American International University- Bangladesh (AIUB)
Faculty of Engineering
Data Communications Lab
Open Ended Lab (OEL)

Course Name:	**Data Communication**		
Course Code:	EEE 3205	**Section:**	**F**
Semester:	Spring 2023-24	**Group No:**	**02**

Assignment Name:	**Open Ended Lab (OEL)**		
Assessed CO2: **Assessed CO5**	Convert this bit stream to digital signal using different Line Coding Method		
Assessed POI:	**P.d.1.C5**		
Student Name:	**MD. JOBAER HOSSAIN**	**Student ID:**	**22-47116-1**
Student Name:	**MD SAMIN YEASAR**	**Student ID:**	**22-47139-1**
Student Name:	**RIFAH SANZIDA**	**Student ID:**	**22-47154-1**
Student Name:	**SHAYAN ABRAR**	**Student ID:**	**22-47156-1**

Mark distribution (to be filled by Faculty):

Objectives	Proficient [10-8]	Good [7-4]	Needs Improvement [3-1]	Secured Marks
Depth of knowledge displayed through appropriate research (P1)	Student was able to apply in-depth engineering knowledge achieved by appropriate research about digital/analog communication to design the communication model correctly and **fulfilled all design criteria**.	Design process is not completely supported by in-depth engineering knowledge achieved by appropriate research about digital/analog communication, **some but not all of the design criteria are fulfilled**.	Design process contains mistakes and does not display enough in-depth engineering knowledge achieved by appropriate research about digital/analog communication. **Most of the design criteria are not fulfilled**.	
Depth of analysis (P3)	Student defended the diversified approach taken to solve the problem with **well-justified in-depth analysis that demonstrated abstract thinking**.	Student's attempts to analyze the diversified approach taken to solve the problem **is not enough in-depth, some of design choices do not demonstrate adequate abstract thinking** and are not properly justified.	Student **did not attempt any in-depth analysis** of the designed system and **displayed no abstract thinking**.	
Level of integration of multiple sections of design for solution of high-level problem (P7)	Student correctly identified all problems and successfully integrated the interdependent parts into a high-level design using a block diagram. Block diagram was at best match with the given problem.	Student was able to identify some of the problems correctly and integrated the interdependent parts into a high-level design using a block diagram. Some parts of the block diagram were not a good match for the given problem.	Student was able to identify only one/two of the problems correctly and could not properly integrate the interdependent parts into a high-level design using a block diagram. Only one/two blocks were correct and/or block diagram was incomplete.	
Comments:			**Total Marks (Out of 30):**	

Question: Digital to digital conversion is the way of representing digital data by using digital signals. The conversion involves three techniques: line coding, block coding, and scrambling. Assume your ID is AB-CDEFG-H, and then convert 'B', 'E' and 'G' to 4-bit binary to have a bit stream of 12 bits. Convert this bit stream to digital signal using the following methods:

1. Polar NRZ-I (invert) assuming bit rate is 4 kbps.

2. Differential Manchester assuming bit rate is 2 kbps.

3. Pesudoternary assuming bit rate is 5 kbps.

4. MLT-3 assuming bit rate is 10 kbps.

Software:

- MATLAB

Task:

- Demonstrate an experiment using MATLAB tool to the convert digital bit stream to digitalsignal

Lab Report

Your lab report and presentation should include the following sections:

- **Purpose**

This is a summary statement of the work to be accomplished in this experiment. An overall direction forlaboratory investigation, the obtained result and summary of conclusions must be provided.

- **Procedure**

Explain step-by-step procedure in a numbered sequence so that other learners can comprehend theexperiment and be able to reproduce the experiment by reading your procedure.

- **Results**

The MATLAB code used along with the necessary diagrams to represent the proper functioning of theexperiment should be provided with proper labeling.

- **Discussion and Conclusions**

This section should be based on the information described in the report and is the closure of your report.Any advantages or limitations of the experiment should be included here. Any problems encountered while performing a particular step in the experiment can also be mentioned here.

- **Reference**

  Proper referencing of the report is required.

Abstract:

This OEL report introduces line coding schemes that convert digital data to analog signals for communication channels. The experiment employs MATLAB to explore Digital-to-Digital Conversion through schemes like Polar NRZ-I, Differential Manchester, Pseudoternary, and MLT-3. Resulting simulation graphs validate successful coding implementations. The experiment underscores the crucial role of suitable line coding in ensuring reliable and efficient digital communication systems.

Keywords:

Line coding, MATLAB, communication engineering, signal vs. data elements, coding schemes.

Introduction:

Line coding schemes are techniques used in digital communication systems to convert digital data into analog signals that can be transmitted over physical communication channels, such as cables or wireless links. These schemes are crucial because most communication channels are designed to handle analog signals. Line coding ensures that digital information is accurately and efficiently transmitted over these channels.

Purpose:

This experiment is designed to:

1. To understand the use of MATLAB for solving communication engineering problems.
2. To develop understanding of Digital-to-Digital Conversion (Line Coding) using MATLAB for the following schemes:
    a. Polar NRZ-I (invert) assuming bit rate is 4 kbps.
    b. Differential Manchester assuming bit rate is 2 kbps.
    c. Pesudoternary assuming bit rate is 5 kbps.
    d. MLT-3 assuming bit rate is 10 kbps.

Procedure:

1. Written the code for Polar NRZ-I (invert) and took the screenshot of the output graph.
2. Written the code for Differential Manchester and took the screenshot of the output graph.
3. Written the code for Pesudoternary and took and took the screenshot of the output graph.
4. Written the code for MLT-3 and took and took the screenshot of the output graph.

Results:

The ID is used 22-47154-1;
BEG = 214;
Binary = 0010 0001 0100;

Polar NRZ-I (invert)

```
%22-47154-1
%AB-CDEFG-H
%BEG = 214
%Binary  = 0010 0001 0100

clc
clear all
close all
bit_stream = [0 0 1 0 0 0 0 1 0 1 0 0];
no_bits = length(bit_stream);
bit_rate = 4000; % 4 kbps
pulse_per_bit = 1; % for Polar NRZ-I
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 = 4;
min_voltage = -4;
last_non_zero_bit = 1;

for i = 1:no_bits
   if bit_stream(i) == 1
        if last_non_zero_bit == 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
        last_non_zero_bit = last_non_zero_bit*-1;
        else
```

```
        if
            last_non_zero_bit == 1
                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
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(['Polar NRZ-I for ',num2str(bit_stream),''])
```

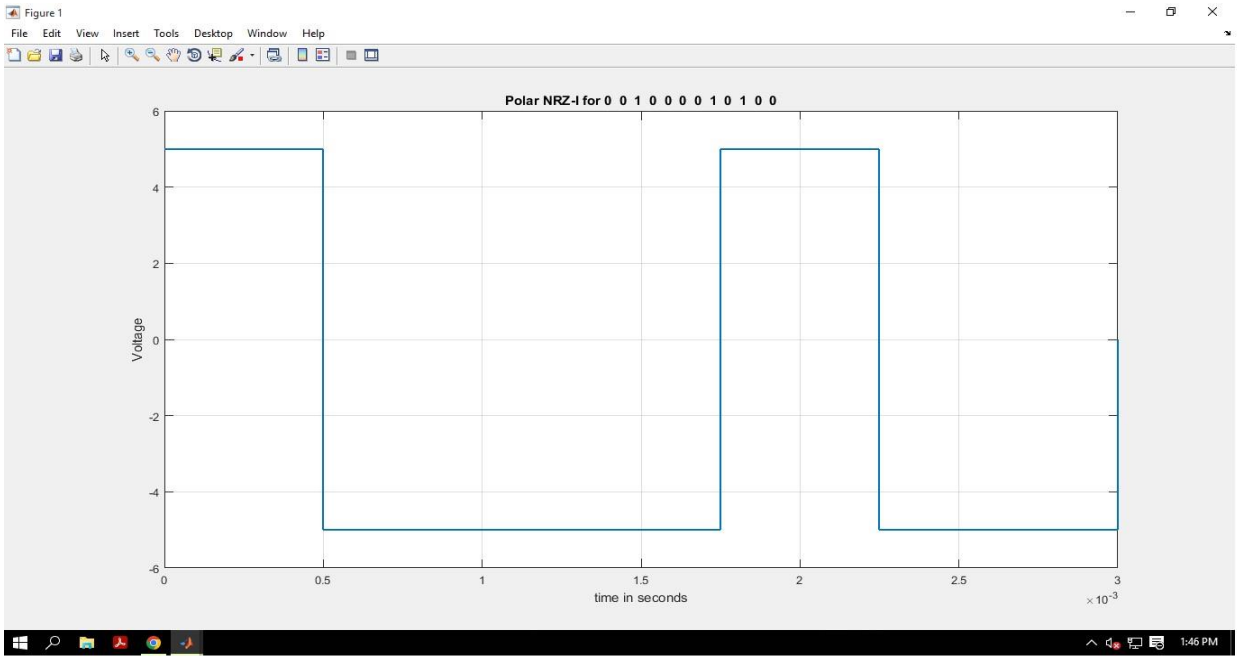

Figure 1: Simulation Graph for Polar NRZ-I

Differential Manchester

```
%22-47154-1
%AB-CDEFG-H
%BEG = 214
%Binary  = 0010 0001 0100

clc
clear all
close all
bit_stream = [0 0 1 0 0 0 0 1 0 1 0 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));
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 = 4;
min_voltage = -4;
last_non_zero_bit = 1;
 for i = 1:no_bits
   j = (i-1)*2;
   if
   bit_stream(i) == 1
        if
           last_non_zero_bit == 1
           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);
      else
           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);
      end
           last_non_zero_bit = last_non_zero_bit *-1;
      else
```

```
            if
              last_non_zero_bit == 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
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(['Differential Manchester for ',num2str(bit_stream),''])
```

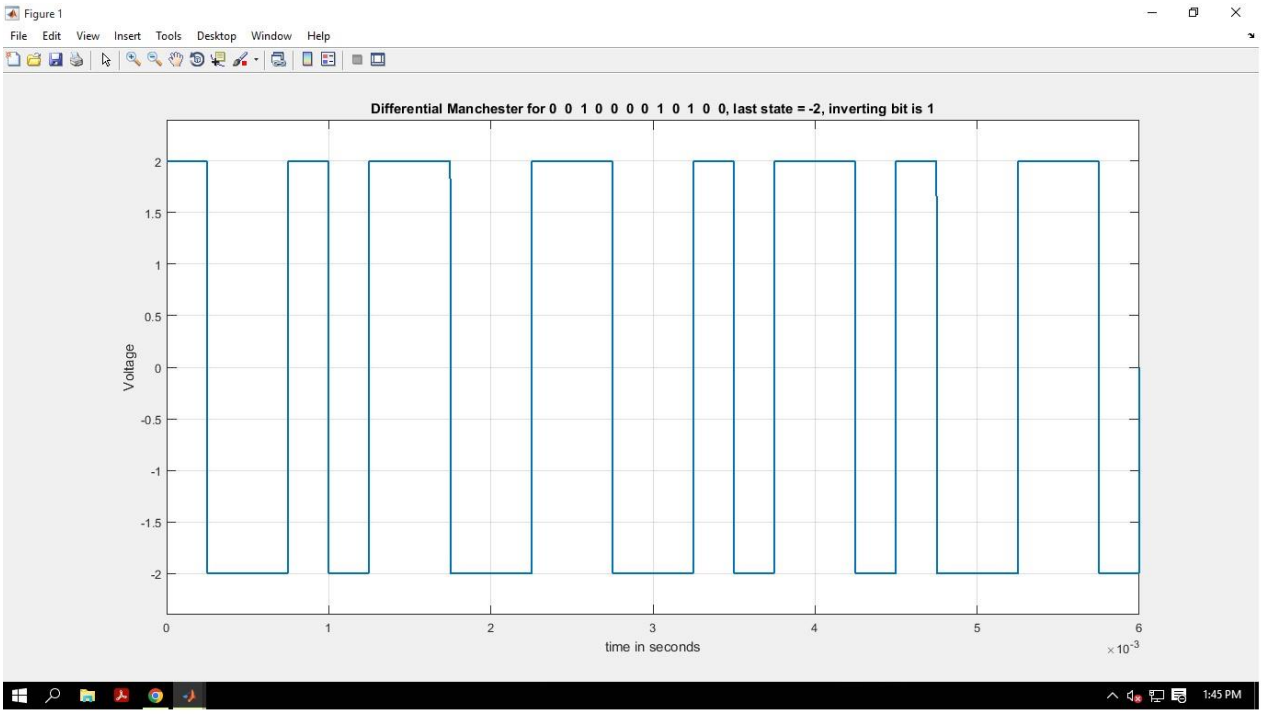

Figure 2: Simulation Graph for Differential Manchester

Pseudoternary

```
%22-47154-1
%AB-CDEFG-H
%BEG = 214
%Binary  = 0010 0001 0100

 clc
clear all
close all
bit_stream = [0 0 1 0 0 0 0 1 0 1 0 0];
no_bits = length(bit_stream);
bit_rate = 5000; % 5 kbps
 pulse_per_bit = 1; % for pseudoternary
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 = 4;
min_voltage = -4;
last_non_zero_bit = 1;
 for i = 1:no_bits
   j = (i-1)*2;
   if
   bit_stream(i) == 1
         if
            last_non_zero_bit == 1
            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);
      else
            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);
      end
            last_non_zero_bit = last_non_zero_bit *-1;
      else
```

```
            if
              last_non_zero_bit == 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
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([' Pseudoternary for ',num2str(bit_stream),''])
```

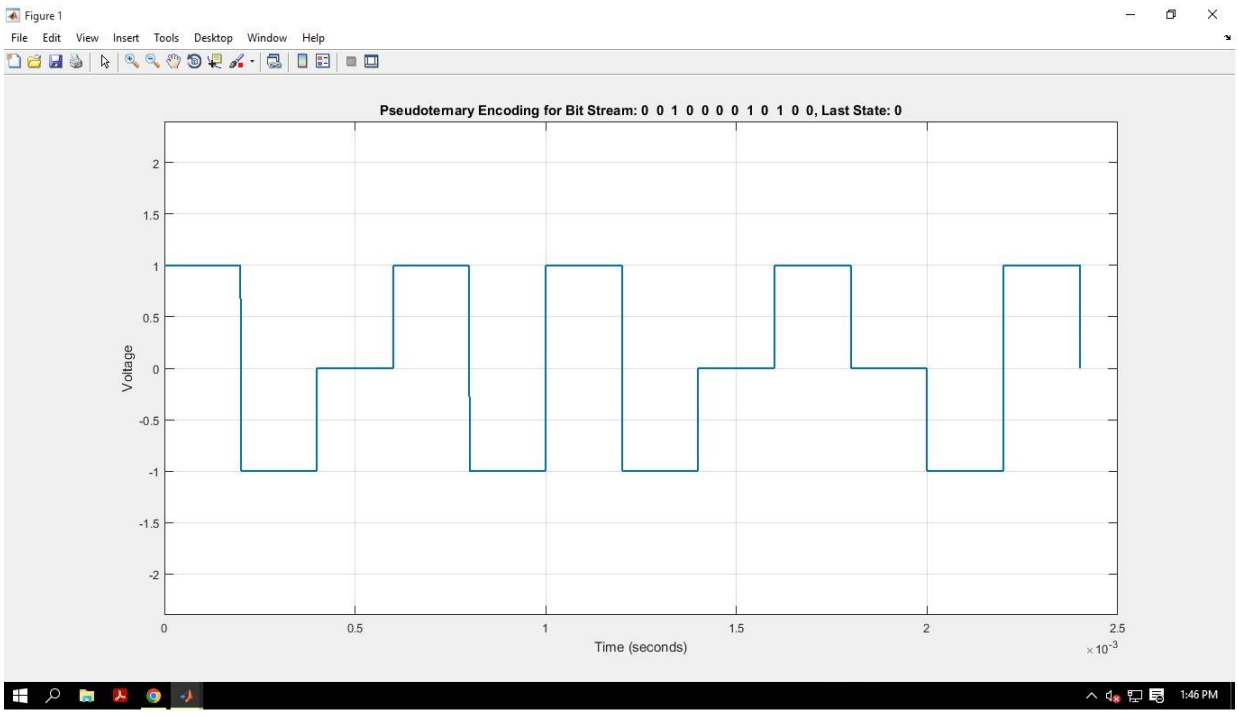

Figure 3: Simulation Graph for Pseudoternary

MLT-3

```
%22-47154-1
%AB-CDEFG-H
%BEG = 214
%Binary  = 0010 0001 0100

clc
clear all
close all
bit_stream = [0 0 1 0 0 0 0 1 0 1 0 0];
 no_bits = length(bit_stream);
bit_rate = 10000; % 10 kbps pulse_per_bit = 1;
% for MLT-3
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)*2; % sampling interval
% total duration = (no_pulse)*(pulse_duration)
no_samples = length(t); % total number of samples
dig_sig = zeros(1,no_samples);
max_voltage = 4;
min_voltage = -4;
last_bit=0;
last_non0_bit=1;
for i = 1:no_bits
   j = (i-1)*2;
   if bit_stream(i) == 1
     if(last_bit==1)
                dig_sig(((i-1)*(samples_per_pulse)+1):i*(samples_per_pulse)) =
                0*ones(1,samples_per_pulse);
              last_bit = 0;
       else
            if(last_bit==-1)
                dig_sig(((i-1)*(samples_per_pulse)+1):i*(samples_per_pulse))
                0*ones(1,samples_per_pulse);
              last_bit = 0;
          else
              if(last_non0_bit == 1)
                    dig_sig(((i-1)*(samples_per_pulse)+1):i*(samples_per_pulse)) =
                    min_voltage*ones(1,samples_per_pulse);
                 last_bit = -1;
```

```
                else
                    dig_sig(((i-1)*(samples_per_pulse)+1):i*(samples_per_pulse)) =
                    max_voltage*ones(1,samples_per_pulse);
                last_bit =1;
            end
            last_non0_bit  =  last_non0_bit*-1;
        end
    else
        if(last_bit==1)
            dig_sig(((i-1)*(samples_per_pulse)+1):i*(samples_per_pulse)) =
            max_voltage*ones(1,samples_per_pulse);
        else
            if(last_bit==-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))  =
            0*ones(1,samples_per_pulse);
            end
        end
    end
plot(t,dig_sig,'linewidth',1.5)
grid on
xlabel('time in seconds')
ylabel('Voltage')
xlim([0,(no_bits)/bit_rate])
ylim([(min_voltage - (max_voltage)*0.2),(max_voltage+max_voltage*0.2)])
title(['MLT-3 for ',num2str(bit_stream),''])
```

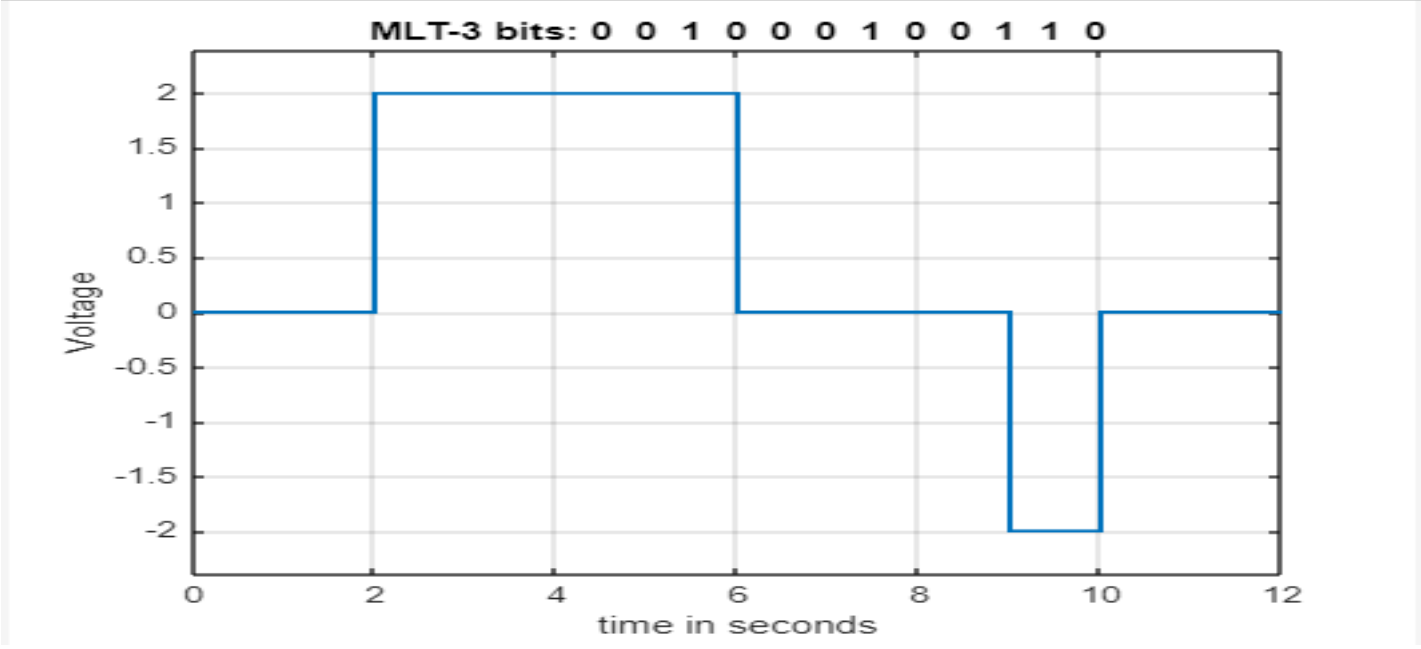

Figure 4: Simulation Graph for MLT-3

Discussion:

After the codes were placed in the script the graphs shown above were shown. There are a few things that were taken by self-consideration. For the cases of provided codes the last bit (before the given signal) is Zero and the last non-zero bit is Positive.

Conclusion:

This OEL report is about the concept of line coding schemes, essential for converting digital data into analog signals suitable for communication channels. The experiment's objectives included using MATLAB to solve communication engineering problems and developing an understanding of Digital-to-Digital Conversion (Line Coding). The report outlined key line coding schemes, including Polar NRZ-I, Polar Biphase Differential Manchester, Pseudoternary and MLT-3. The diagrams provided (Figures 1 to 4) visually demonstrated the success of the experiment in generating simulation graphs for various line coding schemes. The experiment highlighted the significance of appropriate line coding selection in achieving reliable and efficient digital communication systems.

Reference:

[1] The MathWorks, Inc. “MATLAB The Language of Technical Computing” mathworks.com. Accessed: June 20, 2023. [Online]. Available: https://www.mathworks.com/help/matlab/index.html

[2] B. A. Forouzan, "Data Communication and Networking," Tata McGraw-Hill, 2005.