Green University of Bangladesh
Department of Computer Science and Engineering (CSE)

Faculty of Sciences and Engineering
Semester: (Spring, Year:2024), B.Sc. in CSE (Day)

Lab Report NO #02
Course Title: Data Communication Lab
Course Code:CSE-308 Section:221-D10

Lab Experiment Name: Encoding & Decoding in MATLAB.

Student Details

Name		ID
1.	Masum Hossain	221902164

Lab Date : 04/03/2024
Submission Date : 25/03/2024
Course Teacher's Name : Rusmita Halim Chaity

Lab Report Status
Marks: …… Signature:.....
Comments:..... Date:.....

1. TITLE OF THE LAB REPORT EXPERIMENT

Implementing Encoding and Decoding Scheme Using NRZ-L andEncoding and Decoding Scheme Using Pseudo ternary.

2. OBJECTIVES

This lab report aims at the implementation line coding for encoding and decoding scheme using NRZ-L and encoding and decoding scheme using pseudo ternary . This lab mainly focuses on :

- To To understand the principles and mechanisms of Non-Return-to-Zero (NRZ) Line coding scheme, specifically the NRZ-L variant, and the Pseudoternary line coding technique.
- To implement encoding and decoding algorithms for NRZ-L and Pseudo-ternary schemes in a programming language.
- To compare and analyze the performance and characteristics of NRZ-L and Pseudo-ternary coding schemes.

3. ANALYSIS

NRZ-L represents binary data using different voltage levels for 1s and 0s. It offers simplicity and constant bandwidth but is susceptible to interference and signal loss issues. The Pseudo-ternary scheme uses transitions between positive, negative, and zero voltage levels to represent data. It provides synchronization and interference immunity but requires more bandwidth and complex implementation. The choice depends on system requirements, with NRZ-L being simpler and Pseudo-ternary offering better synchronization and interference resilience.

NRZ-L Algorithm:

For Encoding:

1. Initialize output array/buffer
2. Iterate through each bit in the input binary data
3. If the current bit is 1, set the output value to a high positive voltage level
4. If the current bit is 0, set the output value to a low negative voltage level
5. Store the output value in the output array/buffer
6. Move to the next bit

For Decoding:

1. Initialize input array/buffer with received voltage levels
2. Iterate through each voltage level in the input array/buffer
3. If the current voltage level is high positive, decode as a 1
4. If the current voltage level is low negative, decode as a 0

5. Store the decoded bit in the output binary data
6. Move to the next voltage level

Pseudo-ternary Algorithm:

For Encoding:

1. Initialize output array/buffer
2. Set the initial voltage level to zero
3. Iterate through each bit in the input binary data
4. If the current bit is 1 and the previous voltage level was positive, set the output to negative
5. If the current bit is 1 and the previous voltage level was negative, set the output to positive
6. If the current bit is 1 and the previous voltage level was zero, set the output to negative
7. If the current bit is 0 and the previous voltage level was positive, set the output to zero
8. If the current bit is 0 and the previous voltage level was negative, set the output to zero
9. If the current bit is 0 and the previous voltage level was zero, set the output to positive
10. Store the output voltage level in the output array/buffer
11. Move to the next bit

For Decoding:

1. Initialize input array/buffer with received voltage levels
2. Initialize previous voltage level to zero
3. Iterate through each voltage level in the input array/buffer
4. If the current voltage level is negative and the previous voltage level was positive, decode as a 1
5. If the current voltage level is positive and the previous voltage level was negative, decode as a 1
6. If the current voltage level is negative and the previous voltage level was zero, decode as a 1
7. If the current voltage level is zero and the previous voltage level was positive, decode as a 0
8. If the current voltage level is zero and the previous voltage level was negative, decode as a 0
9. If the current voltage level is positive and the previous voltage level was zero, decode as a 0
10. Store the decoded bit in the output binary data
11. Update the previous voltage level to the current voltage level
12. Move to the next voltage level

4. IMPLEMENTATION

In this section, we critically analyze the implementation of the encoding and decoding scheme using NRZ-L and the encoding and decoding scheme using pseudo ternary.

The NRZ-L encoding and decoding algorithms were implemented in MATLAB, utilizing vectorized operations and logical indexing to efficiently map between binary data and voltage

level representations.

For Pseudo-ternary, the MATLAB implementation shows the ability to store and manipulate previous state information, enabling the tracking of voltage level transitions based on current bit values and previous levels.

Code for both Encoding and decoding in NRZ-L:

```
>> bits=input('prompt');
prompt [1 0 1 1 1 0 0 1]
>> bitrate = 1;
n = 1000;
T = length(bits)/bitrate;
N = n*length(bits);
dt = T/N;
t = 0:dt:T;
% Encoding (NRZ-L)
x = zeros(1, length(t));
for i=1:length(bits)
if bits(i)==0
x((i-1)*n+1:i*n) = 1;
else
x((i-1)*n+1:i*n) = -1;
end
end
plot(t, x, 'LineWidth',3);
title('NRZ-L Encoding');
counter = 0;
result = zeros(1, length(bits));
for i = 1:length(t)
if t(i) > counter
counter = counter + 1;
if x(i) > 0
result(counter) = 0;
else
result(counter) = 1;
end
end
end
disp('NRZ-L decoding:');
disp(result);
```

Code for both Encoding and decoding in Pseudo ternary:

```
bits=input('prompt');
prompt [1 0 1 1 1 0 0 1]
>> bitrate = 1;
n = 1000;
T = length(bits)/bitrate;
N = n*length(bits);
dt = T/N;
t = 0:dt:T;
x = zeros(1, length(t));
prev = 1;
for i=1:length(bits)
if bits(i) == 1
if prev == 1
x((i-1)*n+1:i*n) = -1;
prev = -1;
else
x((i-1)*n+1:i*n) = 1;
prev = 1;
end
else
x((i-1)*n+1:i*n) = 0;
end
end
plot(t, x, 'LineWidth',3);
title('Pseudo Ternary Encoding');
counter = 0;
result = zeros(1, length(bits));
for i = 1:length(t)
if t(i) > counter
counter = counter + 1;
if x(i) == 0
result(counter) = 1;
else
result(counter) = 0;
end
end
end
disp('Pseudo Ternary decoding:');
disp(result);
```

5. OUTPUT

The implementation of various test cases has been done for both encoding and decoding. Below are the summarized results of the tests conducted:

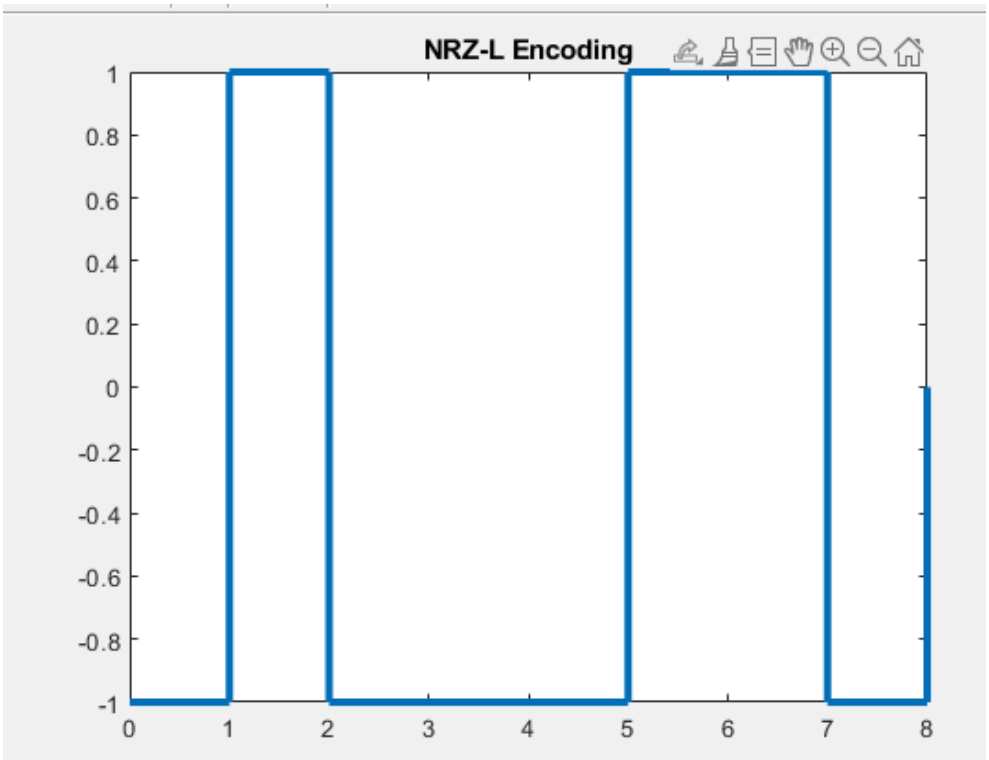

Fig01: Showing the NRZ-L output for the desired bit string.

Output for Pseudo ternary:

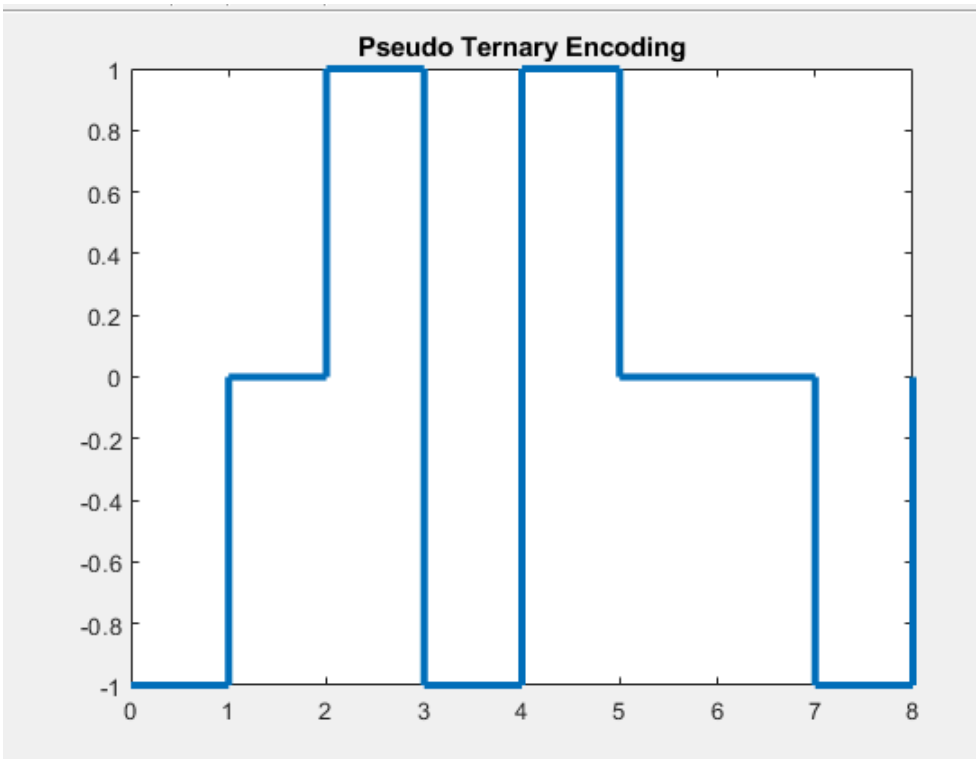

Fig02: Showing the Pseudo ternary output for the desired bit string.

6. ANALYSIS AND DISCUSSION

The NRZ-L scheme offers simplicity and bandwidth efficiency but is susceptible to interference. Pseudo-ternary provides synchronization and interference resilience through voltage level transitions but requires more bandwidth and complex implementation. NRZ-L encoding maps 1s to positive and 0s to negative voltages, while decoding reverses the process. Pseudo-ternary encoding and decoding involve tracking previous voltage levels and handling transitions based on current bit values.