EXPERIMENT – 7

BINARY FREQUENCY SHIFT KEYING BFSK

March 18 - 2025

B Siddharth Sekhar - EC22B1064

Aim:

To implement Binary Frequency Shift Keying (BFSK) on a specified binary message sequence, decode the signal, and compare it with the original message. Additionally, analyze the effect of noise on the BFSK demodulation process and plot the corresponding waveforms.

Theory:

BINARY FREQUENCY SHIFT KEYING (BFSK):

Binary Frequency Shift Keying (BFSK) is a digital modulation technique where the frequency of the carrier signal changes based on the binary data. The two frequencies represent binary '1' and '0'. The amplitude and phase of the carrier remain constant.

Mathematical Representation

The BFSK modulated signal is given by:

$$s(t) = \begin{cases} A\,cos(2\pi f_1 t) & if\; b(t) = 1 \\ A\,cos(2\pi f_2 t) & if\, b(t) = 0 \end{cases}$$

where:

- A is the carrier amplitude,
- f_1 and f_2 are the two different carrier frequencies,
- $b(t)$ is the binary message signal.

Advantages of BFSK:

- Less susceptible to noise compared to ASK.
- Suitable for non-coherent demodulation.

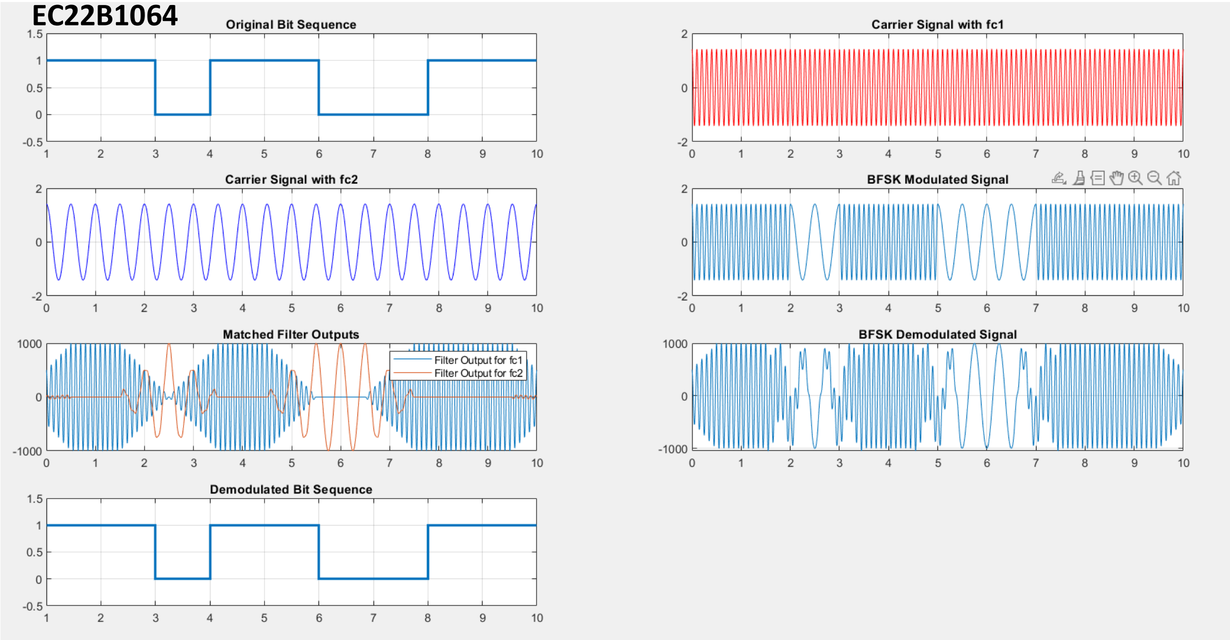
Disadvantages of BFSK:

- Requires a wider bandwidth compared to ASK.
- More complex circuitry compared to BASK.

Q1)Perform Binary Frequency Shift Keying (BFSK) and decode the signal.

```
clc; clear; close all;
fs = 1000;
Tb = 1;
Eb = 1;
fc1 = 10;
fc2 = 2;
N = 10;

bits = randi([0 1], 1, N);

t = 0:1/fs:Tb-1/fs;
t_total = 0:1/fs:N*Tb-1/fs;

A = sqrt(2*Eb/Tb);
carrier1 = A * cos(2*pi*fc1*t);
carrier1_ = A * cos(2*pi*fc1*t_total);
carrier2 = A * cos(2*pi*fc2*t);
carrier2_ = A * cos(2*pi*fc2*t_total);

matched_filter1 = fliplr(carrier1);
matched_filter2 = fliplr(carrier2);


modulated_signal = [];
for i = 1:N
    if bits(i) == 1
        modulated_signal = [modulated_signal, carrier1];
    else
        modulated_signal = [modulated_signal, carrier2];
    end
end

filtered_signal1 = conv(modulated_signal, matched_filter1, 'same');
filtered_signal2 = conv(modulated_signal, matched_filter2, 'same');

received_bits = zeros(1, N);

for i = 1:N
    index_start = (i-1)*length(t) + 1;
    index_end = i*length(t);

    y1 = sum(filtered_signal1(index_start:index_end).^2);
    y2 = sum(filtered_signal2(index_start:index_end).^2);

    if (y1-y2) > 0


        received_bits(i) = 1;
    else
        received_bits(i) = 0;
    end
end

figure;
subplot(4,2,1);
stairs(bits, 'linewidth', 2);
```

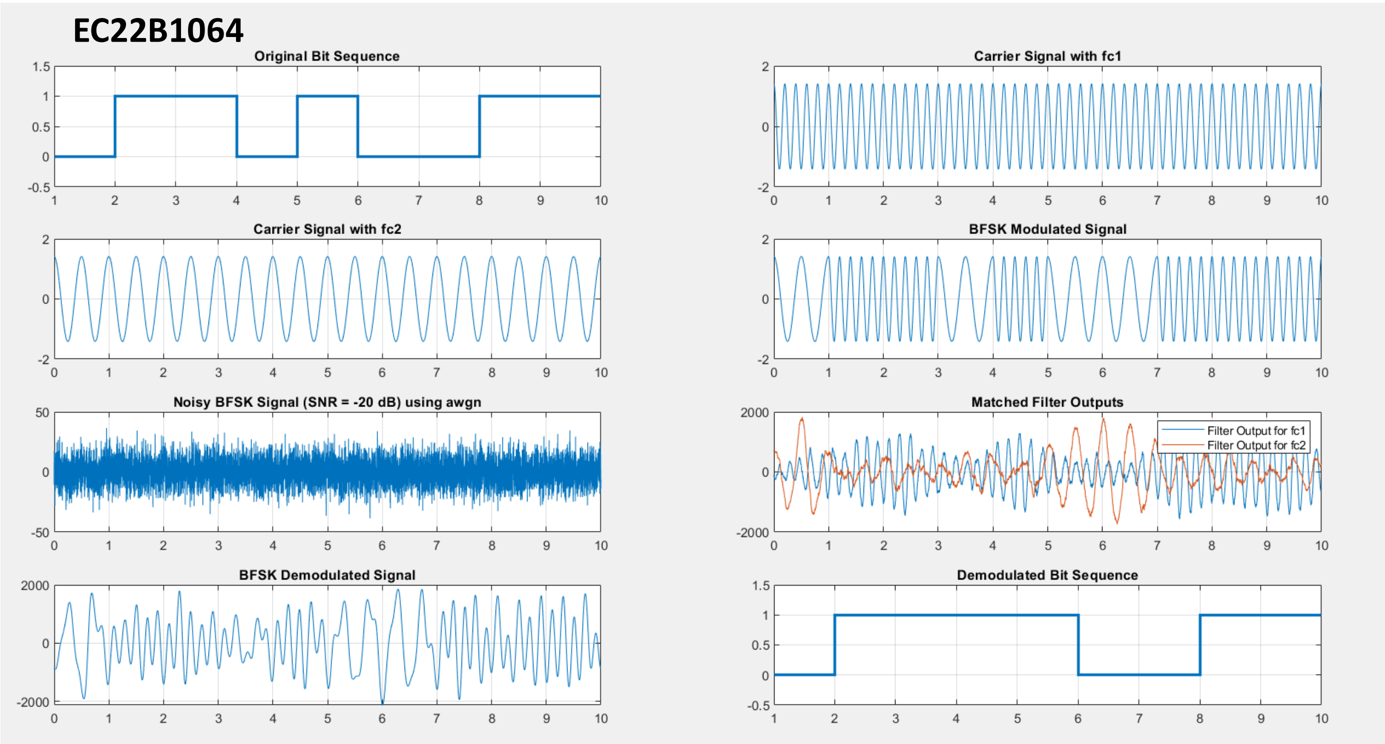
```
ylim([-0.5 1.5]);
title('Original Bit Sequence');
grid on;

subplot(4,2,2);
plot(t_total, carrier1_, 'r');
title('Carrier Signal with fc1');
grid on;

subplot(4,2,3);
plot(t_total, carrier2_,'b');
title('Carrier Signal with fc2');
grid on;

subplot(4,2,4);
plot(t_total, modulated_signal);
title('BFSK Modulated Signal');
grid on;

subplot(4,2,5);
plot(t_total, filtered_signal1); hold on;
plot(t_total, filtered_signal2); hold off;
title('Matched Filter Outputs');
legend('Filter Output for fc1', 'Filter Output for fc2');
grid on;

subplot(4,2,6);
plot(t_total, filtered_signal1 - filtered_signal2);
title('BFSK Demodulated Signal');
grid on;

subplot(4,2,7);
stairs(received_bits, 'linewidth', 2);
ylim([-0.5 1.5]);
title('Demodulated Bit Sequence');
grid on;
```



Q2) Analyze the effect of noise on BFSK demodulation process by adding AWGN to the BFSK Modulated signal

```
clc; clear; close all;

fs = 1000;
Tb = 1;
Eb = 1;
fc1 = 5;
fc2 = 2;
N = 10;
SNR_dB = -20;

bits = randi([0 1], 1, N);

t = 0:1/fs:Tb-1/fs;
t_total = 0:1/fs:N*Tb-1/fs;

A = sqrt(2*Eb/Tb);
carrier1 = A * cos(2*pi*fc1*t);
carrier1_ = A * cos(2*pi*fc1*t_total);
carrier2 = A * cos(2*pi*fc2*t);
carrier2_ = A * cos(2*pi*fc2*t_total);

matched_filter1 = fliplr(carrier1);
matched filter2 = fliplr(carrier2);
modulated_signal = [];
for i = 1:N
    if bits(i) == 1
        modulated_signal = [modulated_signal, carrier1];
    else
        modulated_signal = [modulated_signal, carrier2];
    end
end

received_signal = awgn(modulated_signal, SNR_dB, 'measured');

filtered_signal1 = conv(received_signal, matched_filter1, 'same');
filtered_signal2 = conv(received_signal, matched_filter2, 'same');

received_bits = zeros(1, N);

for i = 1:N
    index_start = (i-1)*length(t) + 1;
    index_end = i*length(t);

    y1 = sum(filtered_signal1(index_start:index_end).^2);
    y2 = sum(filtered_signal2(index_start:index_end).^2);

   decision_signal(i) = y1 - y2;

   if (y1-y2) > 0
       received_bits(i) = 1;
   else
       received_bits(i) = 0;
   end
end


figure;
subplot(4,2,1);
stairs(bits, 'linewidth', 2);
ylim([-0.5 1.5]);
title('Original Bit Sequence');
grid on;
```

```
subplot(4,2,2);
plot(t_total, carrier1_');
title('Carrier Signal with fc1');
grid on;

subplot(4,2,3);
plot(t_total, carrier2_);
title('Carrier Signal with fc2');
grid on;

subplot(4,2,4);
plot(t_total, modulated_signal);
title('BFSK Modulated Signal');
grid on;

subplot(4,2,5);
plot(t_total, received_signal);
title(['Noisy BFSK Signal (SNR = ' num2str(SNR_dB) ' dB) using awgn']);
grid on;

subplot(4,2,6);
plot(t_total, filtered_signal1); hold on;
plot(t_total, filtered_signal2); hold off;
title('Matched Filter Outputs');
legend('Filter Output for fc1', 'Filter Output for fc2');
grid on;

subplot(4,2,7);
plot(t_total, filtered_signal1 - filtered_signal2);
title('BFSK Demodulated Signal');
grid on;

subplot(4,2,8);
stairs(received_bits, 'linewidth', 2);
ylim([-0.5 1.5]);
title('Demodulated Bit Sequence');
grid on;
```



Inference:

- In BFSK, different carrier frequencies represent binary symbols, making it more robust against noise than ASK.
- The presence of noise (AWGN) affects the demodulation process, increasing the probability of bit errors.
- Trade-off: BFSK requires more bandwidth but is more reliable in noisy environments

Conclusion:

- Binary Frequency Shift Keying (BFSK) is an efficient modulation technique used in wireless communication.
- BFSK provides better noise immunity than ASK but at the cost of increased bandwidth.
- The experiment successfully demonstrates BFSK modulation and demodulation, verifying the transmitted and received signals.

References:

[1] Simon Haykins, Communication systems, 2nd ed. (New York John Wiley and Sons, 2005).