

WEEK 13

Write a program for error detecting code using CRC- CCITT (16-bits).

CODE:

```
#include<stdio.h>
int arr[17];

void xor(int x[], int y[])
{
    int k=0;
    for(int i=1;i<16;i++)
    {
        if(x[i]==y[i])
            arr[k++]=0;
        else
            arr[i]=1;
    }
}

void main()
{
    int dd[17],div[33],ze[17],i,k;

    printf("Enter the dataword \n");
    for(i=0;i<17;i++)
        scanf("%d",&div[i]);

    for(i=i;i<33;i++)
        div[i]=0;

    for(i=0;i<17;i++)
        ze[i]=0;
    printf("Enter dividend \n");
```

```

for(i=0;i<17;i++)
    scanf("%d",&dd[i]);

i=0;
k=0;
    for(i=i;i<17;i++)
        arr[k++]=div[i];
while(i<33)
{
    if(arr[0]==0)
        xor(arr,ze);
    else
        xor(arr,dd);

    arr[16]=div[i++];

}
k=0;
for(i=17;i<33;i++)
    div[i]=arr[k++];
printf("Codeword: ");
    for(i=0;i<33;i++)
        printf("%d",div[i]);

for(i=0;i<17;i++)
    arr[i]=0;

printf("\nAt receiver end \n");

k=0;
    for(i=i;i<17;i++)
        arr[k++]=div[i];
while(i<33)
{
    if(arr[0]==0)

```

```

        xor(arr,ze);
    else
        xor(arr,dd);

    arr[16]=div[i++];

}
k=0;
for(i=17;i<33;i++)
    div[i]=arr[k++];

printf("Codeword: ");
for(i=0;i<33;i++)
    printf("%d",div[i]);
}

```

OUTPUT:

```

C:\Users\Admin\Desktop\1BM21CS047\ADA\CRC16\bin\Debug\CRC16.exe
Enter the dataword
1 0 1 1 0 0 1 1 1 1 0 0 1 0 1 1 1
Enter dividend
1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 1
Codeword: 101100111100101110000000000011011
At receiver end
Codeword: 10110011110010111000000000000000
Process returned 1 (0x1)   execution time : 49.507 s
Press any key to continue.

```

OBSERVATION:

13
write a Program for convolution using CRC-CCIT

generating polynomial = "1000101010101"

dataword = input("Enter data in Binary")

length = len(generating polynomial) - 1

for i in range(length):

dataword = '0'

remainder = ''

def xor(a,b):

d = ''

for i in range(len(a)):

if a[i] == b[i]:

d += '0'

else:

d += '1'

return d

S = dataword[length:]

remainder = xor(S, generating polynomial)

C = 0

while C < (len(dataword) - len(generating polynomial)):

remainder = xor(remainder[C+1:length+1])

remainder += dataword[C+length]

remainder = xor(remainder, generating polynomial)

C += 1

dataword += remainder

print(dataword)

Output

Enter data word

1011001110010111

Code word = 101100111100101110000000011011

Algorithm used

Code word: 101100111100101100000000011011

NLP
01/09/2023