Green University of Bangladesh

Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering

Semester: (Spring, Year: 2021), B.Sc. in CSE (Day)

Lab Report No: 02

Course Title: Data Communication Lab

Course Code: CSE 308 **Section:** PC-DD

Lab Experiment Name: Implementing CRC using X-NOR and Hamming code using Odd parity.

Student Details

Name	Id
Md. Romzan Alom	201902144

Lab Date: 02.07. 2022

Submission Date: 18.07.2022

Course Teacher's Name: Md. Nazmus Shakib

[For Teachers use only: Don't Write Anything inside this box]

Lab Report Status	
Marks: …………………………………	Signature:....................
Comments:..	Date:.............................

Title of the Lab Experiment: Implementing CRC using X-NOR and Hamming code using Odd parity.

Objectives / Aim:

We learn about CRC and Hamming Algorithm from this experiment. We can take user input as a string and we can perform various operations on this input and show the result as output.

Introduction:

An error detection technique using a polynomial to generate a series of two 8-bit block check characters that represent the entire block of data. These block check characters are incorporated into the transmission frame and then checked at the receiving end. And, hamming code is a set of error-correction code s that can be used to detect and correct bit errors that can occur when computer data is moved or stored. Hamming code is named for R. W. Hamming of Bell Labs.

Problem:

1. Implement CRC by using X-NOR in binary division for any given data. Choose G=5 bit. Show the encoded message as output.

2. Implement Hamming Code using Odd parity scheme for the message no less than 8bit.

Problem analysis:

1st:

In digital communication system errors are transferred from one communication system to another, along with the data. If these errors are not detected and corrected, data will be lost. For effective communication, data should be transferred with high accuracy. This can be achieved by first detecting the errors and then correcting them. Error detection is the process of detecting the errors that are present in the data transmitted from transmitter to receiver, in a communication system. We use some redundancy codes to detect these errors, by adding to the data while it is transmitted from source (transmitter). These codes are called "Error detecting codes". Types of error detection we will study today: • Parity Checking • Cyclic Redundancy Check (CRC).

2nd:

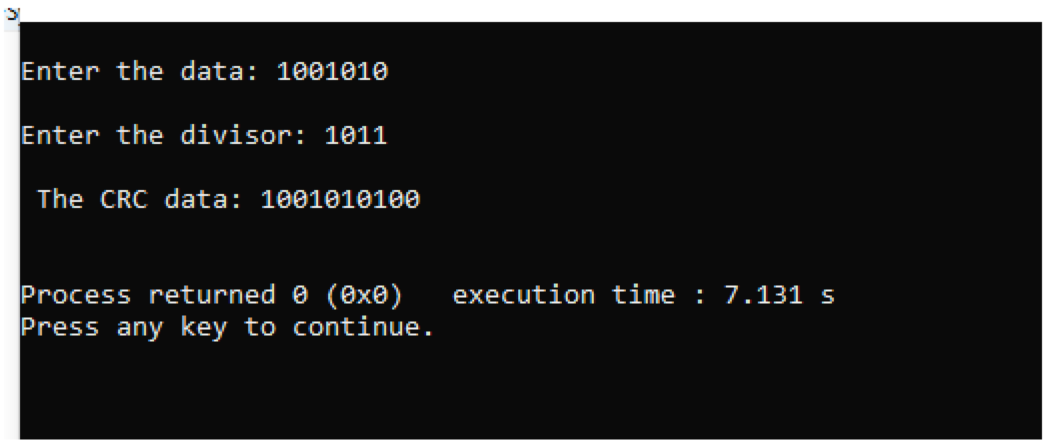
Hamming code is a set of error-correction codes that can be used to detect and correct the errors that can occur when the data is moved or stored from the sender to the receiver. It is technique developed by R.W. Hamming for error correction. Redundant bits are extra binary bits that are generated and added to the information-carrying bits of data transfer to ensure that no bits were lost during the data transfer. The number of redundant bits can be calculated using the following formula: $2\ r \geq m + r + 1$ (1) where, r = redundant bit, m = data bit. Suppose the number of data bits is 7, then the number of redundant bits can be calculated using: $r = 2\ 4 \geq 7 + 4 + 1$ (2) Thus, the number of redundant bits= 4.

1st Code:

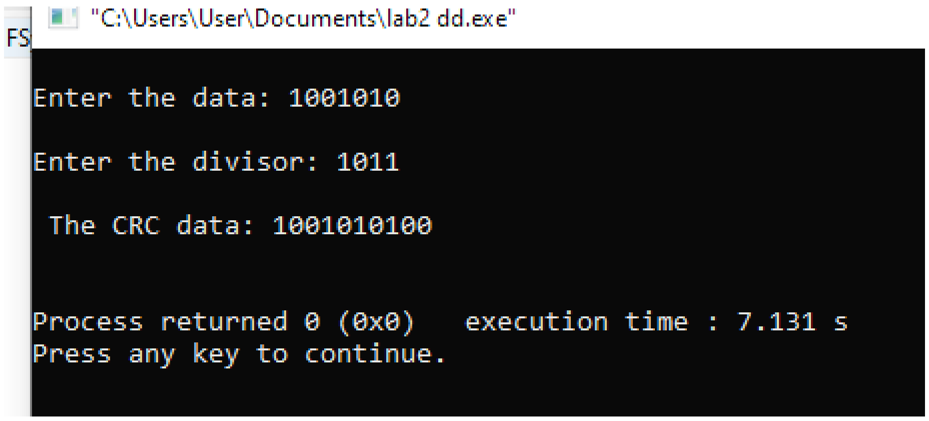
```
#include<stdio.h>

#include <math.h>

char data [100] , div [100] , temp [100] , total [ 100 ] ;

int i , j , datalen , divlen , len;


void check ( )

{

for ( j =0; j<divlen ; j++){
```

```
data [j]= total [j] ;


}


while ( j<=len )

{

if ( data [0]== '1' ){


 for ( i = 1 ; i <divlen ; i++){

   if ( data [i] == div [i] ){

        data [i] = '1';

 }

 else{

    data [i] = '0';

 }

}

}

for ( i =0; i<divlen - 1; i++){

 data [ i ]= data [ i + 1];
}
data [ i ]= total [ j++];
}
}
int main ( )
{
printf("\nEnter the data: " ) ;
scanf  ("%s",data ) ;
int x=0;
 while(data[x]!='\0')
 {
     x++;
}
datalen=x;

printf ("\nEnter the divisor: " ) ;
scanf  ("%s" , div ) ;

 x=0;
 while( div[x]!='\0')
 {
    x++;
}
divlen=x;
len = datalen + divlen - 1;

for ( i =0; i< datalen ; i++)
{
total [i]= data [i] ;
temp [i]= data [i] ;
}
for ( i=datalen ; i<len ; i++)
total [i] =  '0';

check ( ) ;
for ( i =0; i<divlen ; i++)
temp [ i + datalen ]= data [i] ;
printf ("\n The CRC data: %s " , temp ) ;
printf("\n\n");
}
```

Output:

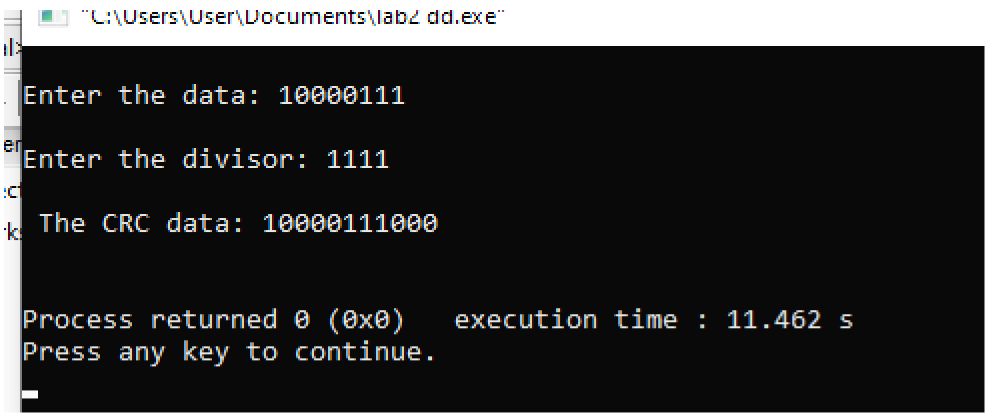


```
Enter the data: 1001010

Enter the divisor: 1011

 The CRC data: 1001010100


Process returned 0 (0x0)   execution time : 7.131 s
Press any key to continue.
```



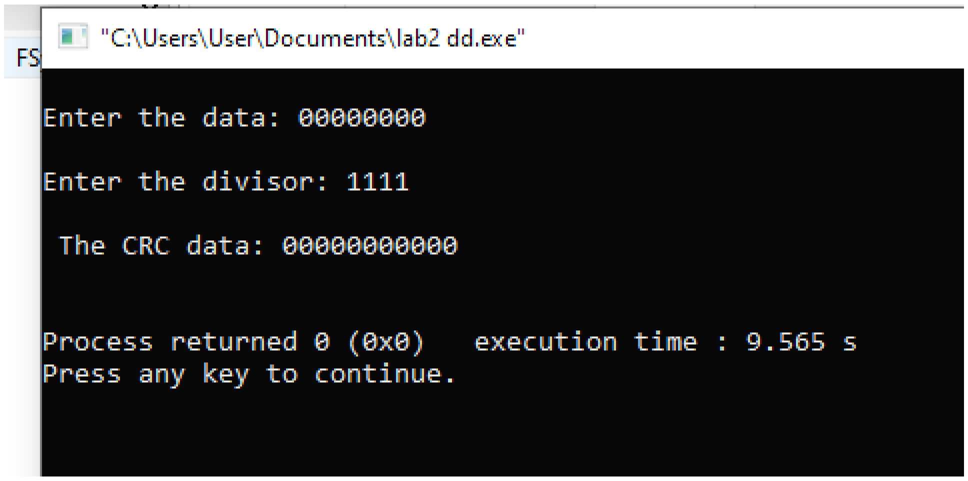
"C:\Users\User\Documents\lab2 dd.exe"

```
Enter the data: 1001010

Enter the divisor: 1011

 The CRC data: 1001010100


Process returned 0 (0x0)   execution time : 7.131 s
Press any key to continue.
```



"C:\Users\User\Documents\lab2 dd.exe"

```
Enter the data: 10000111

Enter the divisor: 1111

 The CRC data: 10000111000


Process returned 0 (0x0)   execution time : 11.462 s
Press any key to continue.
```



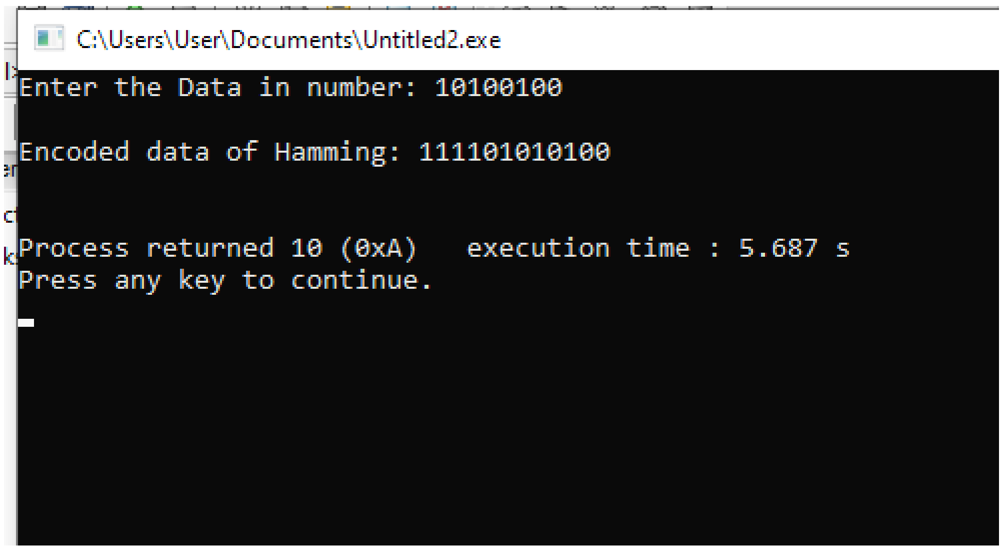
```
"C:\Users\User\Documents\lab2 dd.exe"

Enter the data: 00000000

Enter the divisor: 1111

 The CRC data: 00000000000


Process returned 0 (0x0)   execution time : 9.565 s
Press any key to continue.
```

2nd Code:

```
#include <stdio.h>

#include <math.h>

char main_Data[100];

char Encoded[100]

;

int Odd_parity_Hamming(int a,int b)

{

int count=0,i,j;

 i=a-1;

 while(i<b)

{
```

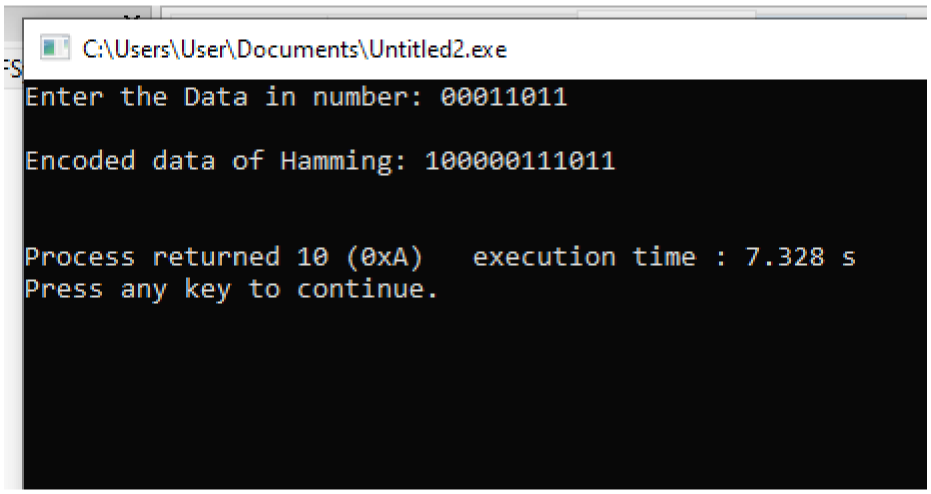
```
for(j=i;j<i+a;j++)
 {
if(Encoded[j] == '1')
 count++;
 }
 i=i+2*a;
}
if(count%2 == 0)
return '0';
else
 return '1';
 }
void main()
{
 int i=0,G=0,R=0,Z,j,k;

printf("Enter the Data in number: ");
 scanf("%s", main_Data);
printf("\n");

 while( main_Data[i]!='\0')
 {
    G++;
    i++;
}

i=0;
while(G>(int)pow(2,i)-(i+1))
 {
 R++;
 i++;
 }
 Z = R + G;
 j=k=0;
 for(i=0;i<Z;i++)
 {
 if(i==((int)pow(2,k)-1))
 {
Encoded[i]=0;
 k++;
 }
 else
{
Encoded[i]=main_Data[j];
 j++;
 }
 }
 for(i=0;i<R;i++)
 {
 int X = (int)pow(2,i);
 int Y = Odd_parity_Hamming(X,Z);
Encoded[X-1]= Y;
 }

printf("Encoded data of Hamming: %s",Encoded);
printf("\n");
printf("\n");
}
```

Output:



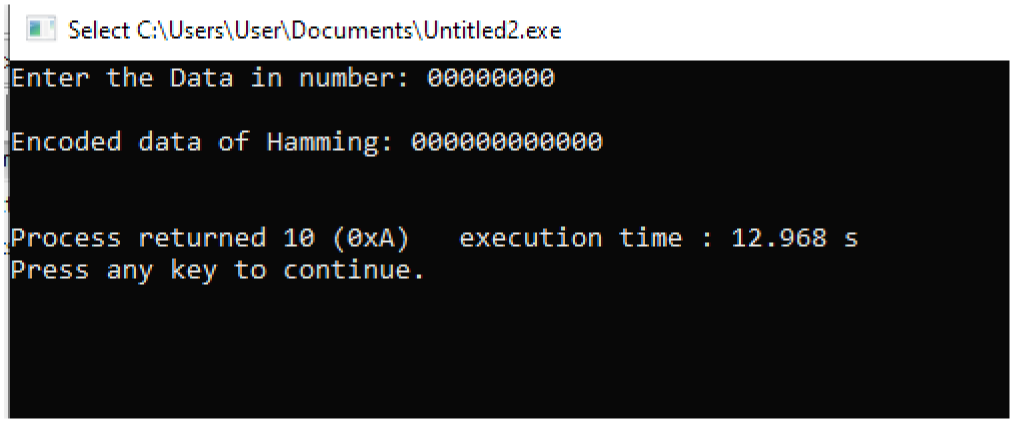
C:\Users\User\Documents\Untitled2.exe

```
Enter the Data in number: 10100100

Encoded data of Hamming: 111101010100


Process returned 10 (0xA)   execution time : 5.687 s
Press any key to continue.
```

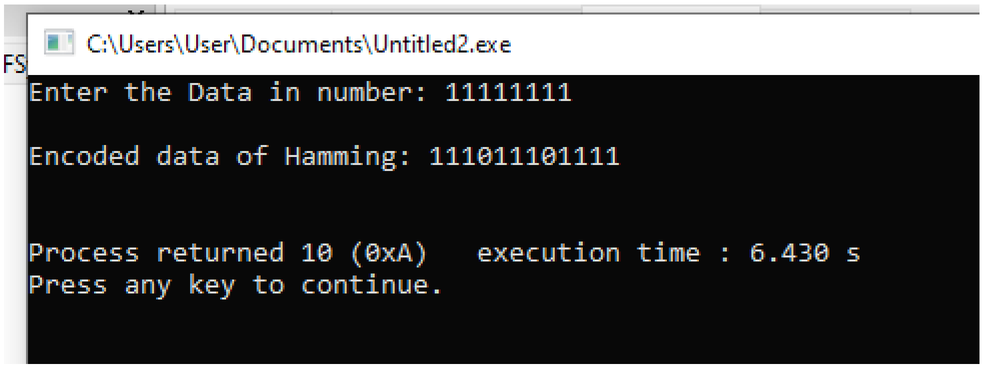


C:\Users\User\Documents\Untitled2.exe

```
Enter the Data in number: 00011011

Encoded data of Hamming: 100000111011


Process returned 10 (0xA)   execution time : 7.328 s
Press any key to continue.
```



```
Select C:\Users\User\Documents\Untitled2.exe
Enter the Data in number: 00000000

Encoded data of Hamming: 000000000000


Process returned 10 (0xA)   execution time : 12.968 s
Press any key to continue.
```



```
C:\Users\User\Documents\Untitled2.exe
Enter the Data in number: 11111111

Encoded data of Hamming: 111011101111


Process returned 10 (0xA)   execution time : 6.430 s
Press any key to continue.
```

Analysis and Discussion:

1. From this lab, we knew CRC and Hamming. And the working system of CRC and Hamming. From those are error detection techniques. Those are very important for save data.

2. Due to temperature problem in our classroom, we can’t do this lab with proper attention. So, we have some theoretical problem.

3. This lab is completely based on software. So it may have some Software and Mechanical errors.

4. When we use scanf() function to take user input then we can’t access space and after the data of space. It is an error.

5. From this problem, we use parity bit. When we operate with array, we facing some problem to understanding it.

6. Based on the focused objective(s) to understand about the Hamming code and CRC, the additional lab exercise made me more confident towards the fulfillment of the objectives(s).

7. Compile error.