Academic Year: 2025 – 2026
Course: Signals and Systems
Department: Computer Engineering
Supervisor: Dr. Moradian

Design and Implementation of a Multi-Channel FDM Communication System

Arman Mesgary

February 4, 2026

Abstract

This report explains how we built a Frequency Division Multiplexing (FDM) system using Python. We recorded audio files, checked how "bit-depth" changes the sound quality (8-bit vs 3-bit), and sent three audio signals on different frequencies (5kHz, 12kHz, 19kHz). We also designed a filter to get one specific audio file back. Finally, we looked at what happens if the sampling rate is too low (Aliasing). The results show that the system works well and the recovered sound is clear.

Keywords: FDM, Modulation, Quantization, Nyquist Rule, Filters, Python.

Contents

1 Introduction

Frequency Division Multiplexing (FDM) is a way to send multiple audio signals at the same time over one channel. To do this, we move each audio signal to a different frequency so they don't mix with each other.

In this project, we did the following steps:

1. **Source:** Recording audio and converting it to digital numbers.
2. **Sender:** Mixing the audio with carrier waves.
3. **Analysis:** Checking if the sampling rate is sufficient.
4. **Receiver:** Filtering and getting the original audio back.

2 Input Generation and Recording

We recorded three audio files using Python. Each file is 10 seconds long with a sampling rate of 44100 Hz:

- **Audio 1:** Counting 1 to 10.
- **Audio 2:** Counting 11 to 20 (Target).
- **Audio 3:** Counting 21 to 30.

We adjusted the volume of all files to be between -1 and 1.

3 Quantization Analysis

Quantization means rounding the exact signal values to specific steps. We compared 8-bit (high quality) with 3-bit (low quality).

3.1 Comparison and Results

Figure 1 shows a small part of the signal (200ms). You can see that the 3-bit signal looks like "stairs" instead of a smooth wave.

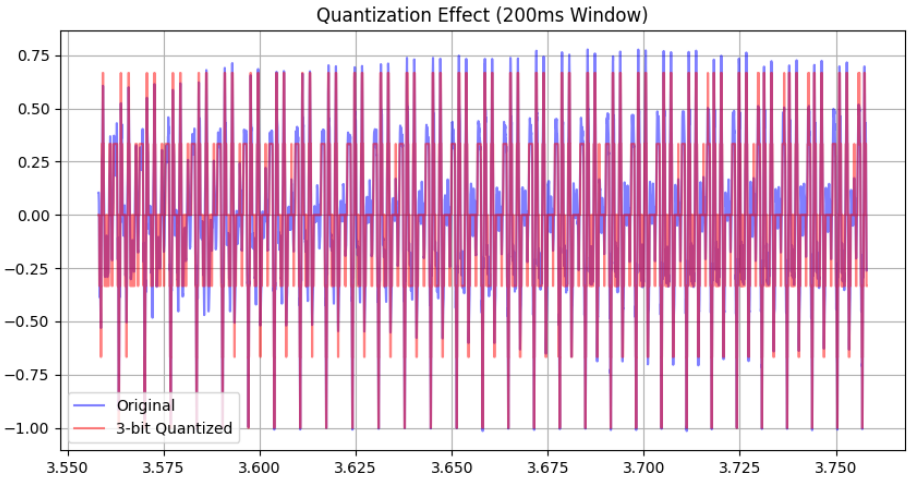

Figure 1: Original Signal vs. 3-bit Quantized Signal (200ms window).

Answer to Project Questions:

Q: What is the difference in sound? Which one has more noise?
A: The 3-bit audio sounds grainy and has a loud background hiss. This is because the "rounding error" is big. So, **noise is dominant in the 3-bit case**. The 8-bit audio sounds very close to the original.

4 Modulation (FDM)

To send the signals, we multiplied each audio file by a cosine wave (Carrier). This moves the sound to a higher frequency.

- Audio 1 $\rightarrow$ 5 kHz
- Audio 2 $\rightarrow$ 12 kHz
- Audio 3 $\rightarrow$ 19 kHz

The formula we used is simply:

$$y(t) = \text{Message}(t) \times \cos(2\pi f_c t) \tag{1}$$

Figure 2 shows the frequencies of the final mixed signal.

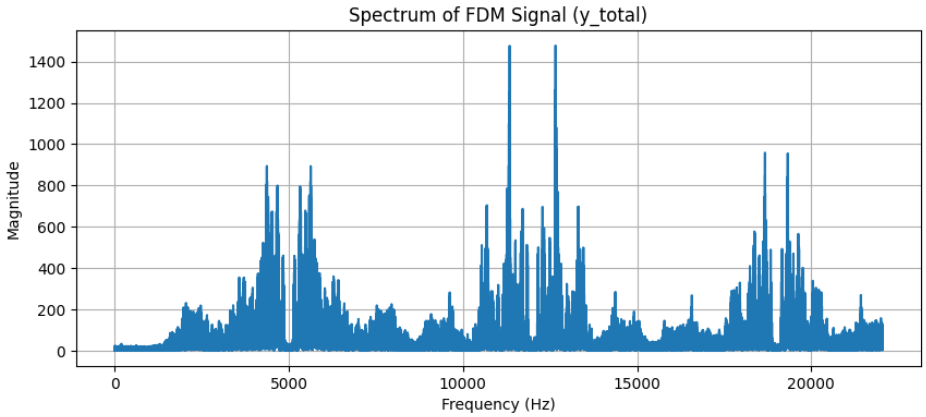

Figure 2: Frequency Spectrum of the mixed signal.

Answer to Project Questions:

Q: Do you see three distinct sections?
A: Yes, we can see three separate spikes at 5kHz, 12kHz, and 19kHz. They do not overlap because we chose frequencies that are far enough apart.

5 Nyquist Rule and Aliasing

The Nyquist rule says we must sample the signal at a **high enough rate** (at least double the highest frequency). Our highest frequency is about 23kHz, so we need at least 46kHz. We tested a lower rate (22050 Hz) to see what happens.

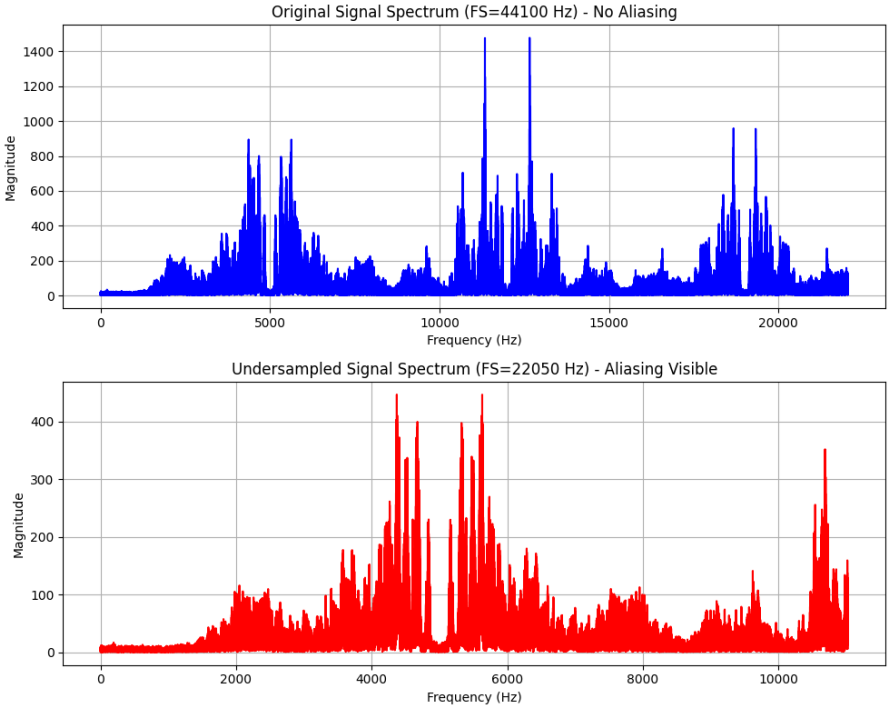

Figure 3: Top: Correct Sampling. Bottom: Insufficient Sampling (Aliasing).

Answer to Project Questions:

Q: When does Aliasing happen? What does it do?
A: Aliasing happens in the second case ($f_s = 22050$ Hz) because the sampling rate is **too low**. As seen in Figure 3 (bottom), the high frequencies (19kHz) crash into the lower frequencies. This destroys the signal and makes it impossible to fix.

6 Receiver Design: Bandpass Filter

We want to get Audio 2 back. We designed a "Bandpass Filter" that only lets frequencies between 9000 Hz and 15000 Hz pass through.

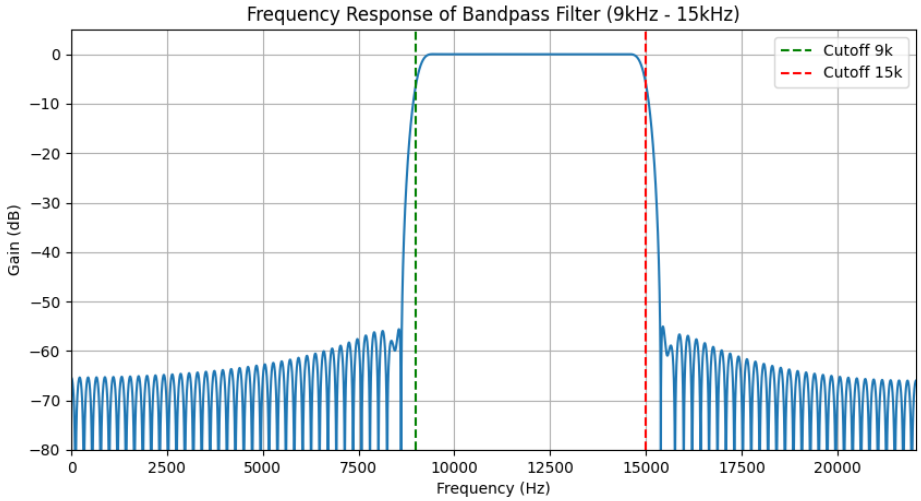

Figure 4: How the Bandpass Filter works.

Answer to Project Questions:

Q: Did the filter remove the other messages?
A: Yes. The graph shows that the filter keeps the 12kHz signal (Audio 2) but blocks the 5kHz and 19kHz signals almost completely.

7 Demodulation and Recovery

To turn the high-frequency signal back into sound, we multiply it by the carrier again and use a Lowpass filter to smooth it out.

Answer to Project Questions:

Q: Why multiply by 2?
A: To answer this, let's look at the math. When we multiply the received signal by the carrier $\cos(2\pi f_c t)$ again, we get:

$$d(t) = m(t)\cos(2\pi f_c t) \times \cos(2\pi f_c t) = m(t)\cos^2(2\pi f_c t) \tag{2}$$

Using the trigonometric identity $\cos^2(A) = \frac{1}{2}[1 + \cos(2A)]$, this becomes:

$$d(t) = \frac{1}{2}m(t) + \frac{1}{2}m(t)\cos(4\pi f_c t) \tag{3}$$

The Lowpass filter removes the high-frequency part ($\cos(4\pi f_c t)$), leaving only $\frac{1}{2}m(t)$. Because the amplitude is now half ($\frac{1}{2}$), we must **multiply by 2** to bring the volume back to its original level ($m(t)$).

7.1 Final Result

Figure 5 compares the recovered audio with the original one. They look almost the same.

Figure 5: Original Audio 2 (Red) vs. Recovered Audio (Blue).

8 Conclusion

In this project, we successfully built an FDM system. We learned that using too few bits (3-bit) makes the sound bad. We also saw that sampling at a rate that is **too low** ruins the signal (Aliasing). Finally, our digital filter worked perfectly to separate the target audio from the mix.