EE451 LAB REPORT

LAB 5 – ASK/FSK/PSK Modulation Techniques

Name : Harun Durmuş **Date:** 16.12.2023

Student Name : 270206025

Introduction

Digital modulation techniques used in telecommunications include Amplitude Shift Keying (ASK) and Frequency Shift Keying (FSK). ASK modulates data bits by varying the amplitude of the carrier signal, with different amplitudes representing different binary states (a high amplitude representing a '1' and a low amplitude representing a '0'). This method is simple, but it is susceptible to noise interference. FSK, on the other hand, alters the frequency of the carrier signal to transmit data, assigning different frequencies to represent binary '1' and '0'. This approach makes FSK more noise resistant than ASK. Both techniques are essential in digital communication, and they were chosen based on specific requirements such as noise immunity, bandwidth efficiency, and power constraints.

Procedure

5.1 B-ASK Modulation and Demodulation

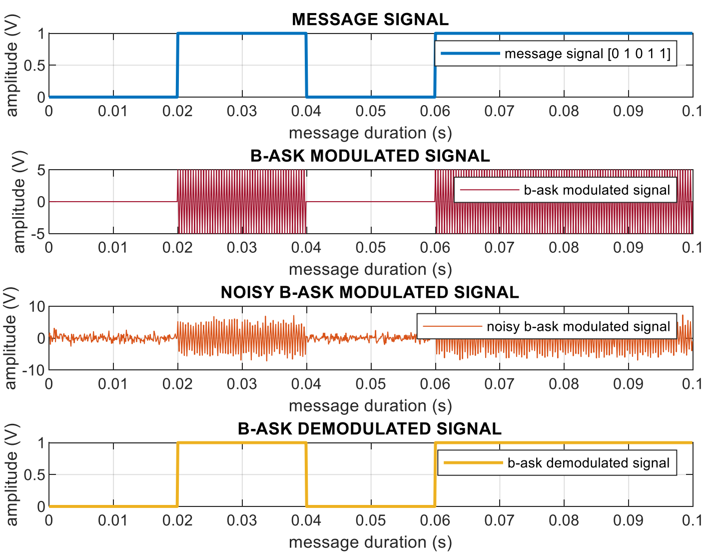

Figure 1: Applying B-ASK Modulation and Demodulation on the message signal

The demodulated signal is shown in the fourth subplot. Despite the presence of noise in the modulated signal, the demodulation process recovers the original binary data. The plot should closely resemble the original message signal, indicating the demodulation technique's effectiveness.

5.2 B-FSK Modulation and Demodulation

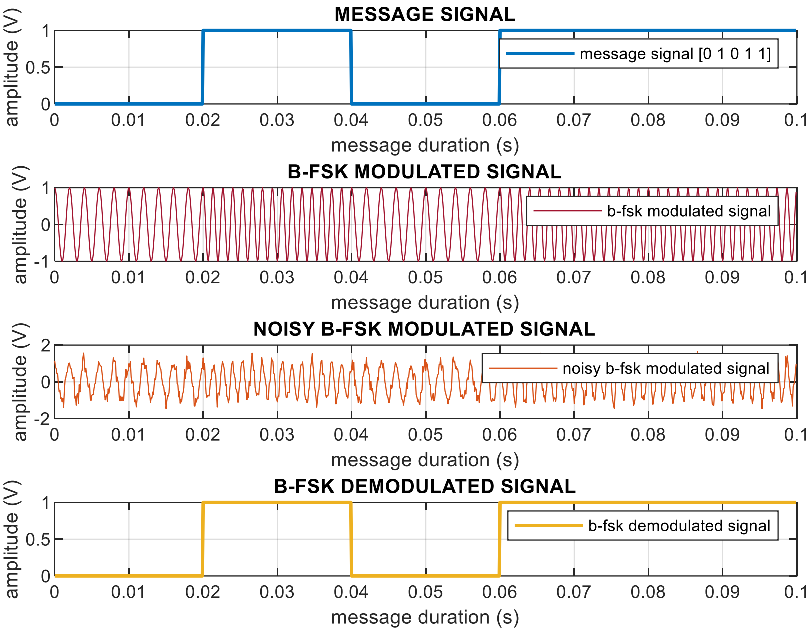

Figure 2: Applying B-FSK Modulation and Demodulation on the message signal

The demodulated B-FSK signal is shown in the fourth subplot. The goal, like with B-ASK, is to recover the original binary data from the noisy modulated signal. If the demodulation process is successful, the demodulated signal should match the original binary message.

5.3 Questions and Answers

Q1. What are the advantages and disadvantages of ASK and FSK?

A1. ASK (Amplitude Shift Keying) and FSK (Frequency Shift Keying) are two modulation techniques used in communication systems, each with their own set of benefits and drawbacks. The benefits of ASK are categorized as simplicity, bandwidth efficiency, and power consumption. ASK is easy to implement because it requires less complex circuitry. It can also be more bandwidth-efficient than FSK, especially in narrowband systems. When transmitting binary zeros, ASK systems use less power. Besides these advantages, ASK could not be the best option for noise susceptibility, robustness and power efficiency. ASK is more susceptible to noise because amplitude can be easily affected by external factors like interference or signal attenuation. Also, it is less robust in environments with varying signal strength or interference. The power efficiency of ASK is generally lower, especially when transmitting binary ones.

The benefits of FSK are categorized as noise immunity, robustness, and non-linear channels. FSK is less susceptible to noise because signal amplitude variations are less likely to affect frequency changes. It is more resistant to interference and fading in noisy environments, making it suitable for mobile and radio frequency applications. Furthermore, FSK outperforms amplitude distortion in nonlinear channels (such as RF transmission). Aside from these benefits, FSK may not be the best option when it comes to non-complexity, low power consumption, and bandwidth requirements. FSK modulation and demodulation require more complex circuitry. It typically necessitates more bandwidth than ASK. Because the transmitter remains active throughout the signal transmission, it may consume more power.

To summarize, the choice between ASK and FSK is largely determined by the communication system's specific requirements, such as the operational environment, power constraints, bandwidth availability, and the level of robustness required against noise and interference.

Q2. What were the observed differences between ASK and FSK signals in terms of waveform and characteristics?

A2. Amplitude Shift Keying (ASK) and Frequency Shift Keying (FSK) differ in their approach to modulating a carrier signal for digital data transmission. ASK works by varying the amplitude of the carrier wave to correspond with the binary data while maintaining the frequency constant; this produces a waveform that mirrors the shape of the original data signal but is susceptible to noise interference because amplitude is easily distorted. FSK, on the other hand, changes the frequency of the carrier wave to represent the data while keeping the amplitude constant; this results in a more complex waveform that is inherently more resistant to noise, particularly amplitude-based interference. While ASK is simpler and more efficient in terms of bandwidth, it is less noise-resistant than FSK, which, despite requiring more bandwidth and complex circuitry, provides greater reliability in noisy environments.

Q3. How was the ASK signal generated in this experiment?

A3. According to the MATLAB code[1] that is provided on appendix, following a sequence of steps to encode a digital binary message into an B-ASK waveform process can be applied as below:

- **Defining Parameters and Data:** The script starts by specifying the binary data to be transmitted ***('bits')***, the bit duration ***('tb')***, and the sampling frequency ***('fs')***. These parameters lay the groundwork for waveform generation.
- **Reshaping the Data:** The binary data ***('bits')*** are reshaped and replicated to match the sampling frequency. This step ensures that each bit of the binary message has an adequate number of samples per bit ***('spb')*** to accurately depict the signal over time.
- **Modulation:** The modulation step is at the heart of the ASK generation process. A cosine function is used to generate a carrier wave at a specific frequency ***('fc')***. This carrier wave's amplitude is modulated by the binary message, which is now represented as a continuous wave ***('message')***. The carrier wave is multiplied by the message signal to achieve this. The amplitude of the carrier wave is directly proportional to the binary values (0 or 1) in the message, which is an important consideration here.
- **Adding Noise:** The script adds noise to the modulated signal to simulate a real-world scenario. This is accomplished by introducing a Gaussian noise component ***('awgn_bask')*** with a set signal-to-noise ratio (SNR). As a result, the ASK signal is noisy, simulating the conditions of a typical communication channel.

- **Demodulation Process:** Demodulation is performed by correlating the received noisy signal with the original carrier wave and then applying a threshold to detect the binary states. This procedure essentially reverses the modulation to recover the original binary data from the ASK signal.

The characteristics of ASK modulation are visible throughout this process: the amplitude of the carrier wave changes according to the binary data, while the frequency remains constant. As seen in the script, this modulation technique is simple yet effective for digital data transmission, albeit susceptible to amplitude-based noise, as addressed in the noise addition step.

Q4. How was the FSK signal generated in this experiment?

A4. In the provided MATLAB code[1], a Binary Frequency Shift Keying (B-FSK) signal is generated following these steps except defining parameters, reshaping the data and noise addition since these sections are same as the previous B-ASK part:

- **Carrier Generation and Modulation:** Two carrier frequencies ***('f0')*** and ***('f1')*** are defined, which correspond to the binary digits '0' and '1', respectively. Cosine functions are used to generate carrier waves ***('carrier_fsk0')*** and ***('carrier_fsk_1)*** at these frequencies. The signal ***('mod_fsk')*** is produced by selecting the appropriate carrier frequency for each bit in the binary data. If a bit is '0', the segment of the ***('mod_fsk')*** signal is modulated with ***('carrier_fsk_0')***; if it is '1', ***('carrier_fsk_1')*** is used.
- **Demodulation and Detection:** The demodulation process involves cross-correlation of each bit segment of the noisy signal with both carrier waves. The correlation result is compared to a threshold to determine whether the bit is a '0' or a '1'.

This procedure demonstrates the fundamental principles of B-FSK modulation and demodulation, a reliable technique for digital signal transmission that is less susceptible to noise and interference than amplitude-based modulation schemes.

Conclusion

In this lab report on "ASK/FSK/PSK Modulation Techniques," we looked at the practical applications and comparisons of Amplitude Shift Keying (ASK) and Frequency Shift Keying (FSK). The MATLAB experiments revealed that, while ASK is simple and bandwidth-efficient, it is more susceptible to noise, as evidenced by the amplitude variations of the modulated signal. FSK, on the other hand, demonstrated its strength in noise immunity and robustness, making it suitable for difficult communication environments. The successful recovery of original binary data during demodulation under noisy conditions demonstrated the efficacy of both techniques. Finally, this report emphasizes that the choice between ASK and FSK is dependent on the specific needs of a communication system, taking into account factors such as complexity, bandwidth efficiency, and noise resilience.

Appendix

[1] MATLAB Code "LAB5_harun_durmus.m"

```
clc;
clear all;
close all;
%% 5.1      B-ASK Modulation and Demodulation
%% 5.1.a    (defining parameters and the data to be transmitted)
%%
bits = [0 1 0 1 1];
N_bits = length(bits);
tb = 0.02;
fs = 10000;
ts = 1/fs;
time = 0:ts:(N_bits*tb)-ts;
%% 5.1.b    (reshaping)
%%
bits_reshaped = reshape(bits, N_bits, 1);
spb = tb*fs; % sample per bit
message = repmat(bits_reshaped, 1, spb);
message = reshape(message', 1, []);
%% 5.1.c    (modulation)
%%
fc = 2500;
carrier_bask = cos(2*pi*fc*time);
mod_bask = 5 * message .* carrier_bask;
% modulation for bit 0 equals to 0, since it multiplies with 0.
%% 5.1.d    (noise addition)
%%
pavg_bask = sum(abs(mod_bask).^2)/length(mod_bask);
snr_db = 10;
snr_lin = 10^(0.1*snr_db);
var_noise_bask = pavg_bask/snr_lin;
awgn_bask = sqrt(var_noise_bask)*randn(1,length(mod_bask));
noisy_mod_bask = mod_bask + awgn_bask;
%% 5.1.e    (demodulation)
%%
demod_bask = zeros(size(message));
max_lag = 0; % defined maximum lag
for i=1:N_bits
    segment_bask = noisy_mod_bask((i-1)*spb + (1:spb));
    correlation_bask = xcorr(segment_bask, carrier_bask(1:spb), max_lag);
    demod_bask(i) = sum(abs(correlation_bask));
end
threshold_bask = max(demod_bask)/2;
demodulation_bask = zeros(1,N_bits);
for i = 1:N_bits
    if demod_bask(i) > threshold_bask
        demodulation_bask(i) = 1;
    else
        demodulation_bask(i) = 0;
    end
end
demodulation_bask = reshape(demodulation_bask, length(demodulation_bask), 1);
demodulation_bask = repmat(demodulation_bask, 1, spb);
demodulation_bask = reshape(demodulation_bask', 1, []);
%% 5.1.f    (results)
%%
```

```
figure(1)
subplot(411);
plot(time, message,"color","#0072BD","LineWidth",1.5);
title("MESSAGE SIGNAL");
xlabel("message duration (s)");
ylabel("amplitude (V)");
legend("message signal [0 1 0 1 1]");
grid on;
subplot(412);
plot(time, mod_bask,"color","#A2142F");
title("B-ASK MODULATED SIGNAL");
xlabel("message duration (s)");
ylabel("amplitude (V)");
legend("b-ask modulated signal");
grid on;
subplot(413);
plot(time, noisy_mod_bask,"color","#D95319");
title("NOISY B-ASK MODULATED SIGNAL");
xlabel("message duration (s)");
ylabel("amplitude (V)");
legend("noisy b-ask modulated signal");
grid on;
subplot(414);
plot(time, demodulation_bask,"color","#EDB120","LineWidth",1.5);
title("B-ASK DEMODULATED SIGNAL");
xlabel("message duration (s)");
ylabel("amplitude (V)");
legend("b-ask demodulated signal");
grid on;
%% 5.2      B-FSK Modulation and Demodulation
%% 5.2.a    (carrier and modulation)
%%
f0=500;
f1=750;
carrier_fsk_0=cos(2*pi*f0*time);
carrier_fsk_1=cos(2*pi*f1*time);
mod_fsk = zeros(1, length(message));
for i=1:N_bits
    start_index = (i-1)*spb + 1;
    end_index = i*spb;
    if bits(i) == 0
        mod_fsk(start_index:end_index) = cos(2*pi*f0*time(start_index:end_index));
    else
        mod_fsk(start_index:end_index) = cos(2*pi*f1*time(start_index:end_index));
    end
end
%% 5.2.d    (noise addition)
%%
pavg_fsk = sum(abs(mod_fsk).^2)/length(mod_fsk);
var_noise_fsk = pavg_fsk/snr_lin;
awgn_fsk = sqrt(var_noise_fsk)*randn(1,length(mod_fsk));
noisy_mod_fsk = mod_fsk + awgn_fsk;
%% 5.2.e    (demodulation and detection)
%%
demod_fsk = zeros(size(message));
for i = 1:N_bits
    segment_fsk = noisy_mod_fsk((i-1)*spb + (1:spb));
    correlation_fsk_0 = xcorr(segment_fsk, carrier_fsk_0(1:spb), max_lag);
    correlation_fsk_1 = xcorr(segment_fsk, carrier_fsk_1(1:spb), max_lag);
```

```
    demodulation_fsk_0(i) = sum(abs(correlation_fsk_0));
    demodulation_fsk_1(i) = sum(abs(correlation_fsk_1));
end
demodulation_fsk_2 = demodulation_fsk_1 - demodulation_fsk_0;
threshold_fsk = max(demodulation_fsk_2)/2;
demodulation_fsk = zeros(1,N_bits);
for i = 1:N_bits
    if demodulation_fsk_2(i) > threshold_fsk
        demodulation_fsk(i) = 1;
    else
        demodulation_fsk(i) = 0;
    end
end
demodulation_fsk = reshape(demodulation_fsk, length(demodulation_fsk), 1);
demodulation_fsk = repmat(demodulation_fsk, 1, spb);
demodulation_fsk = reshape(demodulation_fsk', 1, []);
%% 5.2.f    (results)
%%
figure(2)
subplot(411);
plot(time, message,"color","#0072BD","LineWidth",1.5);
title("MESSAGE SIGNAL");
xlabel("message duration (s)");
ylabel("amplitude (V)");
legend("message signal [0 1 0 1 1]");
grid on;
subplot(412);
plot(time, mod_fsk,"color","#A2142F");
title("B-FSK MODULATED SIGNAL");
xlabel("message duration (s)");
ylabel("amplitude (V)");
legend("b-fsk modulated signal");
grid on;
subplot(413);
plot(time, noisy_mod_fsk,"color","#D95319");
title("NOISY B-FSK MODULATED SIGNAL");
xlabel("message duration (s)");
ylabel("amplitude (V)");
legend("noisy b-fsk modulated signal");
grid on;
subplot(414);
plot(time, demodulation_fsk,"color","#EDB120","LineWidth",1.5);
title("B-FSK DEMODULATED SIGNAL");
xlabel("message duration (s)");
ylabel("amplitude (V)");
legend("b-fsk demodulated signal");
grid on;
```

References

[1] Fundamentals of Communication Systems, 2nd edition, John G. Proakis, Masoud Salehi, 2005