Om Shete
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE
TE C32
2103163

Experiment 3

Aim : Implementation of CRC for error detection.

Theory :

- **Cyclic Redundancy Check (CRC)**

⇒ An error detection mechanism in which is a special number is appended to a block of data in order to detect any changes introduced during storage.

- The CRC is a method of detecting accidental changes/errors in the communication channel.
- CRC uses generator polynomial which is available on both sender and receiver side.
- An example generator polynomial is of the form like x^3+x+1. This generator polynomial represents key 1011.
- Another example is x^2+1 that represents key 101.
- n : Number of bits in data to be sent from sender side.
  k : Number of bits in the key obtained from generator polynomial.
- **Sender side**

⇒

1. The binary data is first argumented by adding k−1 zeros in end of the data.

2. Use modulo - 2 binary division to divide binary data by the key and store remainder of division.
3. Append the remainder at the end of the data to form the encoded data and send the same.

- Receiver Side

⇒ Perform modulo - 2 division again and if the remainder is 0, then there are no errors.

- Modulo 2 Division

⇒ The process of modulo-2 binary division is the same as the familiar division process use for decimal number. Instead of substraction, we use XOR here.

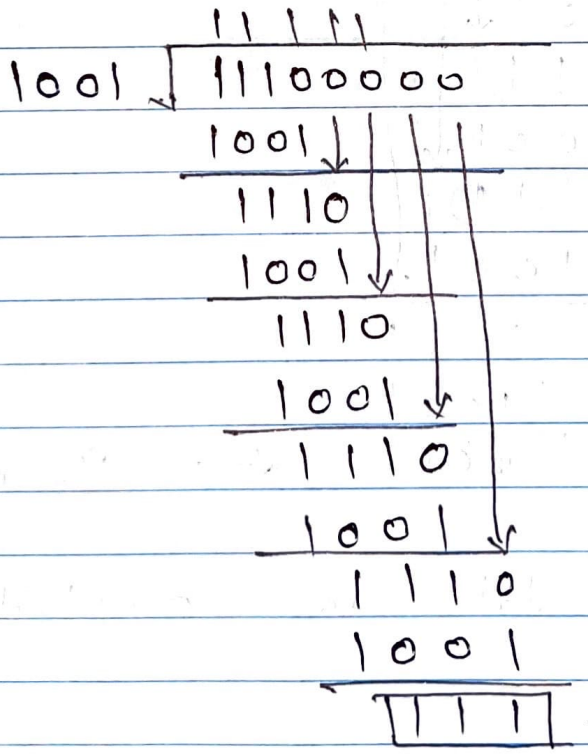
- In each step, a copy of the divisor is XORed with the k bits of the dividend.
- The result of the XOR operation is (n-i) bits, which is used for the next step after 1 extra bit is pulled down to make it n bits long.
- When there are no bits left to pull down, we have a result. The (n-i) bit remainder which is appended at the sender side.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

- **CRC Generator**

⇒

- A CRC generator uses a modulo-2 division. Firstly ~~three zeros~~ (n-1) zeros are appended at the end of the data as the length of divisor is n.
- Now, the consider the string 11100 and divisor 1001.
- Now, the string becomes 11100000, and the resultant string is divided by divisor 1001.
- The remainder generated from the binary division is known as CRC remainder.



- CRC remainder replaces the appended string of 0s at the end of the data unit, and the final string would be 11100111 which is sent across the network.

- <u>CRC Checker</u>

⇒ The functionality of the CRC checker is similar to the CRC generator.

- When the string 111100111 is received at the receiving end, the CRC checker performs the modulo-2 division.
- A string is divided by the same divisor i.e 1001.

```
            111111
1001 ) 111100111
       1001↓
       -----
        1110
        1001↓
        -----
         1111
         1001↓
         -----
          1101
          1001↓
          -----
           1001
           1001
           -----
           0000   ←  Remainder = 0
```

- Therefore, the data is accepted.

29/08/23

Code:

```
#include <bits/stdc++.h>
using namespace std;

string xor1(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;
}

string mod2div(string dividend, string divisor) {
  int pick = divisor.length();
  string tmp = dividend.substr(0, pick);

  int n = dividend.length();

  while (pick < n) {
    if (tmp[0] == '1')
      tmp = xor1(divisor, tmp) + dividend[pick];
    else
      tmp = xor1(std::string(pick, '0'), tmp) + dividend[pick];

    pick += 1;
  }
  if (tmp[0] == '1')
    tmp = xor1(divisor, tmp);
  else
    tmp = xor1(std::string(pick, '0'), tmp);

  return tmp;
}

void encodeData(string data, string key) {
  int l_key = key.length();
  string appended_data = (data + std::string(l_key - 1, '0'));
```

```
  string remainder = mod2div(appended_data, key);

  string codeword = data + remainder;
  cout << "Remainder : " << remainder << "\n";
  cout << "Encoded Data (Data + Remainder) :" << codeword << "\n";
}

void receiver(string data, string key) {
  string currxor = mod2div(data.substr(0, key.size()), key);
  int curr = key.size();
  while (curr != data.size()) {
    if (currxor.size() != key.size()) {
      currxor.push_back(data[curr++]);
    } else {
      currxor = mod2div(currxor, key);
    }
  }
  if (currxor.size() == key.size()) {
    currxor = mod2div(currxor, key);
  }
  if (currxor.find('1') != string::npos) {
    cout << "there is some error in data" << endl;
  } else {
    cout << "correct message received" << endl;
  }
}

int main() {
  string data;
  string generator;
  int frameSize, generatorSize;
  cout << "Enter the frame size: " << endl;
  cin >> frameSize;
  cout << "Enter the generator size: " << endl;
  cin >> generatorSize;
  cout << "Enter the data:";
  cin >> data;
  cout << "Enter the generator:";
  cin >> generator;
  cout << "\nSender side..." << endl;
  encodeData(data, generator);

  cout << "\nReceiver side..." << endl;
  receiver(
```

```
    data + mod2div(data + std::string(generator.size() - 1, '0'), generator),
    generator);

  return 0;
}
```

Output: