AMERICAN INTERNATIONAL UNIVERSITY BANGLADESH

Faculty of Engineering

Laboratory Report Cover Sheet

Students must complete all details except the faculty use part.

Please submit all reports to your subject supervisor or the office of the concerned faculty.

Lab Title: **Study of Digital to Digital Conversion (Line Coding) using MATLAB**

Experiment Number: **04** Due Date: **19 /03/2024** Semester: **Spring 2023-2024**

Subject Code: **COE3103** Subject Name: **DATA COMMUNICATION** Section: **E**

Course Instructor: **NOWSHIN ALAM** Degree Program: **B.Sc. CSE**

Declaration and Statement of Authorship:

1. I/we hold a copy of this report, which can be produced if the original is lost/ damaged.
2. This report is my/our original work and no part of it has been copied from any other student's work or from any other source except where due acknowledgement is made.
3. No part of this report has been written for me/us by any other person except where such collaboration has been authorized by the lecturer/teacher concerned and is clearly acknowledged in the report.
4. I/we have not previously submitted or currently submitting this work for any other course/unit.
5. This work may be reproduced, communicated, compared and archived for the purpose of detecting plagiarism.
6. I/we give permission for a copy of my/our marked work to be retained by the school for review and comparison, including review by external examiners.

I/we understand that

7. Plagiarism is the presentation of the work, idea or creation of another person as though it is your own. It is a form of cheating and is a very serious academic offence that may lead to expulsion from the University. Plagiarized material can be drawn from, and presented in, written, graphic and visual form, including electronic data, and oral presentations. Plagiarism occurs when the origin of the material used is not appropriately cited.
8. Enabling plagiarism is the act of assisting or allowing another person to plagiarize or to copy your work.

Group Number (if applicable): 08 Individual Submission Group Submission

No.	Student Name	Student Number	Student Signature	Date
Submitted by:				
1	**A. F. M. RAFIUL HASSAN**	**22-47048-1**		
Group Members:				
2	**MD. SHOHANUR RAHMAN SHOHAN**	**22-46013-1**		
3	**MD. TAHSIN HASIB**	**22-46026-1**		
4	**KHUSHBU ALAM RAHI**	**22-46947-1**		
5	**MD. ASHIKUZZAMAN ABIR**	**22-47006-1**		

For faculty use only:

Total Marks: _______ Marks Obtained: _______

Faculty comments____

Introduction:-

In this study, we're diving into Digital to Digital - Conversion (Line Coding) with MATLAB. We're exploring - how to change digital data for better sending through - Communication channels. Line coding is really important here - because, it makes sure digital systems can talk to each - other reliably.

MATLAB provides robust tools for analysis, simulation and implementation facilitating our understanding of various line coding techniques. We'll looks these methods like Non-Return-to-Zero (NRZ), - as well as advanced techniques such as Alternate Mark - Inversion (AMI), Polar NRZ-Level (NRZ-L), Manchester encoding, - and Multi-Level Transmit (MLT-3).

We'll see how they affect signal quality, how much bandwidth they use, and how will they handle errors - by doing - some MATLAB Tests.

Our goal is to make communication faster and more - dependable in today's digital world.

Theory:-

Line Coding:- It refers to the conversion of digital data - into digital signals. This process assumes that data, which - could be text, numbers, images, audio or video is stored in - computer's memory as sequence of bits. Line coding takes these bit sequences and transforms them into a digital - signal. At the sender's end, digital data is encoded into a digital signal, and at the receiver's end, the digital - signal is decoded to reconstruct the original data.

Signal Elements and Data Elements:- Distinguishing between signal - elements and data elements is crucial in understanding - data communications. While our aim is to transmit data - elements, which are the smallest units representing information (i.e. bits). Signal elements serve as carriers for these - data elements. A signal element is essentially the shortest - unit of a digital signal in terms of time. Therefore, while data elements are the entities we intend to - transmit, signal elements are the units through which - we actually transmit data.

$S = C * N * (1/R)$; [S = Signal Rate, C = Case factor,
N = Data Rate, R = (Number of data elements) / (Number of signal elements)]

Bandwidth:- It describes the frequency range of a digital signal. While theoretically infinite, real-world digital signals typically have finite bandwidths due to negligible signal components. Hence, when refferring to a digital signal's bandwidth, we consider this effective range.

Line Coding Schemes:-

Line Coding
- Unipolar — NRZ
- Polar — NRZ, RZ & biphase (Manchester, differential Manchester)
- Bipolar — AMI and Pseudoternary
- Multilevel — 2B/1Q, 8B/6T, and 4D-PAM5
- Multitransition — MLT-3

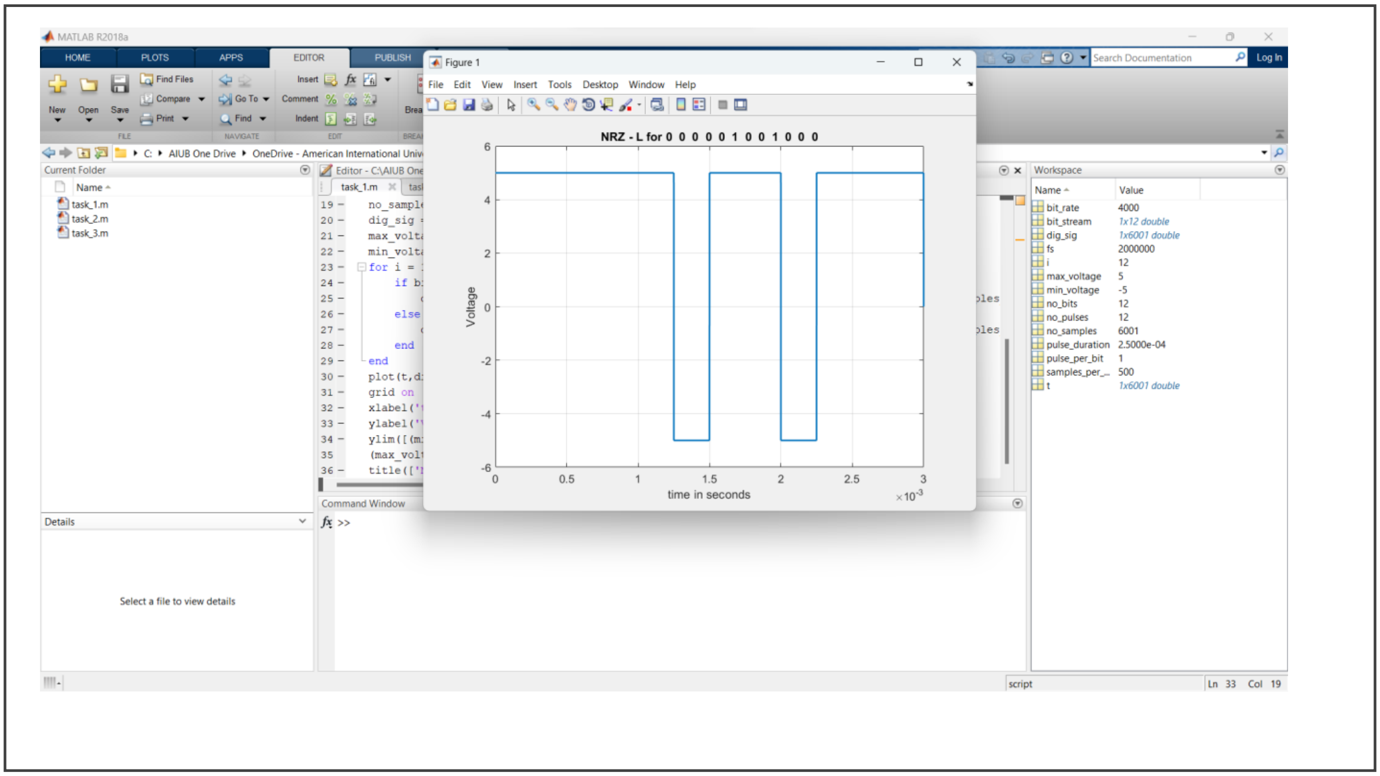
Multilevel:- Multilevel encoding aims to increase data rates or reduce bandwidth by encoding m data elements into n signal elements. With two data elements (0s and 1s), we can generate 2^m data patterns. Different signals levels enable diverse signal elements. If, $2^m = L^n$, each data pattern corresponds to one signal pattern. However, when $2^m > L^n$ some data patterns cannot be encoded.

1

Polar NRZ-L assuming bit rate is 4 kbps.

Code and Simulations:

```
% polar NRZ-L
% 22-47048-1
% AB-CDEFG-H
clc
clear all
close all
bit_stream = [0 0 0 0 0 1 0 0 1 0 0 0];
no_bits = length(bit_stream);
bit_rate = 4000; % 1 kbps
pulse_per_bit = 1; % for unipolar nrz
pulse_duration = 1/((pulse_per_bit)*(bit_rate));
no_pulses = no_bits*pulse_per_bit;
samples_per_pulse = 500;
fs = (samples_per_pulse)/(pulse_duration); %sampling frequency
% including pulse duration in sampling frequency
% ensures having enough samples in each pulse
t = 0:1/fs:(no_pulses)*(pulse_duration); % sampling interval
% total duration = (no_pulse)*(pulse_duration)
no_samples = length(t); % total number of samples
dig_sig = zeros(1,no_samples);
max_voltage = 5;
min_voltage = -5;
for i = 1:no_bits
    if bit_stream(i) == 0
        dig_sig(((i-1)*(samples_per_pulse)+1):i*(samples_per_pulse)) =
max_voltage*ones(1,samples_per_pulse);
    else
        dig_sig(((i-1)*(samples_per_pulse)+1):i*(samples_per_pulse)) =
min_voltage*ones(1,samples_per_pulse);
    end
end
plot(t,dig_sig,'linewidth',1.5)
grid on
xlabel('time in seconds')
ylabel('Voltage')
ylim([(min_voltage - (max_voltage)*0.2)
(max_voltage+max_voltage*0.2)])
title(['NRZ - L for ',num2str(bit_stream),''])
```



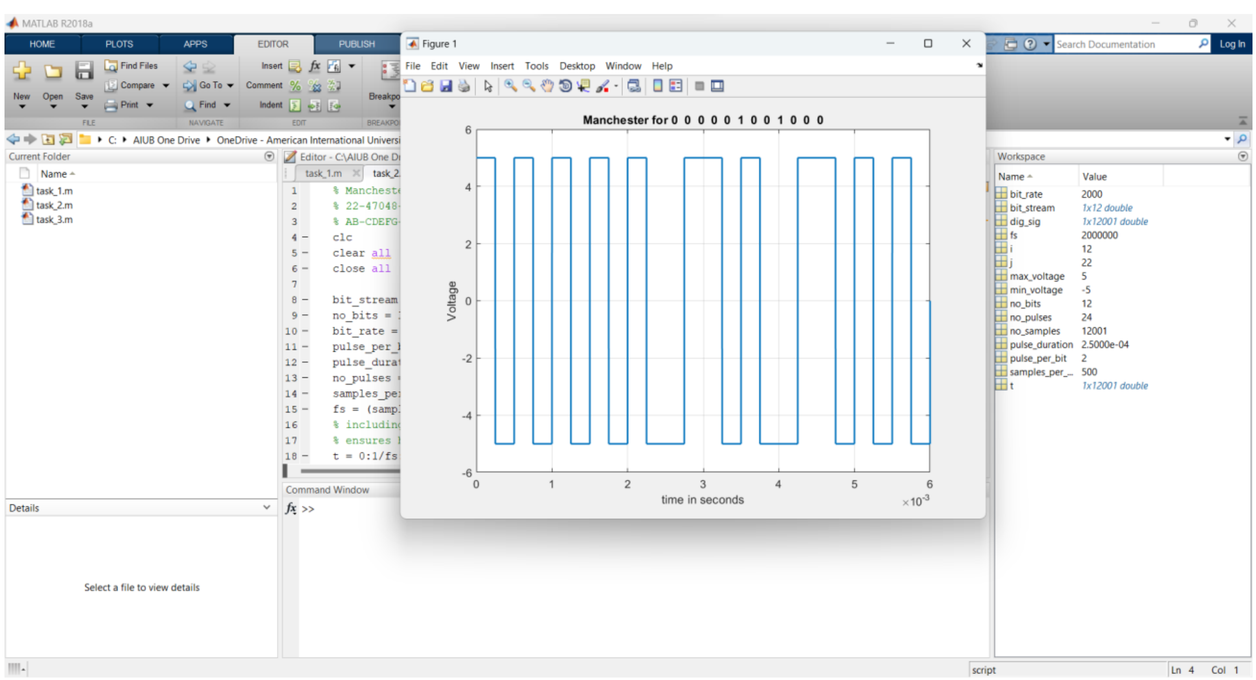
2

Manchester assuming bit rate is 2 kbps.

Code and Simulations:

```
% Manchester
% 22-47048-1
% AB-CDEFG-H
clc
clear all
close all

bit_stream = [0 0 0 0 0 1 0 0 1 0 0 0];
no_bits = length(bit_stream);
bit_rate = 2000; % 1 kbps
pulse_per_bit = 2; % for unipolar rz
pulse_duration = 1/((pulse_per_bit)*(bit_rate));
no_pulses = no_bits*pulse_per_bit;
samples_per_pulse = 500;
fs = (samples_per_pulse)/(pulse_duration); %sampling frequency
% including pulse duration in sampling frequency
% ensures having enough samples in each pulse
t = 0:1/fs:(no_pulses)*(pulse_duration); % sampling interval
% total duration = (no_pulse)*(pulse_duration)
no_samples = length(t); % total number of samples
```

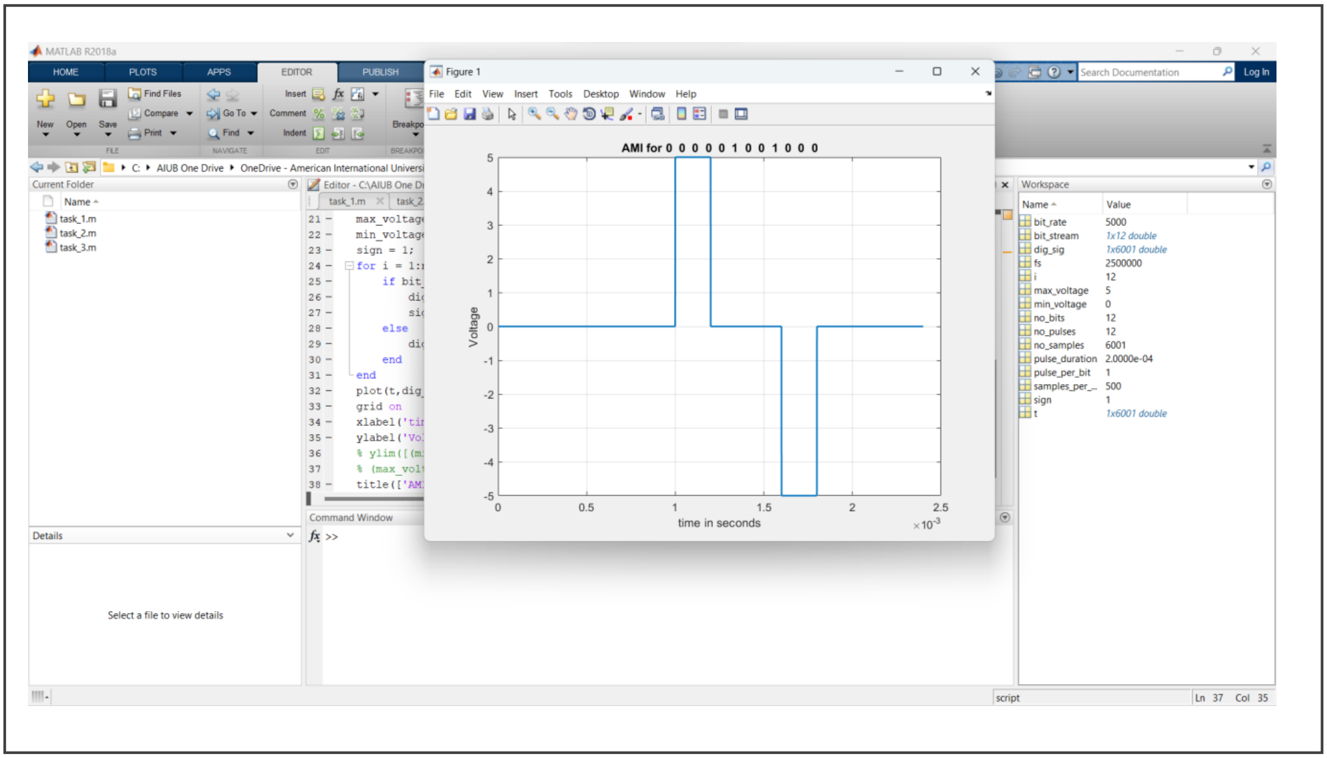
```
dig_sig = zeros(1,no_samples);
max_voltage = 5;
min_voltage = -5;
for i = 1:no_bits
    j = (i-1)*2;
    if bit_stream(i) == 1

dig_sig((j*(samples_per_pulse)+1):(j+1)*(samples_per_pulse)) =
min_voltage*ones(1,samples_per_pulse);

dig_sig(((j+1)*(samples_per_pulse)+1):(j+2)*(samples_per_pulse)) =
max_voltage*ones(1,samples_per_pulse);
    else

dig_sig((j*(samples_per_pulse)+1):(j+1)*(samples_per_pulse)) =
max_voltage*ones(1,samples_per_pulse);

dig_sig(((j+1)*(samples_per_pulse)+1):(j+2)*(samples_per_pulse)) =
min_voltage*ones(1,samples_per_pulse);
    end
end
plot(t,dig_sig,'linewidth',1.5)
grid on
xlabel('time in seconds')

ylabel('Voltage')
ylim([(min_voltage - (max_voltage)*0.2)
(max_voltage+max_voltage*0.2)])
title(['Manchester for ',num2str(bit_stream),''])
```



AMI assuming bit rate is 5 kbps.

Code and Simulations:

```
% AMI
% 22-47048-1
% AB-CDEFG-H
clc
clear all
close all
bit_stream = [0 0 0 0 0 1 0 0 1 0 0 0];
no_bits = length(bit_stream);
bit_rate = 5000; % 1 kbps
pulse_per_bit = 1; % for unipolar nrz
pulse_duration = 1/((pulse_per_bit)*(bit_rate));
no_pulses = no_bits*pulse_per_bit;
samples_per_pulse = 500;
fs = (samples_per_pulse)/(pulse_duration); %sampling frequency
% including pulse duration in sampling frequency
% ensures having enough samples in each pulse
t = 0:1/fs:(no_pulses)*(pulse_duration); % sampling interval
% total duration = (no_pulse)*(pulse_duration)
no_samples = length(t); % total number of samples
dig_sig = zeros(1,no_samples);
max_voltage = 5;
min_voltage = 0;
sign = 1;
for i = 1:no_bits
    if bit_stream(i) == 1
        dig_sig(((i-1)*(samples_per_pulse)+1):i*(samples_per_pulse)) =
sign*max_voltage*ones(1,samples_per_pulse);
        sign = (-1)*sign;
    else
        dig_sig(((i-1)*(samples_per_pulse)+1):i*(samples_per_pulse)) =
min_voltage*ones(1,samples_per_pulse);
    end
end
plot(t,dig_sig,'linewidth',1.5)
grid on
xlabel('time in seconds')
ylabel('Voltage')
% ylim([(min_voltage - (max_voltage)*0.2)
% (max_voltage+max_voltage*0.2)])
title(['AMI for ',num2str(bit_stream),''])
```



Conclusion :- Studying digital-to-digital conversion, especially line coding using MATLAB, has taught us how digital data becomes digital signals. We've also acquired experience in implementing line coding schemes such as Unipolar NRZ, Polar NRZ/RZ, Manchester & differential manchester. This knowledge is essential for communication systems, offering valuable insights to communication engineers and researchers.