**Assignment No. 2**

**Meet Patel**

**( B00899516 )**

**CSCI 6704: Advance Topics in Networks**

**Dr. Srini Sampalli**

**Dalhousie University**

**Fall 2022**

# **Programming Exercise 1**

Write a program that accepts a binary string as input and generates waveforms using a GUI corresponding to the following encodings of the input binary string:

a) Unipolar

b) NRZ

c) Manchester

### **Sample Run for Unipolar Encoding**

Input: 00110100010 Output: Shown in Figure1

Graphical user interface, text, application

Description automatically generated

Figure : Sample Run for Unipolar Encodings

### **Sample Run for Non-return-to-zero (NRZ) Encoding**

Input: 00110100010 Output: Shown in Figure2

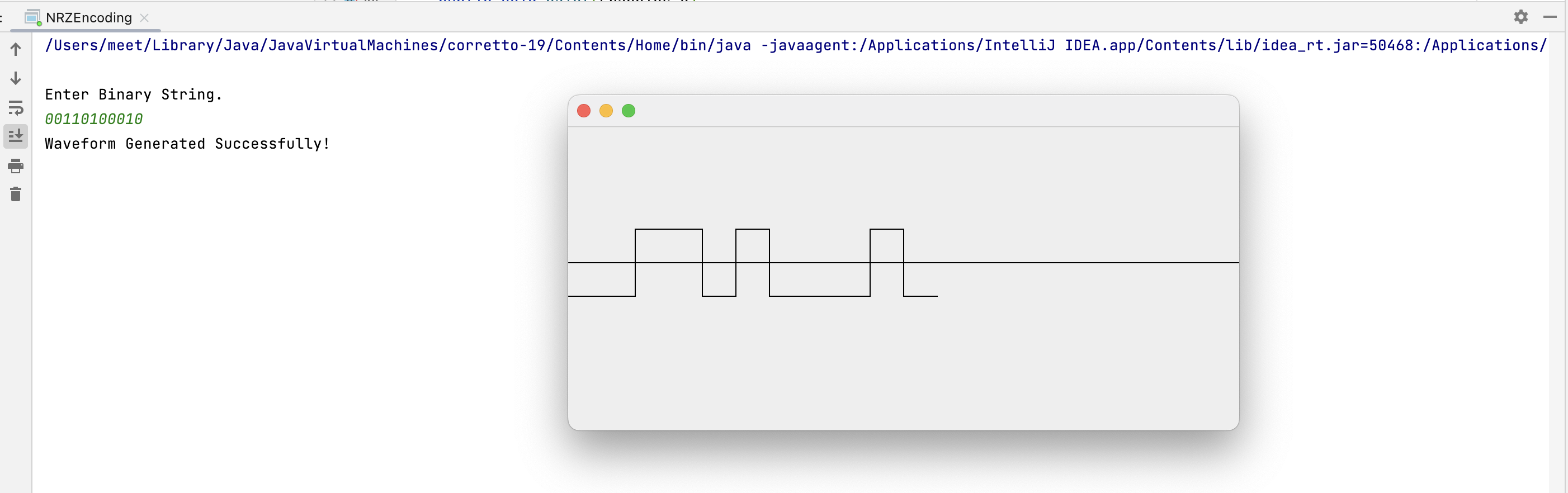


Figure : Sample Run for Non-return-to-zero (NRZ) Encoding

### **Sample Run for Manchester Encoding**

Input: 00110100010 Output: Shown in Figure3

A picture containing text

Description automatically generated

Figure : Sample Run for Manchester Encoding

# **Source Code**

The source code for **Waveform Generation Program** is saved in package “**exercise\_1\_encodings**”.

It contains three Java files which are listed below:

1. **ManchesterEncoding** : Responsible for generating Manchester Waveform
2. **NRZEncoding** : Responsible for generating NRZ Waveform
3. **UnipolarEncoding** : Responsible for generating Unipolar Waveform

# **Programming Exercise 2 <Bit Stuffing Program>**

In this exercise, you will be writing a simple program that does the following:

a) Read a String of hex digits

b) Convert the String into a String of binary numbers

c) Perform bit stuffing on the binary String

d) Unstuff the bits from the binary String

e) Produce the original hex String

### **Sample Run 1**

Enter hexadecimal string.

**ABEFFFF**

Input: ABEFFFF

Conversion to binary: 1010101111101111111111111111

After bit stuffing: 10101011111001111101111101111101

After bit unstuffing: 1010101111101111111111111111

Output: ABEFFFF

Graphical user interface, text, application, email

Description automatically generated

Figure : Output of Bit Stuffing Program for input hexadecimal string ABEFFFF

### **Sample Run 2**

Enter hexadecimal string.

**ABCDEF**

Input: ABCDEF

Conversion to binary: 101010111100110111101111

After bit stuffing: 101010111100110111101111

After bit unstuffing: 101010111100110111101111

Output: ABCDEF

Graphical user interface, text, application, email

Description automatically generated

Figure : Output of Bit Stuffing Program for input hexadecimal string ABCDEF

### **Sample Run 3**

Enter hexadecimal string.

**AFFFEEEFF**

Input: AFFFEEEFF

Conversion to binary: 101011111111111111101110111011111111

After bit stuffing: 1010111110111110111110011101110111110111

After bit unstuffing: 101011111111111111101110111011111111

Output: AFFFEEEFF

Graphical user interface, text, application, email

Description automatically generated

Figure : Output of Bit Stuffing Program for input hexadecimal string AFFFEEEFF

# **Source Code**

The source code for **Bit Stuffing Program** is saved in package “**exercise\_2\_bit\_stuffing**”.

It contains three JAVA files which are listed below:

1. **BitsConversion** – This java file is responsible for the conversion of Hexadecimal Input to the Binary Bits and then conversion of Binary Bits to Hexadecimal Output.
2. **BitsStuffingAndUnstuffing** – This Java file is responsible for stuffing and unstuffing 1’s
3. **BitsMain** – This Java file is responsible for taking the input from the user and then passing it to the another class methods for further processing.