Vandit Gabani
20BCI0090

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

LAB DA – 2

Name : Gabani Vandit Sureshbhai

Reg.No : 20BCI0090

Subject : CSE 1004 – Network and Communication

Faculty : SALEEM DURAI M.A

Topic : Error Detection and Correction

1. Hamming Code
2. CRC
3. Checksum
4. Parity check

Vandit Gabani
20BCI0090

1.Hamming Code : Method-1

Code:

```c
#include <stdio.h>
#include <string.h>
#include <math.h>
int main()
{
    printf("Enter the 4-bit Data word : ");
    int inp[4];
    for (int i = 0; i < 4; i++)
    {
        scanf("%d", &inp[i]);
    }
    int r1, r2, r3;
    r1 = inp[0] ^ inp[1] ^ inp[2];
    r2 = inp[1] ^ inp[2] ^ inp[3];
    r3 = inp[0] ^ inp[1] ^ inp[3];

    int code_word[7];
    for (int i = 0; i < 4; i++)
    {
        code_word[i] = inp[i];
    }
    code_word[4] = r1;
    code_word[5] = r2;
    code_word[6] = r3;

    printf("Message Transmitted by sender : ");
    for (int i = 0; i < 7; i++)
    {
        printf("%d", code_word[i]);
    }

    printf("\nReceiver's Side\n");
    printf("Enter Received Codeword : ");
    int receiver[7];
    for (int i = 0; i < 7; i++)
    {
        scanf("%d", &receiver[i]);
    }

    int s1, s2, s3;
    s1 = receiver[0] ^ receiver[1] ^ receiver[3] ^ receiver[4];
    s2 = receiver[1] ^ receiver[2] ^ receiver[3] ^ receiver[5];
    s3 = receiver[0] ^ receiver[1] ^ receiver[3] ^ receiver[6];

    if (s1 == s2 == s3 == 0)
```

Vandit Gabani
20BCI0090

```
    {
        printf("There is No Error\n\n");
    }
    else
    {
        int s = s1 * pow(2, s1) + s2 * pow(2, s2) + s3 * pow(2, s3);
        if (s == 1)
        {
            printf("Error detected in q0 bit\n");
            receiver[4] = receiver[4] ^ 1;
        }
        else if (s == 2)
        {
            printf("Error detected in q1 bit\n");
            receiver[5] = receiver[5] ^ 1;
        }
        else if (s == 3)
        {
            printf("Error detected in b2 bit\n");
            receiver[2] = receiver[2] ^ 1;
        }
        else if (s == 4)
        {
            printf("Error detected in q2 bit\n");
            receiver[6] = receiver[6] ^ 1;
        }
        else if (s == 5)
        {
            printf("Error detected in b0 bit\n");
            receiver[0] = receiver[0] ^ 1;
        }
        else if (s == 6)
        {
            printf("Error detected in b3 bit\n");
            receiver[3] = receiver[3] ^ 1;
        }
        else if (s == 7)
        {
            printf("Error detected in b1 bit\n");
            receiver[1] = receiver[1] ^ 1;
        }

        printf("Code word after correcting Error\n");
        for (int i = 0; i < 7; i++)
        {
            printf("%d ", receiver[i]);
        }
    }
```

```
    return 0;
}
```

Input and Output:

When no bits are changed:

```
PS F:\NetCom LAB> cd "f:\NetCom LAB\" ; if ($?)
Lab_2_ham_2 }

--------------------Sender's Side--------------

Enter the 4-bit Data word : 1 1 0 1
Message Transmitted by sender : 1101001

---------------Receiver's Side----------------

Enter Received Codeword : 1 1 0 0 0 0 1
There is No Error
```

When bits are changed:

```
PS F:\NetCom LAB> cd "f:\NetCom LAB\" ; if ($?) { gc
Lab_2_ham_2 }
--------------------Sender's Side----------------
Enter the 4-bit Data word : 1 1 0 1
Message Transmitted by sender : 1101001
---------------Receiver's Side----------------
Enter Received Codeword : 1 1 0 1 0 0 1
Error detected in q1 bit
Code word after correcting Error
1 1 0 1 0 1 1
PS F:\NetCom LAB>
```

Vandit Gabani
20BCI0090

2.Hamming Code : Method-2(Generalized)

Code:

```
#include <stdio.h>
#include <stdlib.h>

// to calculate 2^rbit
int mult(int rbit)
{
    int ret = 1, i;
    for (i = 0; i < rbit; i++)
    {
        ret = ret * 2;
    }
    return ret;
}

int main()
{
    int msize, rbit = 0, msg[50], data[60], even;

    printf("Enter Message size:");
    scanf("%d", &msize);

    // no of r bit : 2^r >= msize+rbit+1
    while (1)
    {
        if ((msize + rbit + 1) <= mult(rbit))
            break;
        rbit++;
    }

    // scanning message
    printf("Enter Message: \n");
    for (int i = 1; i <= msize; i++)
    {
        scanf("%d", &msg[i]);
    }

    // array for dataword
    int k = 0;
    int j = 1;
    for (int i = 1; i <= (msize + rbit); i++)
    {
        if (i == mult(k))
        {
            data[i] = 8;
            k++;
```

Vandit Gabani
20BCI0090

```
        }
        else
        {
            data[i] = msg[j];
            j++;
        }
    }

    // calculating r values

    /*
    r1 = 1+3+5+7+9+...
    r2 = 2+3+6+&+10+11+...
    r3 = 4+5+6+7+12+13+...
    r4 = 8 9 10 11 12 13 14 ...

    */
    for (int i = 1; i <= (msize + rbit); i++)
    {
        if (data[i] == 8) // change value from 8 to 0 bcz initial value at
rbit is 0
        {
            data[i] = 0;
            even = 0;
            for (int j = i; j <= (msize + rbit); j++)
            {
                for (int k = 0; k < i; k++)
                {
                    if (data[j] == 1)
                    {
                        even++;
                    }
                    j++;
                }
                j = j + i - 1;
            }
            if (even % 2 == 0)
            {
                data[i] = 0;
            }
            else
            {
                data[i] = 1;
            }
        }
    }
    /*
    suppose
```

Vandit Gabani
20BCI0090

```
    msize = 6
    msg  = 1 0 1 1 1 1
    rbit = 4
    data array = 4+6 = 10
    1 2 3 4 5 6 7 8 9 10
    p p 1 p 0 1 1 p 1 1   data

    for p at position 1  : then data[1] = 0
    */

    printf("Code Word is : \n");
    for (int i = 1; i <= (msize + rbit); i++)
    {
        printf("%d ", data[i]);
    }

    printf("\nEnter Generated Codeword :\n");
    for (int i = 1; i <= (msize + rbit); i++)
    {
        scanf("%d", &data[i]);
    }
    int c = 0;
    for (int i = 1; i <= (msize + rbit); i++)
    {
        if (i == mult(c)) // change value from 8 to 0 bcz initial value at
rbit is 0
        {
            // data[i] = 0;
            c++;
            even = 0;
            for (int j = i; j <= (msize + rbit); j++)
            {
                for (int k = 0; k < i; k++)
                {
                    if (data[j] == 1)
                    {
                        even++;
                    }
                    j++;
                }
                j = j + i - 1;
            }
            if (data[i] == 1)
            {
                even--;
            }

            if (even % 2 == 0 && data[i] == 1)
```

Vandit Gabani
20BCI0090

```
            {
                printf("Error Occured at position %d \n", i);
                data[i] = 0;
                break;
            }
            if (even % 2 == 1 && data[i] == 0)
            {
                printf("Error Occured at position %d \n", i);
                data[i] = 1;
                break;
            }
        }
    }

    printf("Code After Error Correction is : \n");
    for (int i = 1; i <= (msize + rbit); i++)
    {
        printf("%d ", data[i]);
    }
    return 0;
}
```

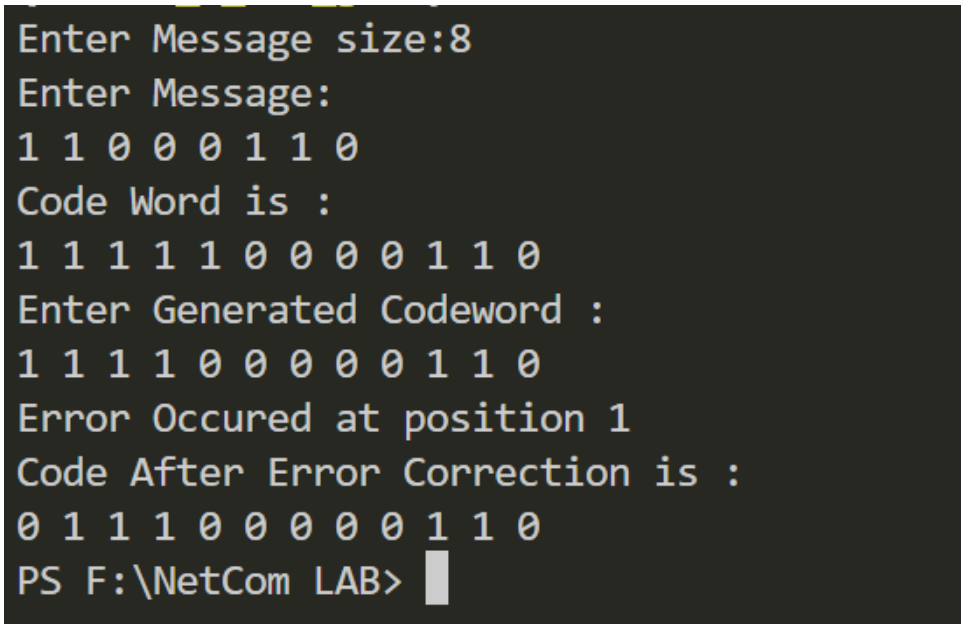
Input and Output:

When no bits are changed:

```
Enter Message size:8
Enter Message:
1 1 0 0 1 1 0 0
Code Word is :
1 0 1 1 1 0 0 0 1 1 0 0
Enter Generated Codeword :
1 0 1 1 1 0 0 0 1 1 0 0
Code After Error Correction is :
1 0 1 1 1 0 0 0 1 1 0 0
PS F:\NetCom LAB>
```

Vandit Gabani
20BCI0090

When bits are changed:



Vandit Gabani
20BCI0090

3.CRC Error Detection

Code:

```c
#include <stdio.h>
#include<math.h>
#include<string.h>
int main()
{
    int divlen, meslen;
    printf("Enter total bits of message : ");
    scanf("%d", &meslen);
    printf("Enter divisor length : ");
    scanf("%d", &divlen);
    int len = divlen + meslen - 1;
    char div[divlen], message[len];
    printf("Enter the Message : ");
    scanf("%s", message);
    printf("Enter the Divisor : ");
    scanf("%s", div);
    int divi[divlen], mess[len];
    for (int i = 0; i < divlen; i++)
        divi[i] = div[i] - '0';
    for (int i = 0; i < meslen; i++)
        mess[i] = message[i] - '0';

    // Appending 0's
    for (int i = 0; i < divlen - 1; i++)
        mess[meslen + i] = 0;

    //final message after appending 0's is now stored in mess array.
    int quo[meslen];
    int k = 0, ptr = 0, move = 0;

    //function to perform XOR division.
    while (ptr < meslen)
    {
        quo[k++] = (mess[ptr]) ? 1 : 0;
        int d = 1;
        if (mess[ptr])
        {
            for (int j = 1 + move; j < divlen + move; j++)
                mess[j] = (mess[j]) ^ (divi[d++]);
        }
        move++;
        ptr++;
    }

    printf("\nSender's End");
```

Vandit Gabani
20BCI0090

```
    printf("\nRemainder : ");
    for (int i = meslen; i < len; i++)
        printf("%d ", mess[i]);

    printf("\nCode word to be transmitted : ");
    int code[len];
    for (int i = 0; i < meslen; i++)
        code[i] = message[i] - '0';
    for (int i = meslen; i < len; i++)
        code[i] = mess[i];

    for (int i = 0; i < len; i++)
        printf("%d ", code[i]);

    code[3] = (code[3]) ? 0 : 1;

    printf("\n\nReceiver's End");
    int chk[meslen], flag = 0;
    k = 0, ptr = 0, move = 0;
    while (ptr < meslen)
    {
        quo[k++] = (code[ptr]) ? 1 : 0; // to check weather first bit is 0 or
not
        int d = 1;
        if (code[ptr])
        {
            for (int j = 1 + move; j < divlen + move; j++)
                code[j] = (code[j]) ^ (divi[d++]);
        }
        move++;
        ptr++;
    }

    printf("\nRemainder : ");
    for (int i = meslen; i < len; i++)
    {
        printf("%d ", code[i]);
        if (code[i])
            flag = 1;
    }
    if (!flag)
        printf("\nNo Error Detected");
    else
        printf("\nError Detected");
    return 0;
}
```

Vandit Gabani
20BCI0090

Input and Output:

When no bits are changed:

```
Enter the length of the data:4
Enter the data:
1 0 0 1
For the entered data the number of redundant bits is 3.
Then the divisor length is 4
Enter the divisor:
1 0 1 1

Quotient is 1010
Remainder is 110
code word is: 1001110


Enter the received codeword:1 0 0 1 1 1 0

The Remainder is 000.
So the code word is accepted.
The dataword is:1001
--------------------------------
```

When bits are changed:

```
Enter the length of the data:4
Enter the data:
1 0 0 1
For the entered data the number of redundant bits is 3.
Then the divisor length is 4
Enter the divisor:
1 0 1 1

Quotient is 1010
Remainder is 110
code word is: 1001110


Enter the received codeword:1 0 0 0 0 1 0

The Remainder is 111.
So the code word is not accepted.

--------------------------------
```

Vandit Gabani
20BCI0090

4.Parity Check

Code:

```c
#include <stdio.h>
int main()
{
    int n;
    printf("Enter length of data word : ");
    scanf("%d", &n);
    int data[n + 1];
    printf("Enter data word : ");
    for (int i = 0; i < n; i++)
    {
        scanf("%d", &data[i]);
    }
    int parity;
    for (int i = 0; i < n - 1; i++)
    {
        parity = data[i] ^ data[i + 1];
    }
    data[n] = parity;
    printf("Codeword generated is : \n");
    for (int i = 0; i < n + 1; i++)
    {
        printf("%d", data[i]);
    }
    printf("\nReceiver's End\n");
    for (int i = 0; i < n; i++)
    {
        parity = data[i] ^ data[i + 1];
    }
    if (parity == 1)
    {
        printf("There is Error\n");
    }
    else
    {
        printf("No Error\n");
        printf("Message is : ");
        for (int i = 0; i < n; i++)
        {
            printf("%d", data[i]);
        }
    }
    return 0;
}
```

Vandit Gabani
20BCI0090

Input and Output:

When no bits are changed:

```
Enter length of data word : 4
Enter data word : 1 1 0 1
Codeword generated is :
11011
Receiver's End
No Error
Message is : 1101
PS F:\NetCom LAB>
```

When bits are changed:

```
Enter length of data word : 4
Enter data word : 1 1 0 1
Codeword generated is :
11011
Receiver's End
There is Error
PS F:\NetCom LAB>
```

Vandit Gabani
20BCI0090

5.Checksum

Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

char data[100];

int rightSum(int l)
{
  int sum = 0, i = 1;
  for (; i < l; i = i + 2)
    sum = sum + (int)data[i];
  return sum;
}

int leftSum(int l)
{
  int sum = 0, i = 0;
  for (; i < l; i = i + 2)
    sum = sum + (int)data[i];
  return sum;
}

int main()
{
  char buf[100];
  int i, n, op = 0, irs = 0, ils = 0, prs = 0, cls = 0, wc = 0, pls = 0, s =
0, ocs = 0, len = 0;
  while (op == 0)
  {
    printf("\n\n1. Sender\n2. Receiver\n3. Exit\nEnter your choice.....  : ");
    scanf("%d", &n);
    switch (n)
    {
    case 1:
    {
      printf("\nEnter the data to be transmitted -> ");
      gets(buf);
      gets(data);
      len = strlen(data);
      if (len % 2 != 0)
        len++;
      irs = rightSum(len);      // initial right sum
      prs = irs % 256;          // partial right sum
      cls = irs / 256;          // carry to left sum
      ils = cls + leftSum(len); // initial left sum
```

```
    pls = ils % 256;          // partial left sum
    wc = ils / 256;           // Wrapping carry
    s = pls * 256 + prs + wc;
    ocs = 65535 - s;
    printf("The checksum generated is %X\n", ocs);
  }
  break;
  case 2:
  {
    char cs[100];
    int ch[100];
    printf("\nEnter the data received -> ");
    gets(buf);
    gets(data);
    printf("\nEnter the received checksum -> ");
    gets(cs);
    len = strlen(data);
    if (len % 2 != 0)
      len++;
    for (i = 0; i < strlen(cs); i++)
    {
      if (cs[i] >= '0' && cs[i] <= '9')
        ch[i] = cs[i] - 48;
      else if (cs[i] >= 'A' && cs[i] <= 'F')
        ch[i] = cs[i] - 55;
      else if (cs[i] >= 'a' && cs[i] <= 'f')
        ch[i] = cs[i] - 87;
    }
    irs = rightSum(len) + ch[2] * 16 + ch[3];      // initial right sum
    prs = irs % 256;                               // partial right sum
    cls = irs / 256;                               // carry to left sum
    ils = cls + leftSum(len) + ch[0] * 16 + ch[1]; // initial left sum
    pls = ils % 256;                               // partial left sum
    wc = ils / 256;                                // Wrapping carry
    s = pls * 256 + prs + wc;
    ocs = 65535 - s;
    if (ocs == 0)
      printf("\nThe message is accepted!\n");
    else
      printf("\nThe message is rejected!\n");
  }
  break;
  case 3:
    exit(0);
  }
  printf("\nPress 1 to return to main menu or 0 to exit...");
  scanf("%d", &i);
  if (i == 0)
```

```
      op = 1;
  }
  return 0;
}
```

Input and Output:

Binary:

```
1. Sender
2. Receiver
3. Exit
Enter your choice.....  : 1

Enter the data to be transmitted -> 1100110010101010
The checksum generated is 787C

Press 1 to return to main menu or 0 to exit...1


1. Sender
2. Receiver
3. Exit
Enter your choice.....  : 2

Enter the data received -> 1100110010101010

Enter the received checksum -> 787C

The message is accepted!

Press 1 to return to main menu or 0 to exit...
```

Vandit Gabani
20BCI0090

```
1. Sender
2. Receiver
3. Exit
Enter your choice.....  : 1

Enter the data to be transmitted -> 1100110011110000
The checksum generated is 7A7A

Press 1 to return to main menu or 0 to exit...1


1. Sender
2. Receiver
3. Exit
Enter your choice.....  : 2

Enter the data received -> 1100110011110000

Enter the received checksum -> 7A7B

The message is rejected!

Press 1 to return to main menu or 0 to exit...0
PS F:\NetCom LAB>
```

Vandit Gabani
20BCI0090

Hexadecimal (Message):

No error:

```
----------Sender Side----------

Enter the Message/Data word : vandit
Enter initial checksum: 0000

The unwrapped sum is 14E39
Enter the inputs for the wrapped sum:
4E39 1

The wrapped sum is 4E3A
The checksum generated is FFFFB1C5
----------Receiver Side----------

Enter the Message/Data word : vandit
Enter checksum: B1C5

The unwrapped sum is 1FFFE
Enter the inputs for the wrapped sum:
FFFE 1

The wrapped sum is FFFF
The checksum generated is FFFF0000
**NOTE:When complementing the hexadecimal value, I am getting 8 bits instead of 4.So please ignore
the first four bits.

No Error in the Transmission
```

Vandit Gabani
20BCI0090

If data changed :

```
---------------Sender Side---------------

Enter the Message/Data word : vandit
Enter initial checksum: 0000

The unwrapped sum is 14E39
Enter the inputs for the wrapped sum:
4E39 1

The wrapped sum is 4E3A
The checksum generated is FFFFB1C5
---------------Receiver Side---------------

Enter the Message/Data word : vandiv
Enter checksum: B1C3

The unwrapped sum is 1FFF8
Enter the inputs for the wrapped sum:
FFF8 1

The wrapped sum is FFF9
The checksum generated is FFFF0006
**NOTE:When complementing the hexadecimal value, I am getting 8 bits instead of 4.So please ignore
the first four bits.

There is an Error in the Transmission
```