Experiment No. 4 **Date:**

Encoding Schemes

Aim:- To study different Encoding schemes

Theory:
Encoding refers to the process of converting digital data into a format suitable for transmission over a communication medium, such as copper wires or fiber optic cables. The primary goal of encoding is to ensure reliable and accurate data transmission while considering factors such as signal integrity, synchronization, and noise resilience.

Different types of Encoding Schemes

1. **Manchester Encoding**:
    - In Manchester encoding, each bit of the digital signal is represented by a transition in the middle of a bit period.
    - A high-to-low transition (also called a falling edge) represents a 1, while a low-to-high transition (rising edge) represents a 0.
    - This encoding scheme ensures both clock recovery and data recovery, making it self-clocking.
    - Manchester encoding effectively doubles the data rate compared to non-return-to-zero (NRZ) encoding, as each bit now occupies half the time.
    - However, Manchester encoding is less efficient in terms of bandwidth utilization compared to NRZ.

2. **Non-Return-to-Zero (NRZ) Encoding**:
    - In NRZ encoding, each bit is represented by a specific voltage level over the duration of the bit period.
    - A high voltage level (e.g., +5V) may represent a 1, while a low voltage level (e.g., 0V) represents a 0.
    - NRZ encoding does not utilize transitions within the bit period, which can simplify hardware implementation.
    - However, NRZ encoding is susceptible to long runs of consecutive 0s or 1s, which can cause synchronization and clock recovery issues.
    - It's worth noting that NRZ can be further categorized into NRZ-L (Non-Return-to-Zero-Level) and NRZ-I (Non-Return-to-Zero-Inverted).

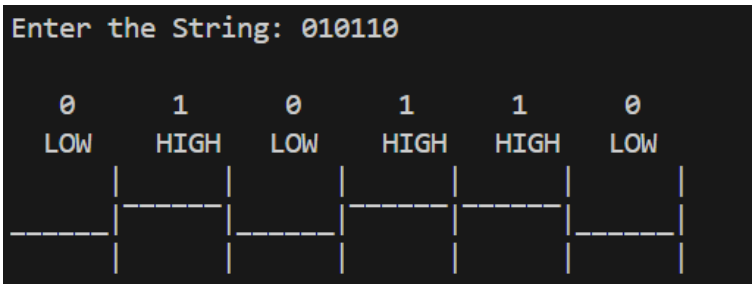
3. **Differential Manchester Encoding**:
    - Differential Manchester encoding is a variation of Manchester encoding where the presence or absence of a transition at the middle of the bit period indicates the bit value, not the direction of the transition.
    - A transition from the previous bit state to the same state represents a 0, while a transition from the previous bit state to the opposite state represents a 1.
    - This encoding scheme ensures synchronization and eliminates the need for a separate clock signal.
    - Differential Manchester encoding is more robust against noise and provides better error detection compared to regular Manchester encoding.
    - Like Manchester encoding, it effectively doubles the data rate compared to NRZ.

Code:

1) Non-Return-to-Zero (NRZ) Encoding

```
#include <iostream>
using namespace std;

int main()
{
   char a[200];
   cout << "Enter the String: ";
   cin >> a;

   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      cout<<"   "<<currentChar<<"   ";
   }
   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      if(currentChar == '0'){
         cout<<"  LOW  ";
      }
      else{
         cout<<"  HIGH ";
      }
   }
   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      cout<<"      |";
   }
   cout<<endl;

   // if 0 the low and 1 then high
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      if(currentChar == '0'){
         cout<<"______|";
      }
      else{
         cout<<"‾‾‾‾‾‾|";
      }
   }
   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      cout<<"      |";
   }

   return 0;
}
```

Output



2) Manchester Encoding

```
#include <iostream>
using namespace std;

int main()
{
   char a[200];
   cout << "Enter the String: ";
   cin >> a;

   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      cout<<"    "<<currentChar<<"    ";
   }
   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      if(currentChar == '0'){
         cout<<" LOW HIGH  ";
      }
      else{
         cout<<" HIGH LOW  ";
      }
   }
   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      cout<<"        |";
   }
   cout<<endl;

   // if 0 the low and 1 then high
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      if(currentChar == '0'){
         cout<<"_____|‾‾‾‾|";
      }
      else{
         cout<<"‾‾‾‾‾|____|";
      }
   }
   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      cout<<"        |";
   }

   return 0;
}
```

Output

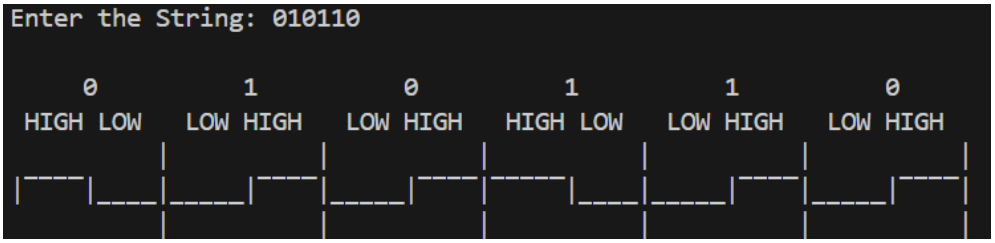
```
Enter the String: 010110

    0         1         0         1         1         0
 LOW HIGH   HIGH LOW   LOW HIGH   HIGH LOW   HIGH LOW   LOW HIGH
         |         |         |         |         |         |
_____|‾‾‾‾|‾‾‾‾‾|____|‾‾‾‾‾|____|‾‾‾‾‾|____|‾‾‾‾‾|‾‾‾‾|
         |         |         |         |         |         |
```

3) Differential Manchester Encoding

```
#include <iostream>
using namespace std;
int main()
{
   char a[200];
   cout << "Enter the String: ";
   cin >> a;
   int pos=0; //0 = low and 1 ==high
   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      cout<<"    "<<currentChar<<"    ";
   }
   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      if(currentChar == '0'){
         if(i == 0){
```

```
            cout<<" HIGH LOW  ";
            pos=0;
         }
         else{
            if(pos==0){
               cout<<" HIGH LOW  ";
               pos=0;
            }
            else{
               cout<<" LOW HIGH  ";
               pos=1;
            } } }
      else{
         if(i == 0){
            cout<<" LOW HIGH  ";
            pos=1;
         }
         else{
            if(pos==0){
               cout<<" LOW HIGH  ";
               pos=1;
            }
            else{
               cout<<" HIGH LOW  ";
               pos=0;
            }}}}
   cout<<endl;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      cout<<"         |";
   }
   cout<<endl;
   // if 0 the low and 1 then high
   pos=0;
   for (int i = 0; a[i] != '\0'; ++i) {
      char currentChar = a[i];
      if(currentChar == '0'){
         if(i == 0){
            cout<<"|‾‾‾‾|____|";
            pos=0;
         }
         else{
            if(pos==0){
               cout<<"‾‾‾‾‾‾|____|";
               pos=0;
            }
            else{
               cout<<"_____|‾‾‾‾|";
               pos=1;
            }}}
      else{
         if(i == 0){
            cout<<"|____|‾‾‾‾|";
            pos=1;
         }
         else{
            if(pos==0){
               cout<<"_____|‾‾‾‾|";
               pos=1;
            }
            else{
               cout<<"‾‾‾‾‾‾|____|";
               pos=0;
            }}} }
      cout<<endl;
      for (int i = 0; a[i] != '\0'; ++i) {
         char currentChar = a[i];
         cout<<"         |";
      }
      return 0;
   }
```

Output



Conclusion : Study of Different Encoding schemes was done Successfully