Communication Project Report

- **Course:** communication
- **Instructor:** Dr.Amr wagih
- **Project name:** RZ
- **Academic year:** 2^{nd} year
- **Department:** Computers and systems engineering

Team members:

NAME	SECTION
Shrouk Mohammed Mahmoud Sabry	2
Rehab Mohammed Ahmed Lotfy	2
Farah Maurice Wanis Rizkallah	2
Maryem Ehab Mostafa Rabie	4
Hoda Mohamed Abdelatif Ahmed	4

Communication Project Report

Project objective

This project aims to convert the NRZ(L) to RZ (encoding) and also to return it back from RZ to NRZ(L) (decoding) and this operations can done by both MATLAB code (software) and by circuit done on multisim which is a simulation that could by implemented in real time hardware.

MATLAB code for encoding:

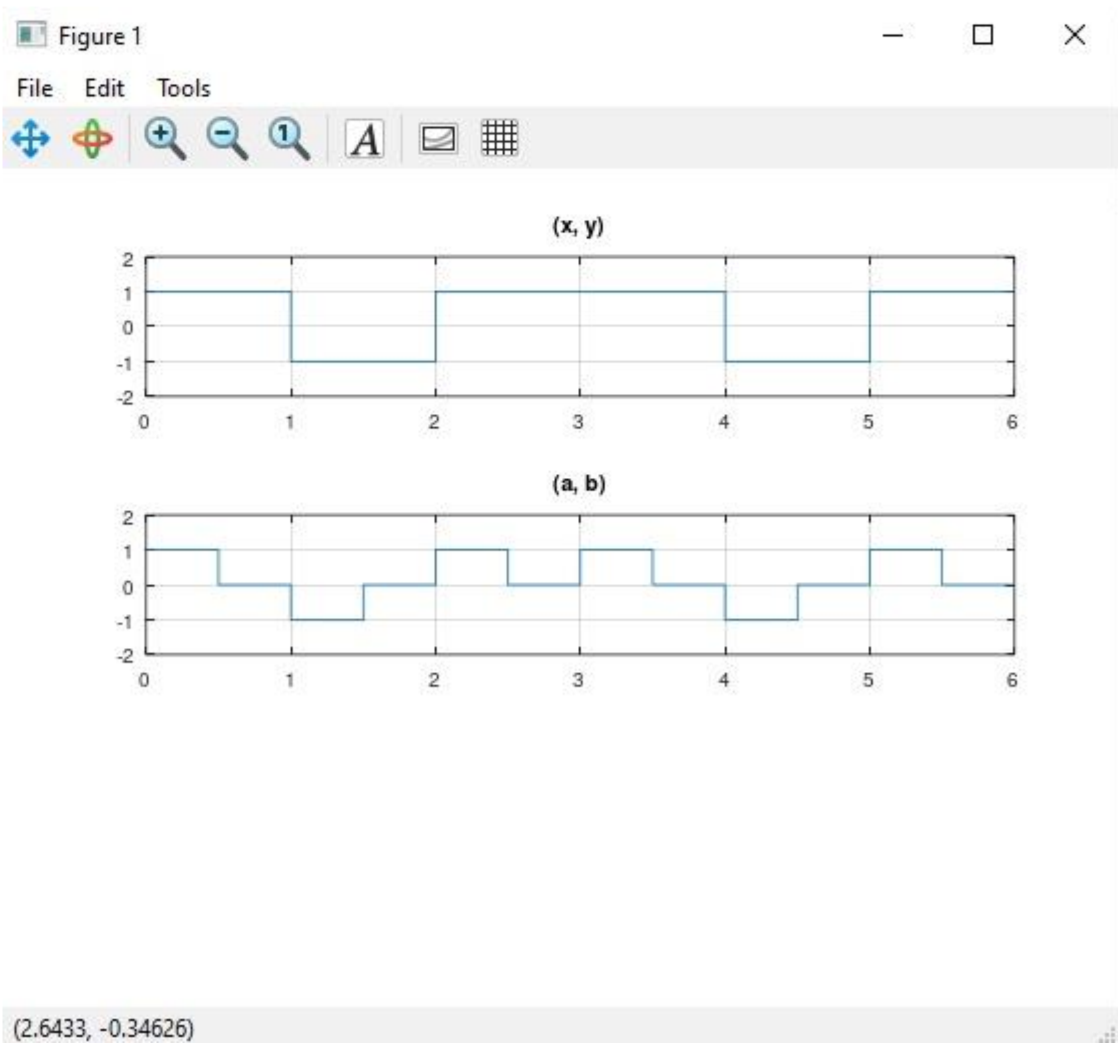
```
input_stream = input('Enter Stream');
%input_stream = [1, 0, 1, 1, 0, 1];
n = length(input_stream);
x = [];
y = [];
a = [];
b = [];
c = [];
for i = 1 : n
 x=[x i-1 i];
 if( input_stream(i) == 0)
 y = [y -1 -1];
 else
 y = [y 1 1];

 end
end
for i = 1:n
 a = [a i-1 i-0.5 i-0.5 i];
 if(y(i * 2) == -1)
 b = [b -1 -1 0 0];
 else
 b = [b 1 1 0 0];
 end
end
% Create figure window
figure;
% Plot first graph (x, y)
subplot(3, 1, 1);
plot(x, y);
```

```
axis([0, n, -2, 2]);
grid on;
title('(x, y)');
% Plot second graph (a, b)
subplot(3, 1, 2);
plot(a, b);
axis([0, n, -2, 2]);
grid on;
title('(a, b)');
```

input stream :

```
>> communication
Enter Stream[1 0 1 1 0 1]
```

Output Of code:



MATLAB code for decoding:

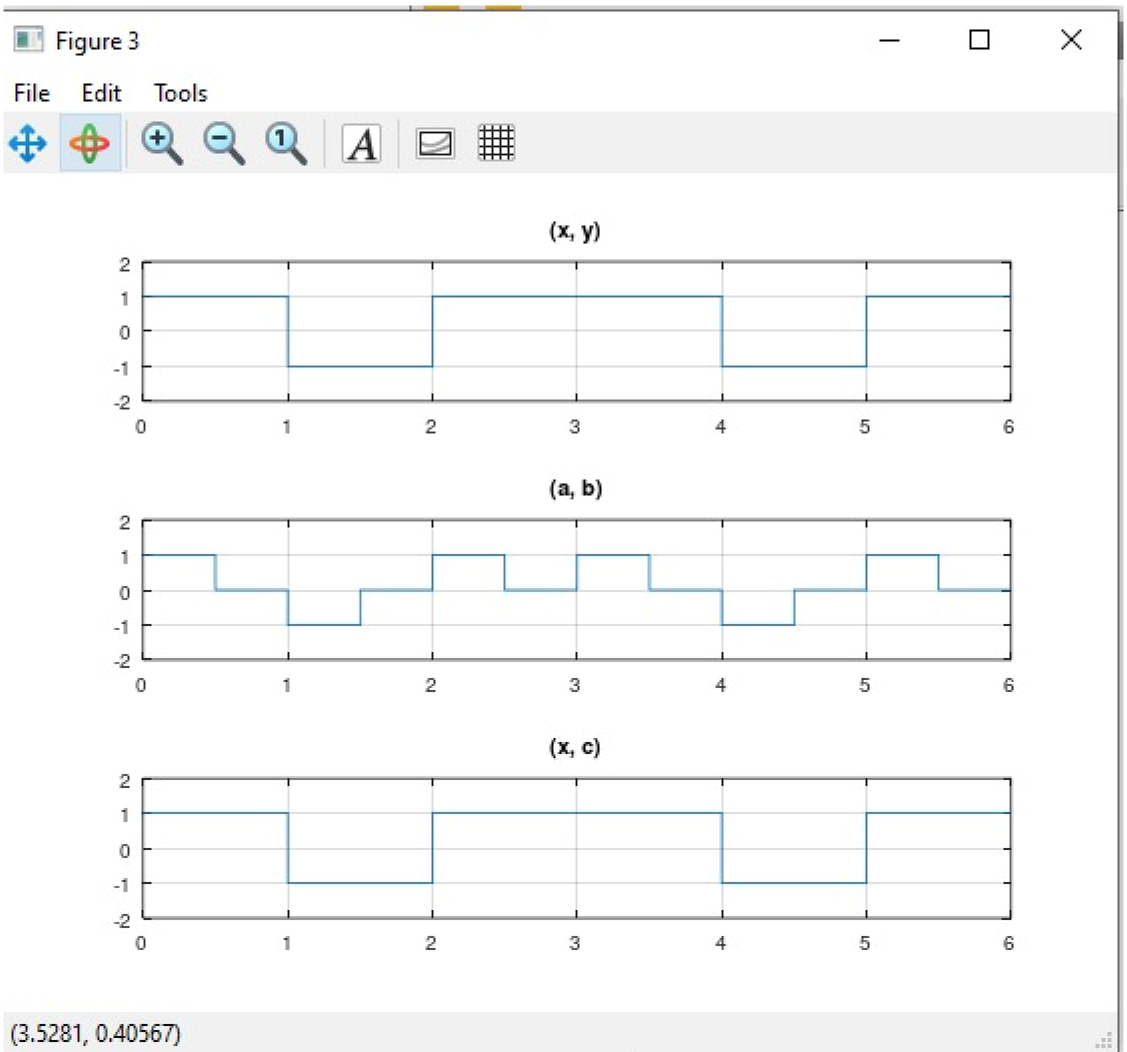
```
input_stream = input('Enter Stream');
%input_stream = [1, 0, 1, 1, 0, 1];
n = length(input_stream);
x = [];
y = [];
a = [];
b = [];
c = [];
for i = 1 : n
 x=[x i-1 i];
 if( input_stream(i) == 0)
 y = [y -1 -1];
 else
 y = [y 1 1];

 end
end
for i = 1:n
 a = [a i-1 i-0.5 i-0.5 i];
 if(y(i * 2) == -1)
 b = [b -1 -1 0 0];
 else
 b = [b 1 1 0 0];
 end
end
for i = 1:4*n
        if (b(i)~=0)
        c = [c b(i)];
    end
end

% Create figure window
figure;

% Plot first graph (x, y)
subplot(3, 1, 1);
plot(x, y);
axis([0, n, -2, 2]);
```

```
grid on;
title('(x, y)');

% Plot second graph (a, b)
subplot(3, 1, 2);
plot(a, b);
axis([0, n, -2, 2]);
grid on;
title('(a, b)');

% Plot third graph (x, c)
subplot(3, 1, 3);
plot(x, c);
axis([0, n, -2, 2]);
grid on;
title('(x, c)');
```

Input stream :

```
>> communication
Enter Stream[1 0 1 1 0 1]
```

Output Of code:



Explanation of encoding code:

This code takes an input stream, represented by a binary sequence of 0's and 1's, and generates three graphs based on the sequence.

In the first for loop, the code initializes empty arrays x and y. The loop then iterates through each element in the input stream. For each element, the code adds two values to the x array (i-1 and i), and two values to the y array. If the element is 0, it adds -1 twice to y. If the element is 1, it adds 1 twice to y. Essentially, this loop transforms the input stream into two arrays, x and y, that represent a piecewise linear function.

In the second for loop, the code initializes empty arrays a and b. The loop iterates through each element in the input stream. For each element, the code adds four values to the a array (i-1, i-0.5, i-0.5, and i), and four values to the b array. If the corresponding value in y is -1 (which was added twice in the previous loop when the input was 0), the code adds -1 twice and 0 twice to the b array. If the corresponding value in y is 1 (which was added twice in the previous loop when the input was 1), the code adds 1 twice and 0 twice to the b array. Essentially, this loop transforms the x-y function into two arrays, a and b, that represent a piecewise linear function.

The code then creates a figure window and plots three graphs using the subplot function. The first graph plots the x-y function, the second graph plots the a-b function. two graphs have the same x-axis (0 to n) and y-axis (-2 to 2), and have a grid and title.

Overall, this code is a simple example of how to generate and plot piecewise linear functions from a binary input stream.

Explanation of decoding code:

This code takes an input stream, represented by a binary sequence of 0's and 1's, and generates three graphs based on the sequence.

In the first for loop, the code initializes empty arrays x and y. The loop then iterates through each element in the input stream. For each element, the code adds two values to the x array (i-1 and i), and two values to the y array. If the element is 0, it adds -1 twice to y. If the element is 1, it adds 1 twice to y. Essentially, this loop transforms the input stream into two arrays, x and y, that represent a piecewise linear function.

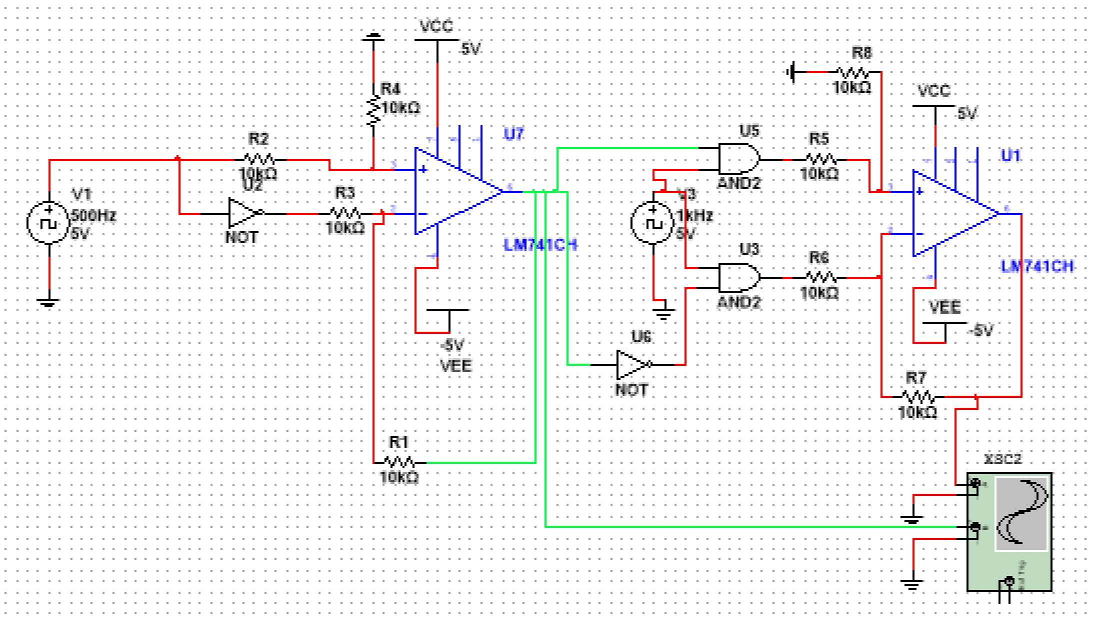
In the second for loop, the code initializes empty arrays a and b. The loop iterates through each element in the input stream. For each element, the code adds four values to the a array (i-1, i-0.5, i-0.5, and i), and four values to the b array. If the corresponding value in y is -1 (which was added twice in the previous loop when the input was 0), the code adds -1 twice and 0 twice to the b array. If the corresponding value in y is 1 (which was added twice in the previous loop when the input was 1), the code adds 1 twice and 0 twice to the b array. Essentially, this loop transforms the x-y function into two arrays, a and b, that represent a piecewise linear function.

In the third for loop, the code initializes an empty array c. The loop iterates through each element in the b array. If the element is not 0, it adds the element to the c array. This loop removes any 0's that were added in the previous loop.

The code then creates a figure window and plots three graphs using the subplot function. The first graph plots the x-y function, the second graph plots the a-b function, and the third graph plots the x-c function. All three graphs have the same x-axis (0 to n) and y-axis (-2 to 2), and have a grid and title.

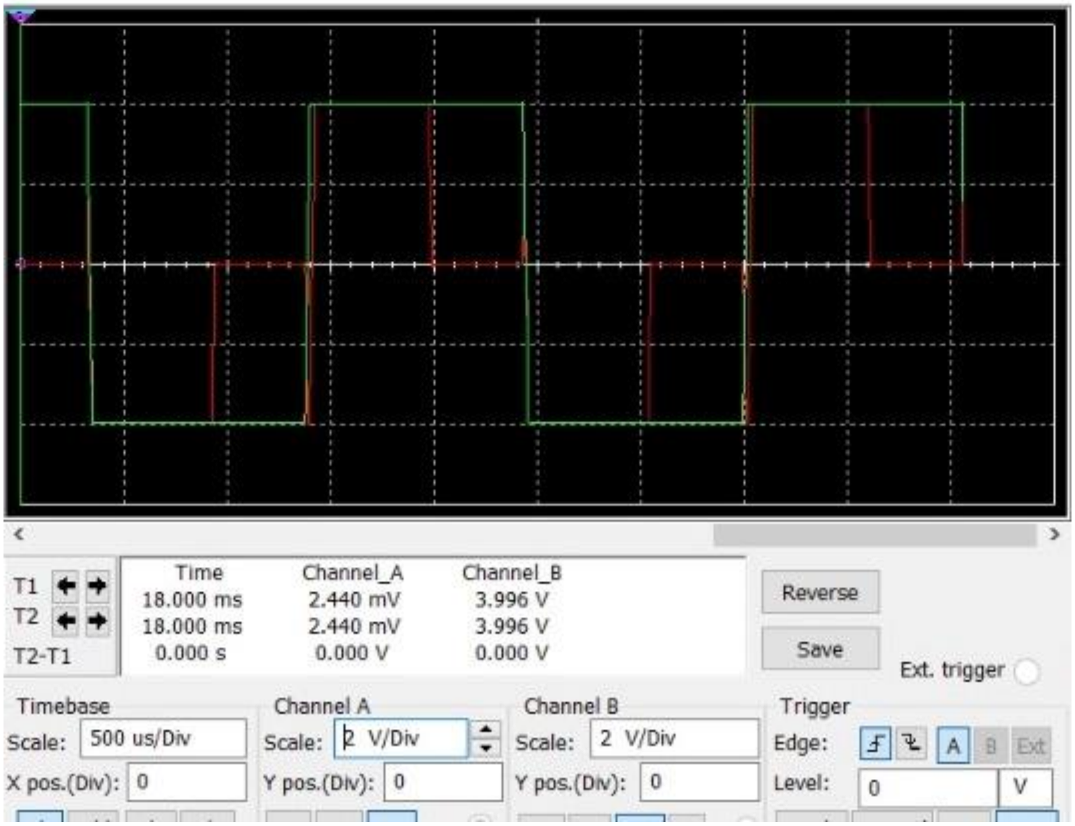
Overall, this code is a simple example of how to generate and plot piecewise linear functions from a binary input stream.

The multisim circuit for encoding code:



Then we replace the input pulse and clock with a 555 to make it implemented in a real time hardware

Output of circuit



Description of encoding circuit:

The circuit works as following :

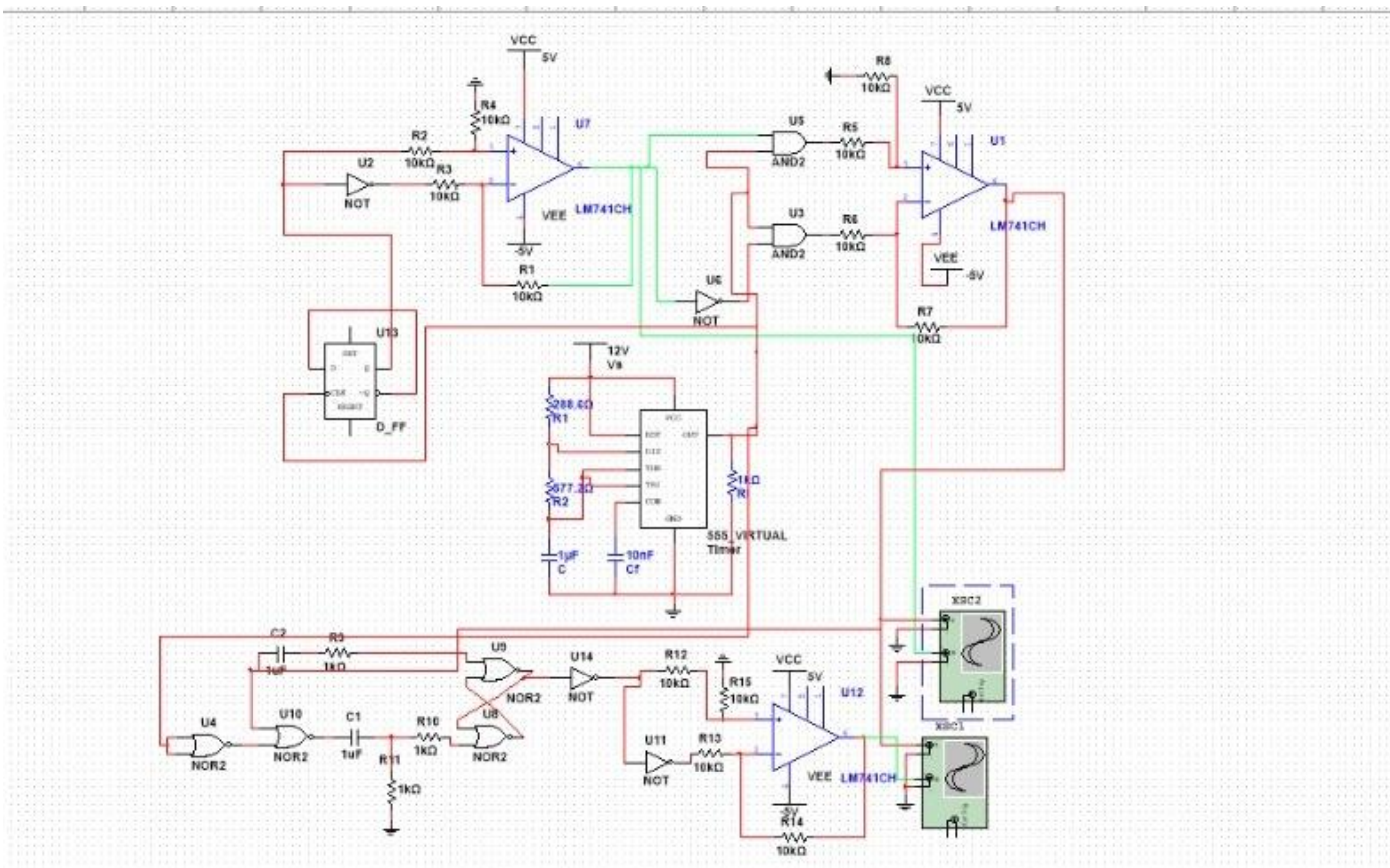
First the entered signal is 1 and the opamp has two inputs first one is not gate which invert the 1 entered to become zero and the second input of opamp is the entered signal which is one

The opamp works as a comparator which compare the inputs if the signal is 1 the output will be one , if the input is zero the output will be -1 , by this way we generate NRZ(L).

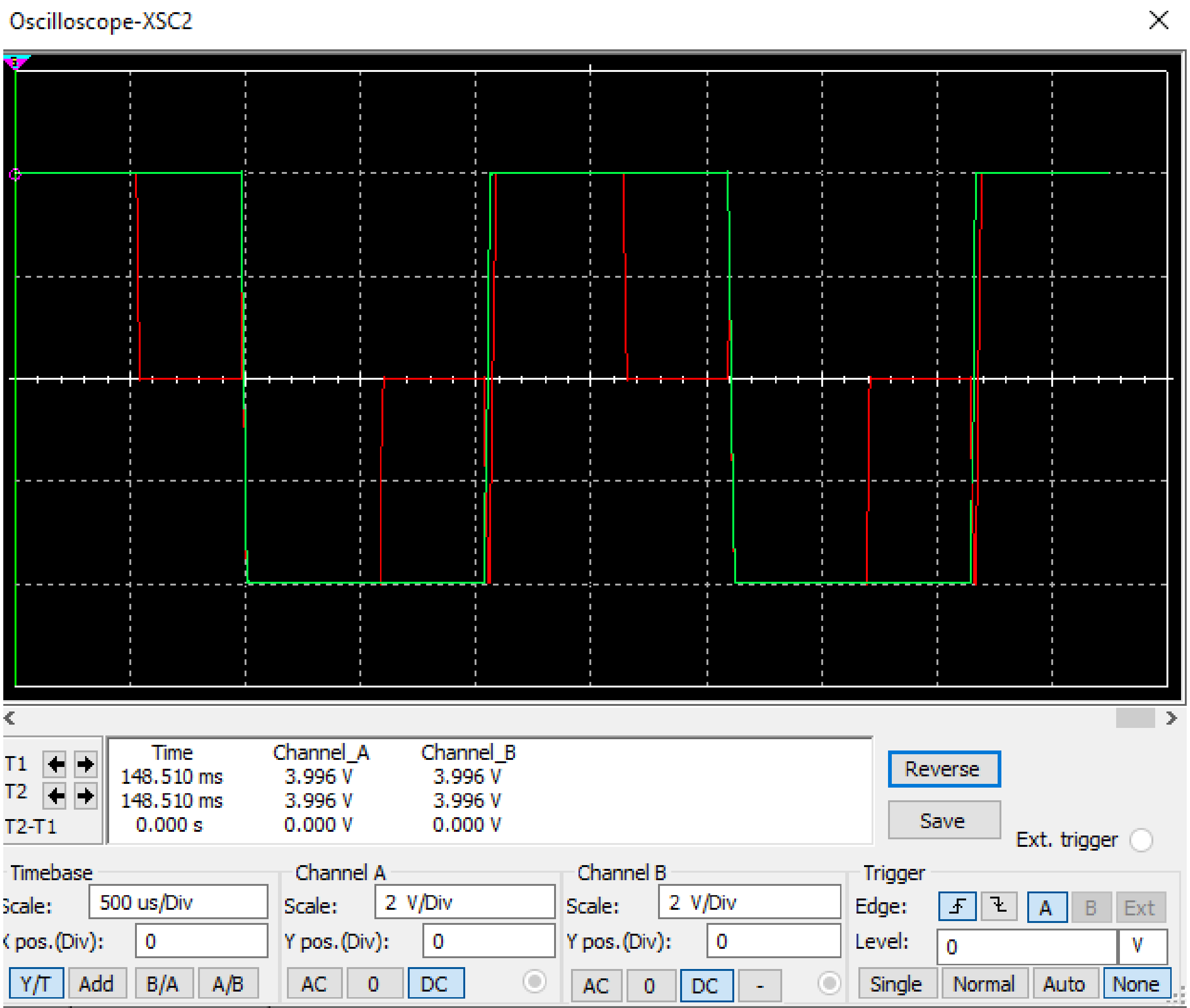
Then the output of the opamp is anded with clock which has a period half the period of entered data

The second opamp has two inputs one is the clock anded with the output of the first opamp and the second input is the clock anded with the inverted output of the first opamp then the second opamp will act as comparator which will compare if its input = 0 the output will be -1 , if its input = 1 then the output will be 1 and by this way we made the RZ

The multisim circuit for decoding code:



Output of decoding circuit



Description of decoding circuit:

The polar RZ signal is provided as the input to the decoder circuit. This signal consists of voltage pulses with varying polarity to represent binary data.

The decoder circuit receives the RZ signal as input and decodes it to generate the NRZ signal. The decoder's purpose is to convert the polar RZ signal into the corresponding NRZ format.

The decoded NRZ signal is produced as the output of the decoder circuit. This signal will have two voltage levels, typically representing binary ones and zeros.

The comparator circuit compares the voltage of the RZ signal with a reference voltage and generates the corresponding NRZ output based on the comparison. If the RZ signal voltage is higher than the reference voltage, the comparator outputs a high voltage (representing binary one), and if the RZ signal voltage is lower than the reference voltage, the comparator outputs a low voltage (representing binary zero).

By repeating this comparison for each pulse in the RZ signal, the decoder circuit converts the polar RZ signal into the desired NRZ format.

Communication Project Report