University of science and technology at Zewailcity

Pulse code modulation communication system

Project Part 2

Instructor: DR. Samy Soliman

Khaled Osama **201600515**

Mustafa Elsayed **201600848**

Hatem Yahia **201600514**

Ahmed Aboulela **201600566**

Introduction

The project is mainly about creating a communication system that uses PCM. the system consists of the sampler, quantizer, encoder, decoder, and reconstruction filter. Let's talk briefly about each one. The sampler takes the continuous input and takes discrete values after a specified interval this role is governed by specific rules that we will illustrate below. The second part is the quantizer, it mainly has amplitude levels and maps each discrete value that comes from the quantizer to a certain level. The encoder helps create a continuous waveform to be transmitted it has a lot of methods to create this waveform like Manchester and PNRZ. In the other side of the channel, we have a decoder which will decode the coming waveform to its analog shape and after that, it will pass it on a reconstruction filter. This filter is a low pass filter which helps attenuate the higher frequencies and the smooth the signal amplitude.

Sampling

This is the first component of our simulated digital communication system. We first have a continuous function (sampled with very high frequency since we are using MatLab) and we want to convert it into a discrete signal by sampling. The idea used in this function is, there is a ratio between the sampling rate we want to apply and the sampling rate of input function (the not really continuous function). Once we have the ration x, we will take a sample once after each x samples from the original signal. An important note to notice here is the relation between the Nyquist rate (from the frequency of the original signal, not related to its MatLab sampling) and the sampling frequency we are applying as the Nyquist rate needs to be higher or we will find a distortion.

We will have 3 examples each with the function $m(t) = 5cos(2\pi t)$, where fs is 10, hence it has a Nyquist frequency of 20. In figure (1), the sampling frequency is 40 which is much higher

than the Nyquist frequency. In figure (2) we will sample with the Nyquist frequency which is 20. There will be a distortion in figure (3) as we are sampling with half of the Nyquist frequency.

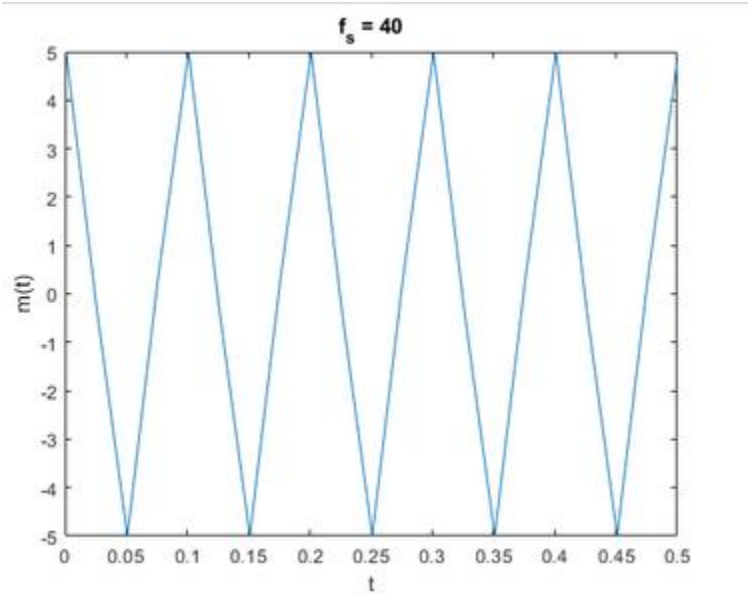

Figure 1 sampling with 2 Nyq_f

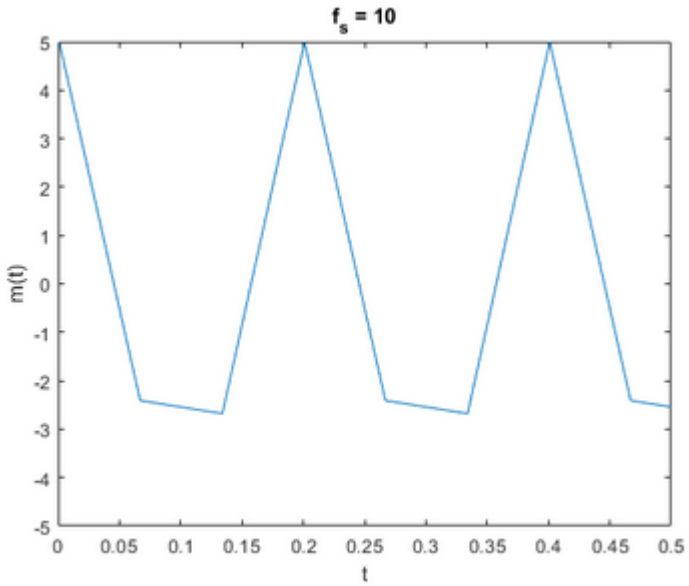

Figure 3 sampling with half Nyq_f

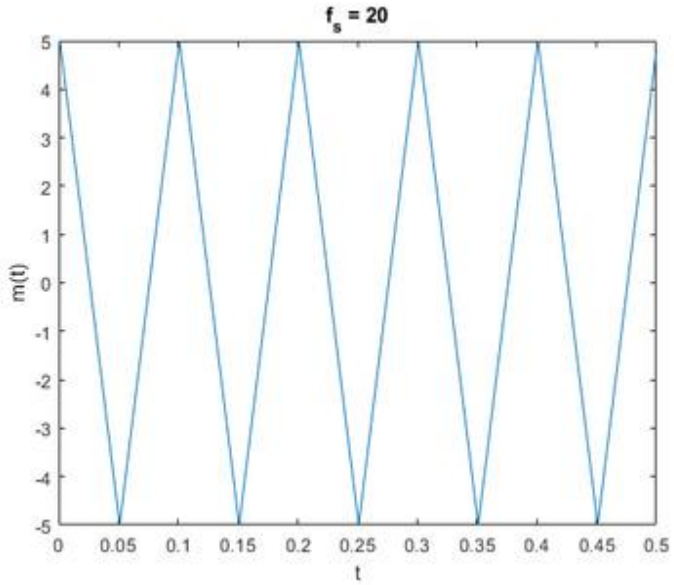

Figure 2 sampling with Nyq_f

Quantizer

In this function, we are trying to give a digital representation to the amplitude of the function instead of the continuous range. We are limited to L levels. So, we remap the amplitudes from mp to -mp to be instead from -L/2 to L/2. Actually, we then remap that to be 0 to L-1 as it will be more convenient when converted to binary. There is an exception when there is a nonzero mu, we need to get the mu-value of the amplitude from mu-law, while the rest of the steps will hold as they are. After we have a level for each time, we need to map that into a binary word with each of length equal to log2(L). We replace each analog value with the b-length word of bits.

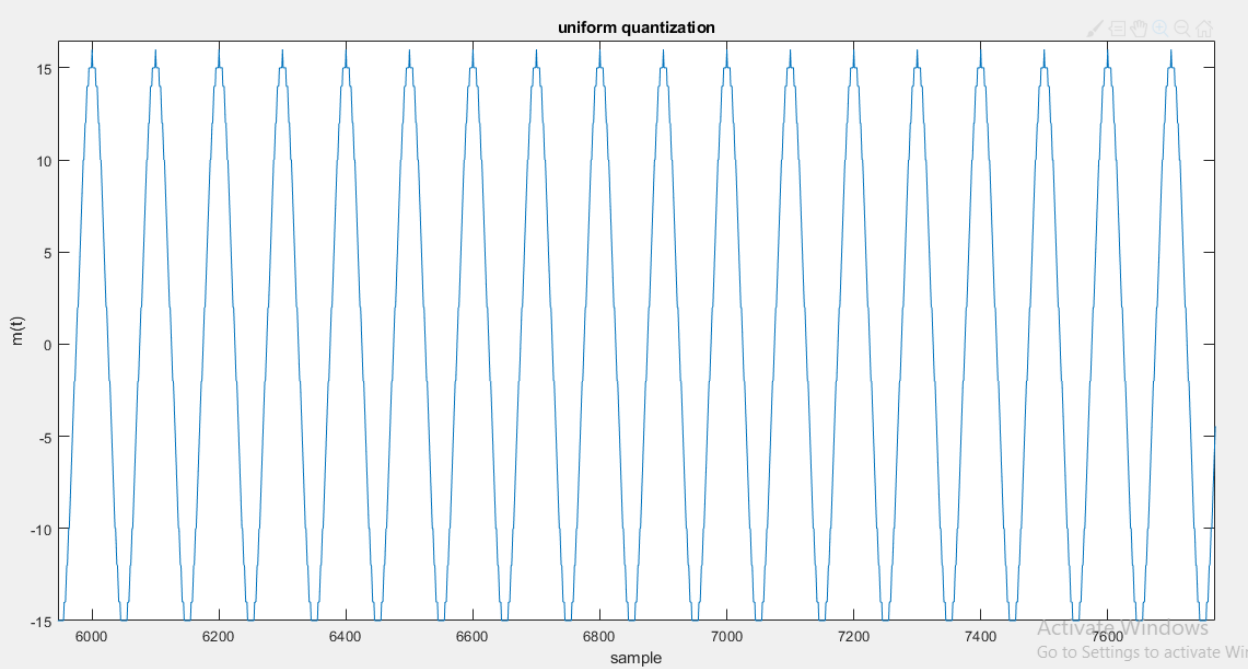

Figure 4 a sample of uniform quantization of m(t) over L=32

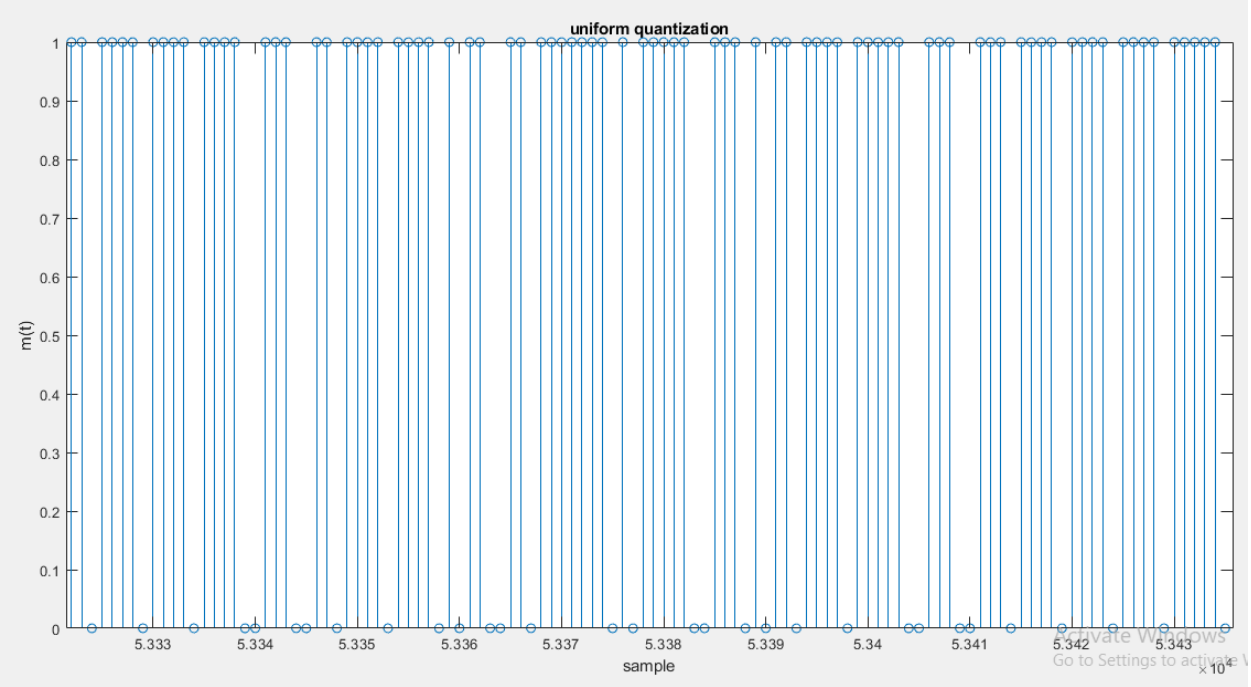

Figure 5 the binary representation of the function

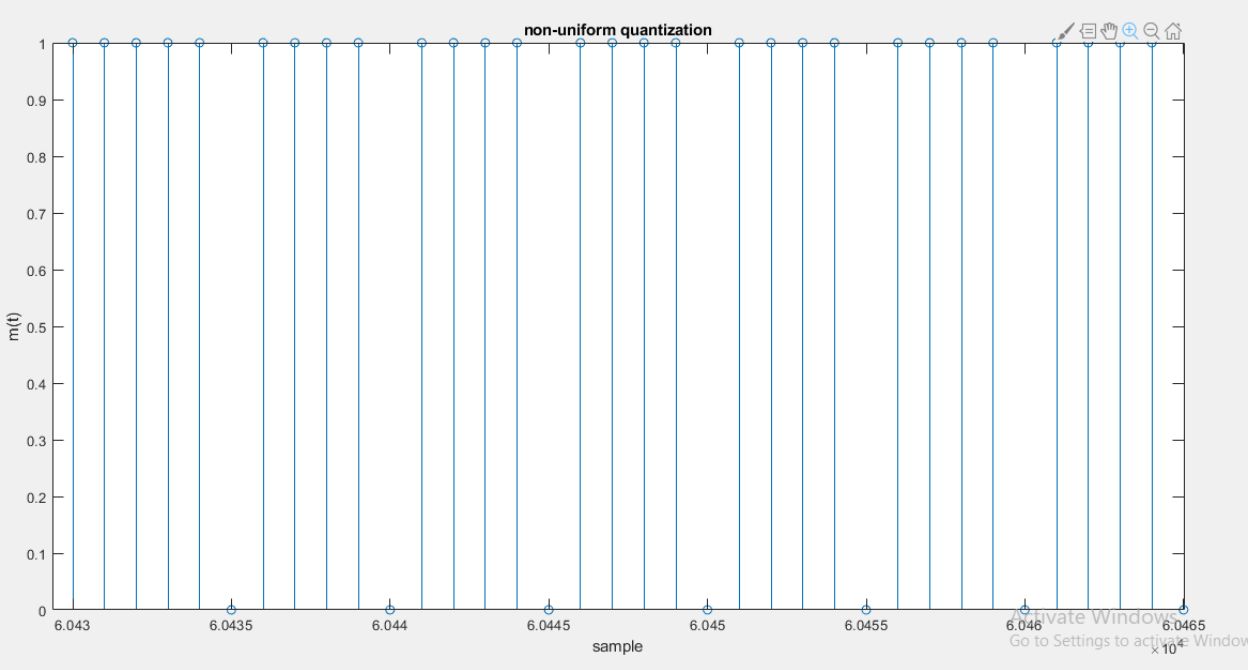

Figure 6 another binary representation of a non-uniform function

Encoder:

The encoder is the module that transforms the bits coming out of the quantizer into a signal with a certain amplitude so that it can be transmitted using the normal transmission procedures. There are numerous types of encoding, they vary in the way they represent the zeros and ones and the properties resulting from those representations i.e. (the average power, the complexity of the receiver and synchronization). We will focus mainly on three types of encoding: Manchester, Polar NRZ and unipolar NRZ.

- The encoder we made has 3 mainline codes
  1- Unipolar nonreturn to zero
  2- Bipolar nonreturn to zero
  3- Manchester
- We get from the user the following parameters:
  1- The information array.
  2- The duration of each bit of information.
  3- The sampling frequency.
  4- The amplitude of the output.
  5- The type of encoding**.**
- Note: if the number of points representing a bit is odd, we make the first half of the pulse gets the extra point in the Manchester encoding, in order to maintain consistency.

Test case:

- We have used a test case to test the encoder alone with:
  1- Fs =10.
  2- bit duration = 0.0001.
  3- information array = [0 1 1 0 1 0 0 1].
  4- amplitude =1.

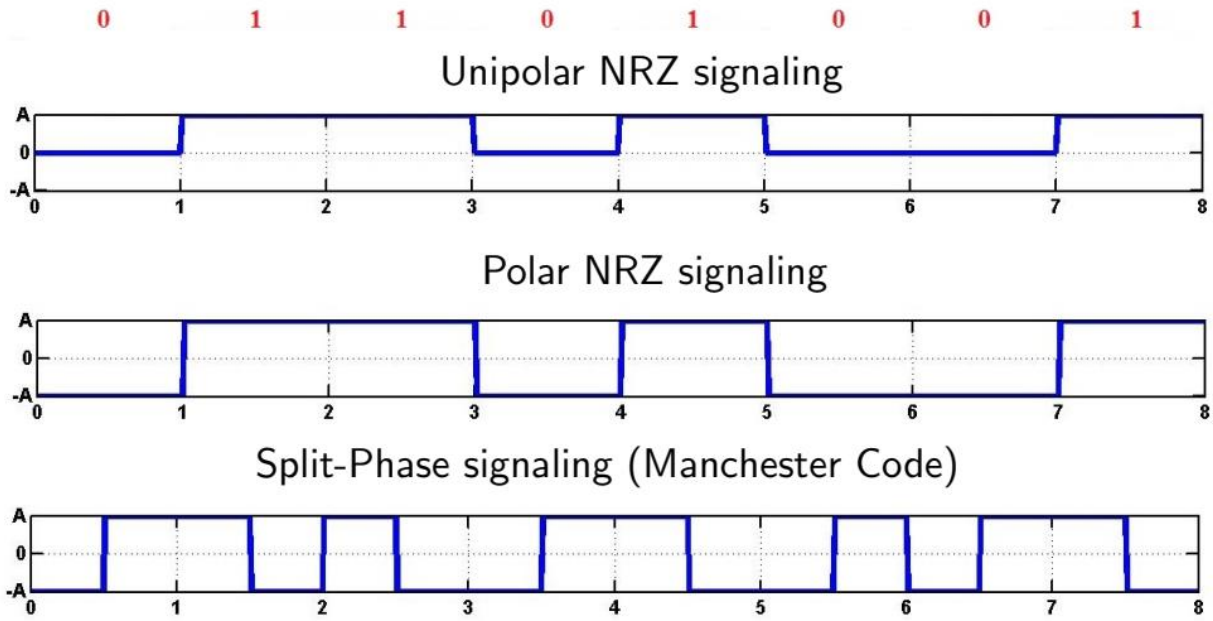

Figure 7 this is the example used to test the function along with the expected output

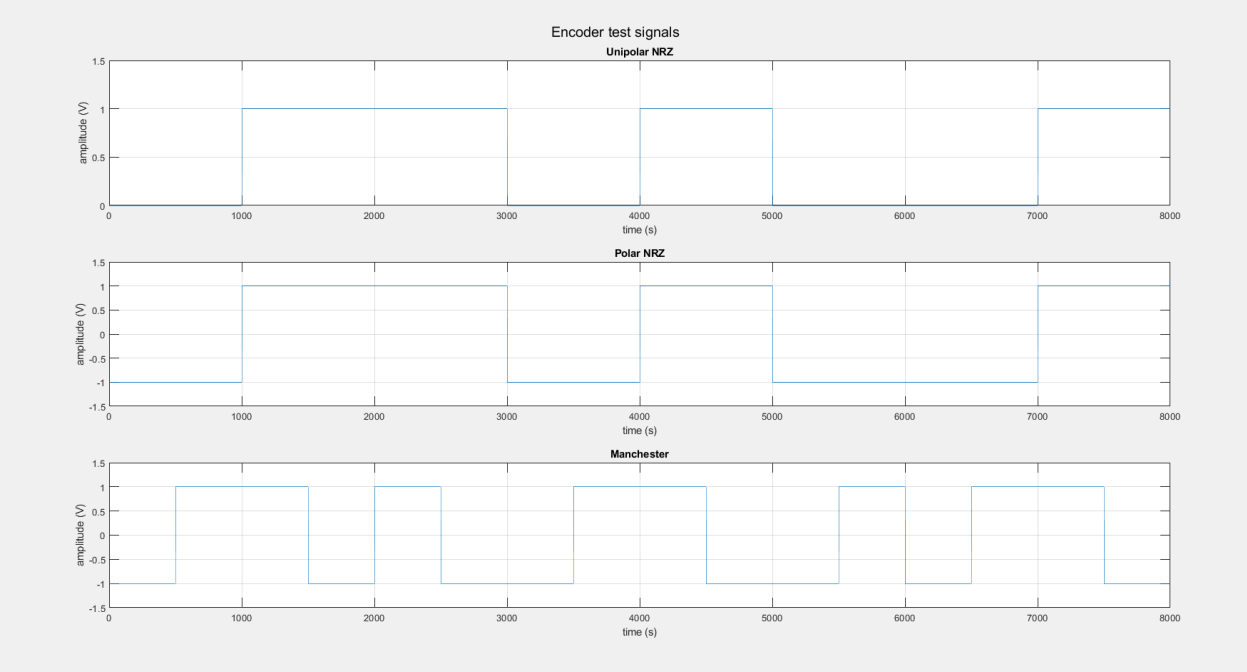

Figure 8 the output of the function, we can observe that it matches the expected results

Decoder:

The decoder is the first block in the receiver module. It is responsible for translating the zeros and ones coming from the encoder, and combining each set of bits and transferring it into the values specified by the quantizer.

- The decoder we made has can translate 3 mainline codes
  1- Unipolar nonreturn to zero
  2- Bipolar nonreturn to zero
  3- Manchester
- We get from the user the following parameters:
  1- The information array.
  2- The number of points per bit.
  3- The number of bits per sample.
  4- The type of encoding**.**

  Test case:

- We have used the output of the encoder's test case as the input for our decoder, we have only need to set the number of bits per sample:
  1- Fs =10.
  2- bit duration = 0.0001.
  3- information array = [0 1 1 0 1 0 0 1].
  4- amplitude =1.
  5- A number of bits per sample = 2.

  The output that has come out from the decoder is = [1,2,2,1]. Which matches the above input given that we treat every two bits as one number to be transformed from binary to decimal.

Reconstruction filter

In the reconstruction filter we pass the output of decoder through a low pass filter the attenuate the frequencies that are outside the bandwidth of the message. Below we passed $\sin(2\pi * 1000t) + \sin(2\pi * 3000t)$ and the figure below show the signal before and after being passed on the filter the photo on the right the that we have removed $\sin(2\pi * 3000t)$ from the incoming signal.

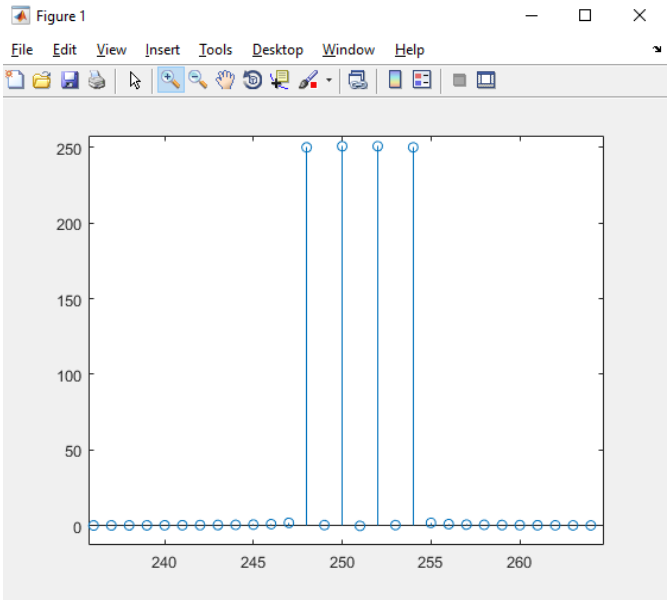

Figure 9

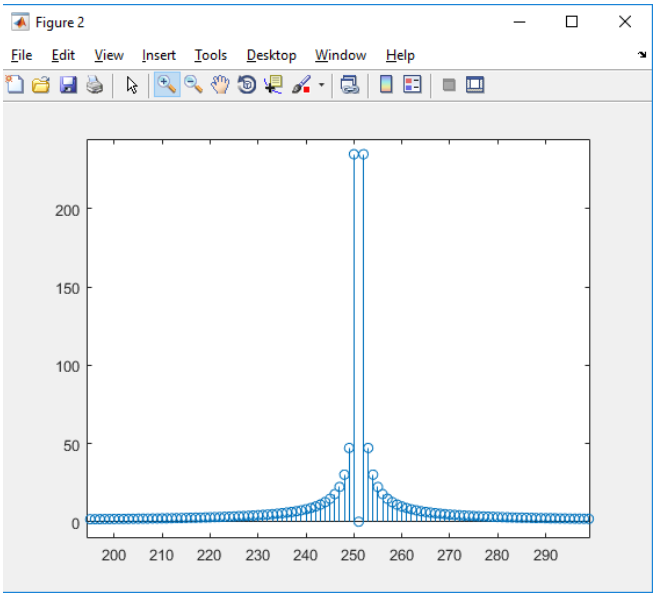

Figure 10