to be sent : 1010001011 Generator : 11101

CRC code : 10100010111000

**Code:**

1. Checksum

#include <iostream>

#include <string>

#include <cmath>

using namespace std;

int To\_Decimal(string data)

{

int n = data.length(); int num = 0;

for (int i = 0; i < n; i++)

{

num = num \* 2 + (data[i] - '0');

}

return num;

}

string To\_Binary(int num, int data\_len)

{

string data = ""; while (num > 0)

{

data = to\_string(num % 2) + data; num = num / 2;

}

int n = data.length(); if (n < data\_len)

{

data = string(data\_len - n, '0') + data;

}

return data;

}

string complement(string data)

{

int n = data.length();

for (int i = 0; i < n; i++)

{

if (data[i] == '0')

{

data[i] = '1';}else

{

data[i] = '0';

}

}

return data;

}

void errorcheck(string data1, string data2, string checksum)

{

int Final = To\_Decimal(data1) + To\_Decimal(data2) + To\_Decimal(checksum);

if (Final > pow(2, data1.length()) - 1)

{

Final = Final - (pow(2, data1.length()) - 1);

}

string To\_Send = To\_Binary(Final, data1.length()); To\_Send = complement(To\_Send);

for (int i = 0; i < To\_Send.length(); i++)

{

if (To\_Send[i] != '0')

{

cout << "\nError in data\n"; return;

}

}

cout << "\nNo error in data\n";

}

int main(void)

{

string data1, data2;

int data\_len = data1.length();

cout << "Enter data 1 : "; cin >> data1;

cout << "Enter data 2 : "; cin >> data2;

int num1 = To\_Decimal(data1); int num2 = To\_Decimal(data2);int sum = num1 + num2;

if (sum > pow(2, data\_len) - 1)

{

sum = sum - (pow(2, data\_len) - 1);

}

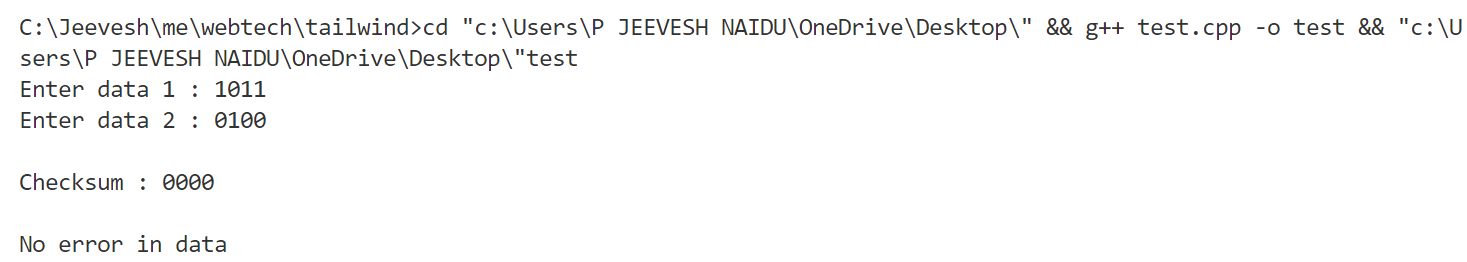
string data = To\_Binary(sum, data\_len); string checksum = complement(data);

cout << "\nChecksum : " << checksum << endl;

errorcheck(data1, data2, checksum); return 0;

}

**Output:**

****

1. CRC

#include<iostream>

using namespace std;

string XOR(string a , string b)

{

string result = ""; int n = b.length();

for(int i =1;i<n;i++)

{

if(a[i] == b[i])

{

result += "0";

}

else

{

result += "1";

}}

return result;

}

void division(string data , string gen ,int data\_len , int gen\_len)

{

int len = gen\_len;

string temp = data.substr(0,len); int n = data\_len;

while(len < n)

{

if(temp[0] == '1')

{

temp = XOR(gen , temp) + data[len];

}

else

{

temp = XOR(string(len , '0') ,temp) + data[len];

}

len += 1;

}

if(temp[0] == '1')

{

temp = XOR(gen , temp);

}

else

{

temp = XOR(string(len , '0') , temp);

}

cout<<"\n\nRemainder is : "<<temp; int new\_len = data\_len - gen\_len + 1; string new\_data= data;

for(int i =new\_len;i<data\_len;i++)

{

new\_data[i] = temp[i - new\_len];

}

cout<<"\n\nData to be sent is : "<<new\_data; cout<<endl;

}

int main(void)

{

string data , generator;cout<<"Enter data to be sent : "; cin >> data;

cout<<"Enter Generator : "; cin>>generator;

int gen\_len = generator.length(); for(int i = 0;i< gen\_len -1 ;i++)

{

data += '0';

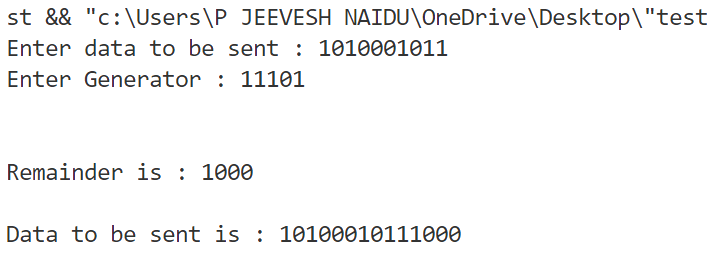
}

int data\_len = data.length();

division(data , generator , data\_len , gen\_len);

}

**Output:**



**Conclusion:**

Studied different types of Error Detection Methods with successful execution of programs.