

LAB 13

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

Code:

```
#include<stdio.h>
#include<string.h>
#define N strlen(gen_poly)
char data[28];
char check_value[28];
char gen_poly[10];
int data_length,i,j;
void XOR(){
    for(j = 1;j < N; j++)
        check_value[j] = (( check_value[j] == gen_poly[j])?'0':'1');
}
void receiver(){
    printf("Enter the received data: ");
    scanf("%s", data);
    printf("\n.....\n");
    printf("Data received: %s", data);
    crc();
    for(i=0;(i<N-1) && (check_value[i]!='1');i++);
    if(i<N-1)
        printf("\nError detected\n\n");
    else
        printf("\nNo error detected\n\n");
}
void crc(){
    for(i=0;i<N;i++)
        check_value[i]=data[i];
```

```

do{
    if(check_value[0]=='1')
        XOR();
    for(j=0;j<N-1;j++)
        check_value[j]=check_value[j+1];
    check_value[j]=data[i++];
}while(i<=data_length+N-1);
}

int main()
{
    printf("\nEnter data to be transmitted: ");
    scanf("%s",data);
    printf("\n Enter the Generating polynomial: ");
    scanf("%s",gen_poly);
    data_length=strlen(data);
    for(i=data_length;i<data_length+N-1;i++)
        data[i]='0';
    printf("\n.....");
    printf("\n Data padded with n-1 zeros : %s",data);
    printf("\n.....");
    crc();
    printf("\nCRC or Check value is : %s",check_value);
    for(i=data_length;i<data_length+N-1;i++)
        data[i]=check_value[i-data_length];
    printf("\n.....");
    printf("\n Final data to be sent : %s",data);
    printf("\n.....\n");
    receiver();
    return 0;
}

```

OUTPUT :

```
C:\Users\NAGARA\Desktop\crc.exe

Enter data to be transmitted: 101101

Enter the Generating polynomial: 1011010011

-----
Data padded with n-1 zeros : 101101000000000
-----
CRC or Check value is : 00110000
-----
Final data to be sent : 10110100110000
-----
Enter the received data: 10110100110000

-----
Data received: 10110100110000
No error detected

Process returned 0 (0x0)   execution time : 25.115 s
Press any key to continue.
```

```
C:\Users\NAGARA\Desktop\crc.exe

Enter data to be transmitted: 101101

Enter the Generating polynomial: 1011010011

-----
Data padded with n-1 zeros : 101101000000000
-----
CRC or Check value is : 00110000
-----
Final data to be sent : 10110100110000
-----
Enter the received data: 1011010011100

-----
Data received: 1011010011100
Error detected

Process returned 0 (0x0)   execution time : 197.443 s
Press any key to continue.
```

b. Write a program for congestion control using Leaky bucket algorithm.

Code :

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#define BUCKET_SIZE 10
#define RATE 1
#define PACKETS 20

int main() {
    int bucket = 0;
    int sent = 0;
    int dropped = 0;
    int i;

    printf("Leaky Bucket Congestion Control Simulation\n\n");

    for (i = 0; i < PACKETS; i++) {
        usleep(500000);

        if (bucket < BUCKET_SIZE) {
            printf("Packet %d sent. Bucket Tokens: %d/%d\n", i + 1, bucket + 1,
BUCKET_SIZE);
            bucket++;
            sent++;
        } else {
            printf("Packet %d dropped (bucket full). Bucket Tokens: %d/%d\n", i + 1,
bucket, BUCKET_SIZE);
            dropped++;
        }
    }

    printf("\nSimulation Summary:\n");
    printf("Packets Sent: %d\n", sent);
    printf("Packets Dropped: %d\n", dropped);

    return 0;
}
```

Output :

```
C:\Users\NAGARAJ\Desktop\bucket.exe
Leaky Bucket Congestion Control Simulation

Packet 1 sent. Bucket Tokens: 1/10
Packet 2 sent. Bucket Tokens: 2/10
Packet 3 sent. Bucket Tokens: 3/10
Packet 4 sent. Bucket Tokens: 4/10
Packet 5 sent. Bucket Tokens: 5/10
Packet 6 sent. Bucket Tokens: 6/10
Packet 7 sent. Bucket Tokens: 7/10
Packet 8 sent. Bucket Tokens: 8/10
Packet 9 sent. Bucket Tokens: 9/10
Packet 10 sent. Bucket Tokens: 10/10
Packet 11 dropped (bucket full). Bucket Tokens: 10/10
Packet 12 dropped (bucket full). Bucket Tokens: 10/10
Packet 13 dropped (bucket full). Bucket Tokens: 10/10
Packet 14 dropped (bucket full). Bucket Tokens: 10/10
Packet 15 dropped (bucket full). Bucket Tokens: 10/10
Packet 16 dropped (bucket full). Bucket Tokens: 10/10
Packet 17 dropped (bucket full). Bucket Tokens: 10/10
Packet 18 dropped (bucket full). Bucket Tokens: 10/10
Packet 19 dropped (bucket full). Bucket Tokens: 10/10
Packet 20 dropped (bucket full). Bucket Tokens: 10/10

Simulation Summary:
Packets Sent: 10
Packets Dropped: 10

Process returned 0 (0x0)   execution time : 10.992 s
Press any key to continue.
```

OBSERVATION :

17-8-23

CRC Implementation

WAP for error detecting code using CRC-CITT

C-code

```
#include <stdio.h>
#include <string.h>
#define N 8000 (poly)

char data[30];
char check_value[30];
char poly[10];
int data_length, i, j;

void XOR
{
    for (j=1; j<N; j++)
        check_value[j] = ((check_value[j] == poly[j]) ?
            '0' : '1');
}

void receiver()
{
    printf("Enter the received data: ");
    scanf("%s", data);
    printf("Data received: %s", data);
    CRC();
    for (i=0; i<N-1; i++) && (check_value[i] != '1')
        i++;
}
```

```

if (N-1)
    pf("Error detected\n");
else
    pf("No error detected\n");
}

void CRC()
{
    for (i=0; i<N; i++)
        check_value[i] = data[i];
    do {
        if (check_value[0] == '1')
            XOR();
        for (i=0; i<N-1; i++)
            check_value[i] = check_value[i+1];
            check_value[i] = data[i+1];
    }
    while (i < data_length + N + 1);
}

int main()
{
    printf("Enter data to be transmitted\n");
    scanf("%s", data);
    printf("Enter divisor polynomial:\n");
    scanf("%s", poly);
    data_length = strlen(data);
}

```


46

```

for (i = data_length; i < data_length + n - 1; i++)
    data[i] = '0';
P("Data padded with n-1 zeroes: %s", data);
acc();
P("CRC value is %s", check_value);
for (i = data_length; i < data_length + n - 1; i++)
    data[i] = check_value[i - data_length];
P("Final codeword to be sent: %s", data);
receiver();
return 0;
}

```

output :-

Enter data to be transmitted: 101010

Enter the divisor polynomial: 1011

Data padded with n-1 zeroes: 10101000

CRC value is: 001

Final codeword to be sent: 101010001

Enter the received data: 10001000

Error detected

Enter data to be transmitted: 101100

Enter divisor poly: 1001

Data padded with n-1 zeroes: 10110000

CRC value is: 001

Final codeword to be sent: 101100001

Enter received data: 101100001

no error detected

WAP for congestion control using Leaky Bucket
algm:

C-code:-

```
#include <stdio.h>

int main()
{
    int incoming, outgoing, back-size, n, store = 0;
    printf("Enter bucket size");
    scanf("%d", &back-size);
    printf("Enter outgoing size");
    scanf("%d", &outgoing);
    printf("Enter no of inputs:");
    scanf("%d", &n);

    while (n != 0)
    {
        printf("Enter the incoming bucket size:");
        scanf("%d", &incoming);
        if (incoming <= (back-size - store))
        {
            store += incoming;
            printf("Bucket buffer size %d out of %d",
                   store, back-size);
        }
        else
        {
            // ... (code for when incoming > back-size - store)
        }
    }
}
```

{

Pt ("Dropped %d no of packet in", incoming -
(back-size-store);

Pt ("Bucket buffer size %d out of %d in", store,
back-size);

store = back-size;

}

store = store - outgoing;

Pt ("After outgoing %d packets left out of
%d in buffer in", store, back-size);

n--;

{
}

Output:-

Enter bucket size: 5000

Enter outgoing rate: 2000

Enter no of inputs: 2

Enter the incoming packet size: 3000

Bucket buffer size 3000 out of 5000

After outgoing 1000 packets left out of 5000
in buffer.

Enter the incoming packet size: 1000

Bucket buffer size 2000 out of 5000

After outgoing 0 packets left out of 5000
in buffer.