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.

Laboratory Title: Study of Digital to Digital Conversion (Line Coding) Using MATLAB Experiment Number: 04 Submission Date: 22/02/2023 Semester: Spring 2022 – 2023

Subject Code: COE 3201 Subject Name: Data Communication Section: K

Course Instructor: DR. SHUVRA MONDAL Degree Program: BSc 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): **01** ☐ Individual Submission ☒ Group Submission

No.	Student Name	Student ID	Contribution
1	SHEAKH, MOHAMMAD BIN AB. JALIL SHEAKH	20-42132-1	PerformanceTask (a) & (b)
2	AURTHY, MOST. LILUN NAHAR	20-43997-2	Abstract, Performance Task (a)
3	NISHAT, TARIKUL ISLAM	21-44632-1	Discussion, Conclusion, Performance Task (c)
4	MULLICK, IFTEKHAR UDDIN	21-44649-1	Performance Task (b) & (d)
5	ULLAH, MD ISMAIL JOBI	21-44747-1	Introduction, Performance Task (c)
6	ALANSAR, SADIAH	21-45612-3	

For faculty use only:

Total Marks: **Marks Obtained:**

Faculty comments

Title:
Study of Digital to Digital Conversion (Line Coding) Using MATLAB

Abstract:
The objective of this experiment was to understand the use of MATLAB for solving communication engineering problems. It also developed an understanding of Digital to Digital Conversion (Line Coding) using MATLAB

Introduction:

Line Coding: Line coding is the process of converting digital data to digital signals. We assume that data, in the form of text, numbers, graphical images, audio, or video, are stored in computer memory as sequences of bits. Line coding converts a sequence of bits to a digital signal. At the sender, digital data are encoded into a digital signal; at the receiver, the digital data are recreated by decoding the digital signal.

Signal Elements and Data Elements:
Let us distinguish between a data element and a signal element. In datacommunications, our goal is to send data elements. A data element is the smallest entity that can represent a piece of information: this is the bit. In digital data communications, a signal element carries data elements. A signal element is the shortest unit (timewise) of a digital signal. In other words, data elements are what we need to send; signal elements are what we can send.

$$S = c * N * (1/r);$$

[S = Signal Rate, c = case factor, N = Data Rate, r = (Number of Data Elements)/(Number of Signal Elements)]

Bandwidth: We know that a digital signal that carries information is nonperiodic. We also know that the bandwidth of a nonperiodic signal is continuous with an infinite range. However, most digital signals we encounter in real life have a bandwidth with finite values. In other words, the bandwidth is theoretically infinite, but many of the components have such a small amplitude that they can be ignored. The effective bandwidth is finite. From now on, when we talk about the bandwidth of a digital signal, we need to remember that we are talking about this effective bandwidth.

Multilevel: The desire to increase the data rate or decrease the required bandwidth hasresulted in the creation of many schemes. The goal is to increase the number of bits per baud by encoding a pattern of ***m*** data elements into a pattern of ***n*** signal elements. We only have two types of data elements (***0***s and ***1***s), which means that a group of ***m*** data elements can produce a combination of $\mathbf{2^m}$ data patterns. We can have different types of signal elements by allowing different signal levels. If we have ***L*** different levels, then we can produce $\mathbf{L^n}$ combinations of signal patterns. If $\mathbf{2^m = L^n}$, then each data pattern is encoded into one signal pattern. If $\mathbf{2^m < L^n}$, data patterns occupy only a subset of signal patterns. The subset can be carefully designed to prevent baseline wandering, to provide synchronization, and to detect errors that occurred during data transmission. Data encoding is not possible if $\mathbf{2^m > L^n}$ because some of the data patterns cannot be encoded.

Performance Task:

The selected ID is the following:

2	0	-	4	2	1	3	2	-	1
A	**B**		**C**	**D**	**E**	**F**	**G**		**H**

Therefore,

4 bit binary to bit stream of 12 bits.
E -> 1 -> 0001
F -> 3 -> 0011
G -> 2 -> 0010

12 bit data stream = 0001 0011 0010

Performance Task 1 : Polar NRZ-L assuming bit rate is 4 kbps.

Code:

```
bit_stream = [0 0 0 1 0 0 1 1 0 0 1 0];
no_bits = length(bit_stream);
bit_rate = 4000; % 4 kbps
pulse_per_bit = 1; % for unipolar nrz
pulse_duration = (1/((pulse_per_bit)*(bit_rate)))*(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) == 1
    dig_sig(((i-1)*(samples_per_pulse)+1):
i*(samples_per_pulse))=min_voltage*ones(1,samples_per_pulse);
 else
   dig_sig(((i-
1)*(samples_per_pulse)+1):i*(samples_per_pulse))=max_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(['Unipolar NRZ for ',num2str(bit_stream),''])
```

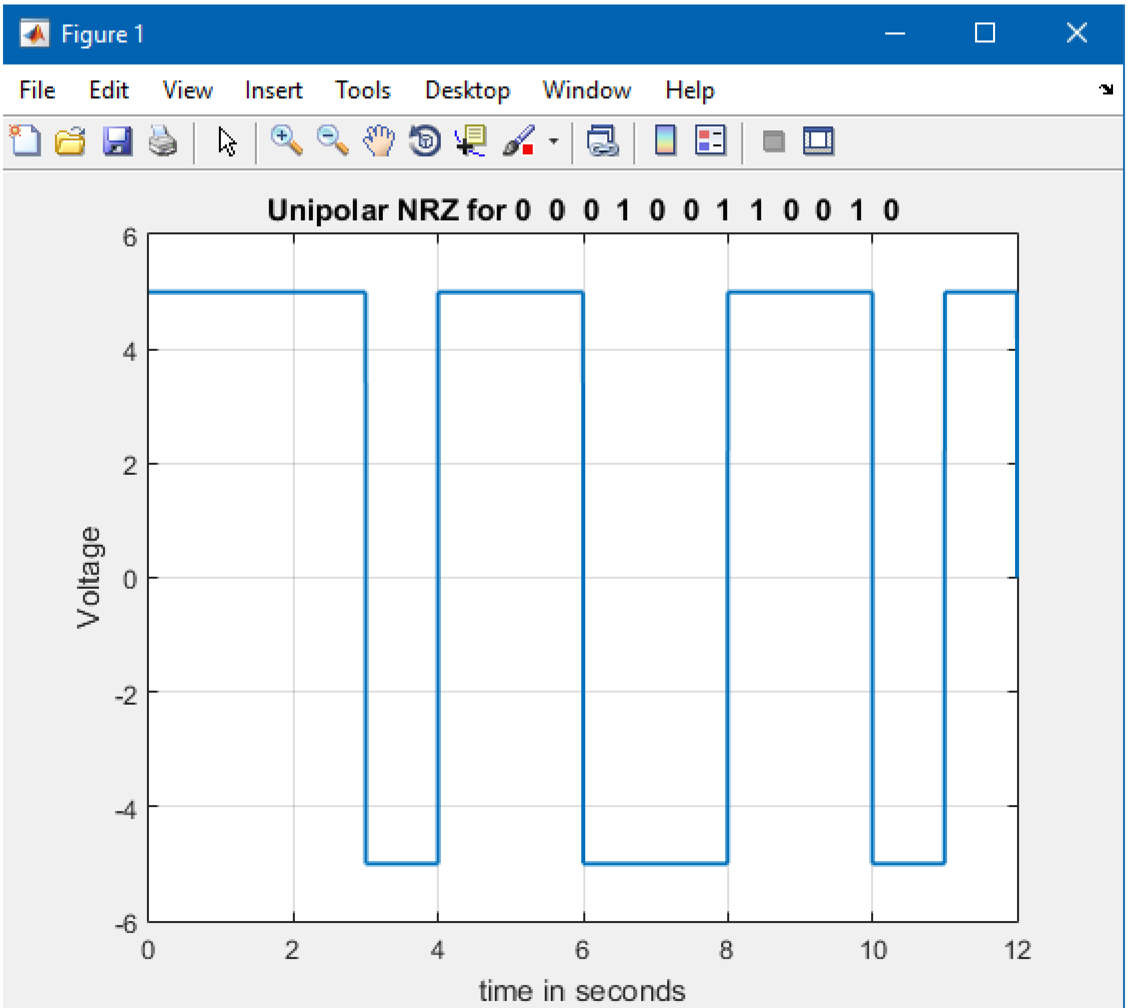

Figure 1: Unipolar NRZ

Performance Task 2 : Manchester assuming bit rate is 2 kbps.

Code:

```
bit_stream = [0 0 0 1 0 0 1 1 0 0 1 0];
no_bits = length(bit_stream);
bit_rate = 2000; % 2 kbps
pulse_per_bit = 2; % for differential
manchester
pulse_duration =
(1/((pulse_per_bit)*(bit_rate))) * 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 = +2;
min_voltage = -2;
%inv_bit = 1; % inverting bit
%last_state = max_voltage;
%inv_last_state = min_voltage; % inverse
of last state
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
figure
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),''])
```

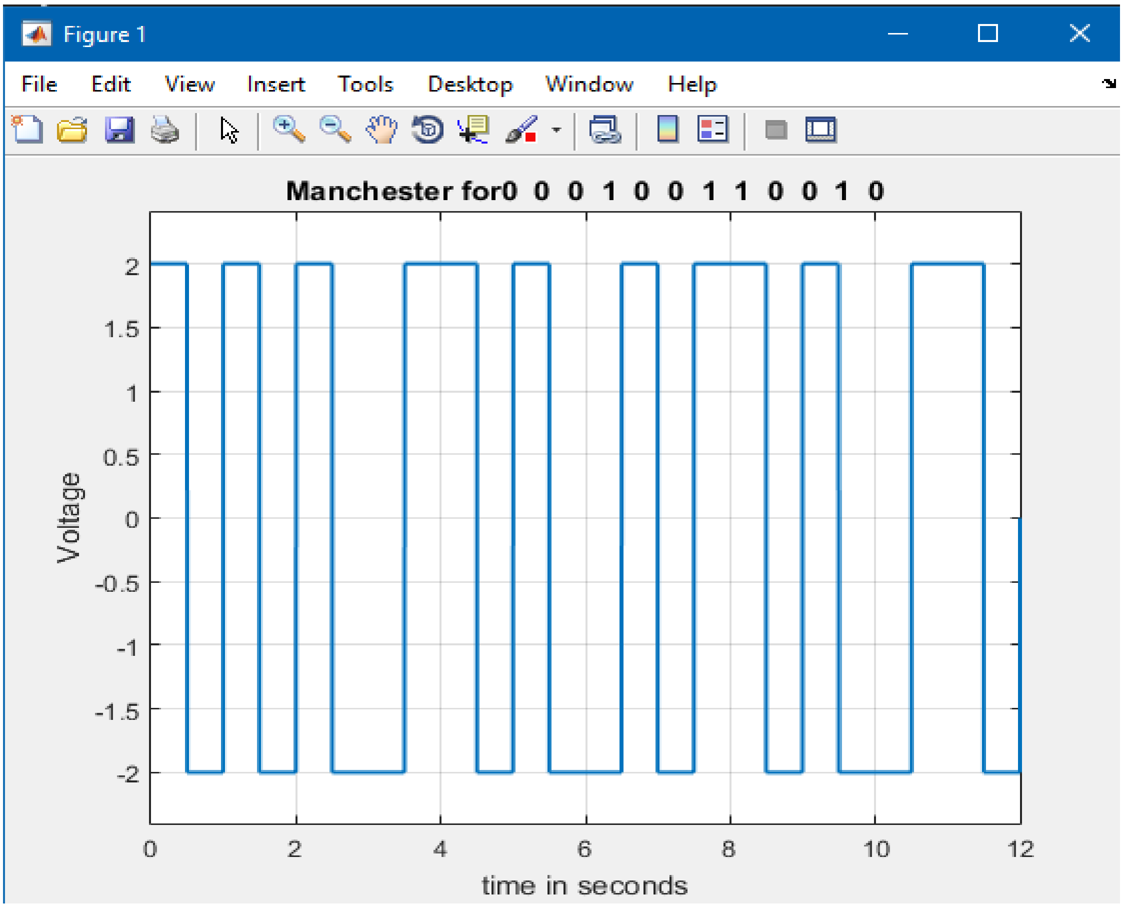

Figure 2: Manchester

Performance Task 3: AMI assuming bit rate is 5 kbps.

```
bit_stream = [0 0 0 1 0 0 1 1 0 0 1 0];
no_bits = length(bit_stream);
bit_rate = 5000;
pulse_per_bit = 1;
pulse_duration = 1;
no_pulses = no_bits*pulse_per_bit;
samples_per_pulse = 500;
fs = (samples_per_pulse)/(pulse_duration);
t = 0:1/fs:(no_pulses)*(pulse_duration);
no_samples = length(t);
dig_sig = zeros(1,no_samples);
max_voltage = +2;
avg_voltage=0;
min_voltage = -2;
inv_bit=1;
for i = 1:no_bits
        if bit_stream(i) == 1
                if inv_bit == 1
                        dig_sig(((i-
1)*(samples_per_pulse)+1):i*(samples_per_pulse))=max_voltage*ones(1,samples_per_puls
e);
                        inv_bit=0;
        else
                dig_sig(((i-
1)*(samples_per_pulse)+1):i*(samples_per_pulse))=min_voltage*ones(1,samples_per_pulse
);
                inv_bit=1;
        end
        else
                dig_sig(((i-
1)*(samples_per_pulse)+1):i*(samples_per_pulse))=avg_voltage*ones(1,samples_per_pulse
);
        end
        end
figure
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),''])
```

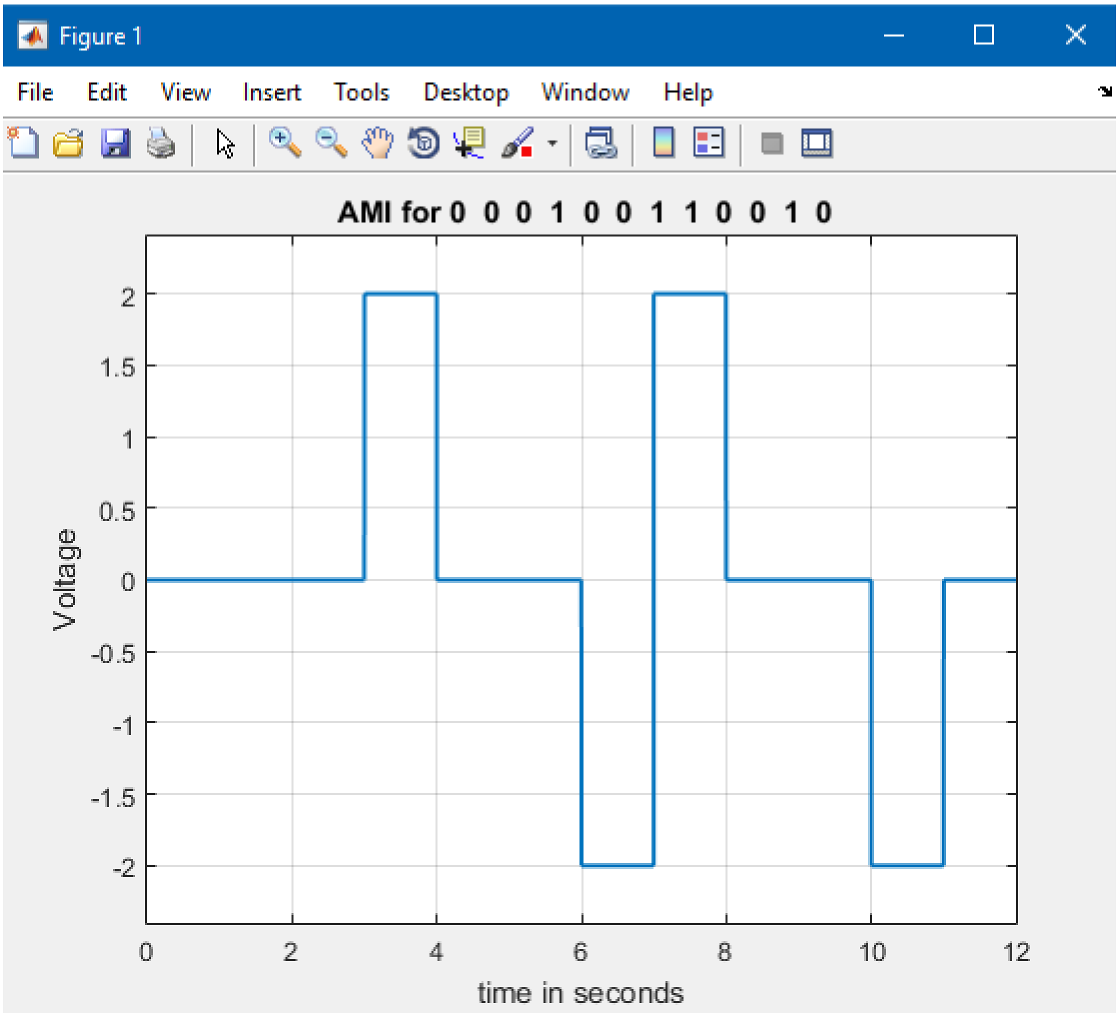

Figure 3: AMI

Performance Task 4: MLT-3 assuming bit rate is 10 kbps.

Code :

```
bit_stream = [0 0 0 1 0 0 1 1 0 0 1 0];
no_bits = length(bit_stream);
bit_rate = 10000;
pulse_per_bit =1;
pulse_duration = 1;
no_pulses = no_bits*pulse_per_bit;
samples_per_pulse = 500;
fs = (samples_per_pulse)/(pulse_duration);
t = 0:1/fs:(no_pulses)*(pulse_duration);
no_samples = length(t);
dig_sig = zeros(1,no_samples);
max_voltage =+2;
min_voltage =-2;
neutral_volt=0;
last_state=neutral_volt;
```

```
prev_last_state=min_voltage;
for i = 1:no_bits
 if bit_stream(i) == 1
 if last_state==max_voltage
 dig_sig(((i-1)*(samples_per_pulse)+1):(i*samples_per_pulse)) =
neutral_volt*ones(1,samples_per_pulse);
 last_state=neutral_volt;
 prev_last_state=max_voltage;
 elseif last_state==min_voltage
 dig_sig(((i-1)*(samples_per_pulse)+1):(i*samples_per_pulse))
=neutral_volt*ones(1,samples_per_pulse);
 last_state=neutral_volt;
 prev_last_state=min_voltage;
 else
 if prev_last_state==max_voltage
 dig_sig(((i-1)*(samples_per_pulse)+1):(i*samples_per_pulse))
=min_voltage*ones(1,samples_per_pulse);
 last_state=min_voltage;
 prev_last_state=neutral_volt;
 else
 dig_sig(((i-1)*(samples_per_pulse)+1):(i*samples_per_pulse))
=max_voltage*ones(1,samples_per_pulse);
 last_state=max_voltage;
 prev_last_state=neutral_volt;
 end
 end
 else
 dig_sig(((i-1)*(samples_per_pulse)+1):(i*samples_per_pulse))
=last_state*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(['MLT-3 bits: ',num2str(bit_stream),''])
```

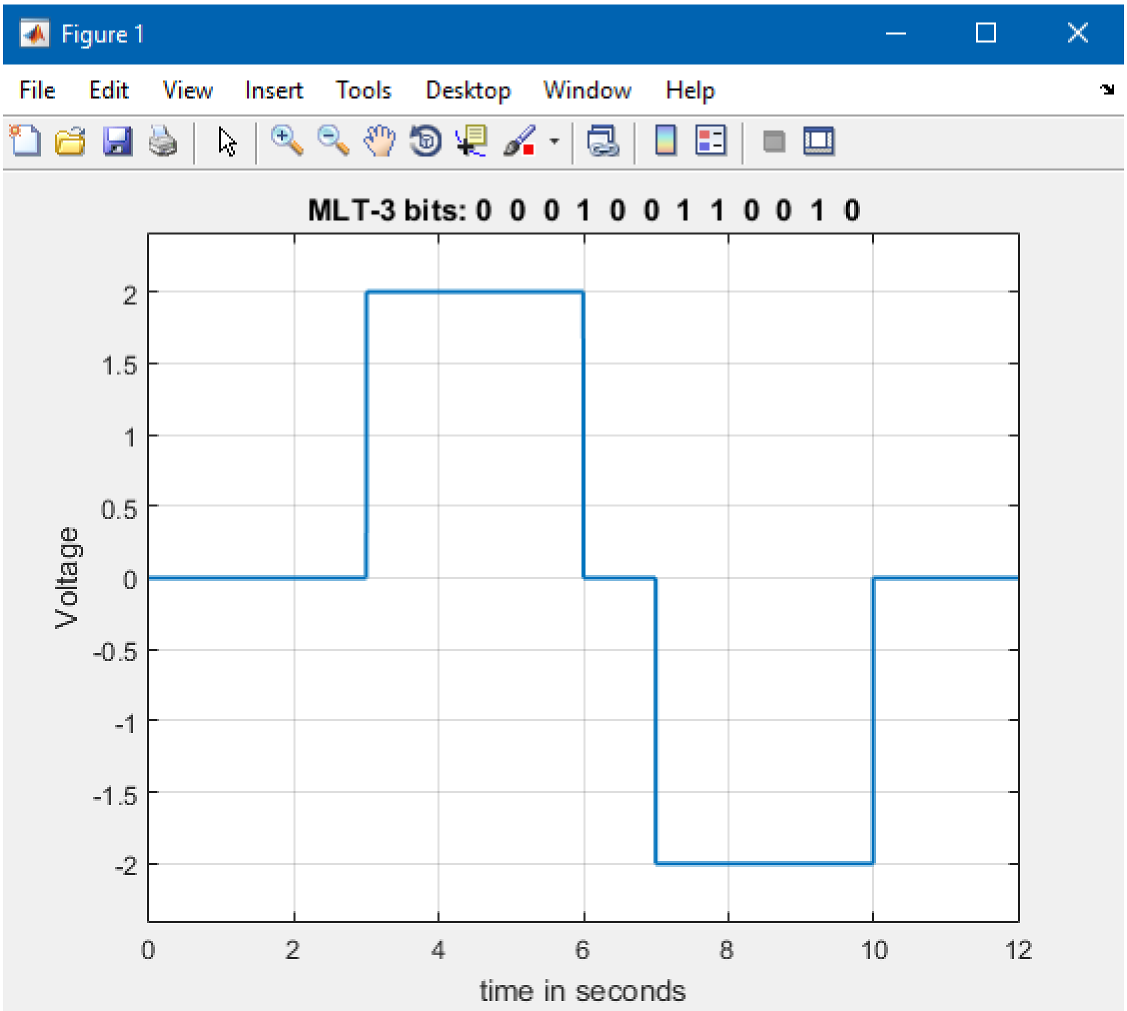

Figure 4: MLT-3

Discussion and Conclusion:

From the above simulations, various functionalities of MATLAB were observed in hand. Various functions that were available on MATLAB were learned and observed. Using this knowledge, MATLAB software plotted digital to digital conversion (line coding). There are several line coding schemes. We convert a bit stream to digital signal using those line coding scheme methods. Various formatting on the graph was learned from this experiment as well. Hence, it can be said that all the objectives of this experiment were obtained properly.

References:

- Prakash C. Gupta, “Data communications”, Prentice Hall India Pvt.
- William Stallings, "Data and Computer Communications”, Pearson
- Forouzan, B. A. "Data Communication and Networking. Tata McGraw." (2005).
- AIUB Data Communication Engineering Lab Manual, Report 04