

EE 354 Design Project

Objective

For this project, you will investigate the transmission of voice signals via digital modulation. The primary objective is to transmit a voice signal using the smallest possible bandwidth while maintaining the highest possible quality. The project is composed of three phases:

- **Phase 1:** PCM encoding and decoding of a voice signal using a simulated ADC and DAC.
- **Phase 2:** Evaluation of several different digital communication schemes to investigate the performance of design alternatives.
- **Phase 3:** Design and implementation of your end-to-end digital communication scheme.

Detailed instructions for getting started on Phase 1 and Phase 2 are given below, and the Matlab code you need can be found on the course Blackboard site in the Project folder.

Administrative Info

Two reports and one demonstration are required as part of this project, as detailed below. The due dates are:

Peer Design Review: April 3, 2009
Final Design Report: April 28, 2009
Project Demonstration: April 21, 2009

Reports/Demonstrations are due not later than 1700 on the days indicated. Late work will be penalized at the rate of 10 points for every hour past the deadline.

Project Groups

Students will generally be expected to work in groups of 2 (single-person and three-person groups will be authorized at the instructor's discretion). Midway through the design project, you will pair up with another group and conduct a Peer Design Review of each other's progress to that point (more info to follow). While discussion of theory, techniques, results, etc. among groups and/or with other faculty members is allowed and encouraged, copying another group's work in whole or in part (to include Matlab code, designs and/or design elements, or project write-up) is an honor violation.

Peer Design Review

For the PDR, team up with another group and evaluate each other's progress on the project. At a minimum, the group should have completed Phase 1 and have simulated the 2-ASK and 4-ASK communication schemes in Phase 2. You should be able to evaluate whether the Phase 1 results make sense, verify that the simulated BER curves in Phase 2 match the theoretical curves, and evaluate the group's progress on Phase 3. Go through the group's theoretical analysis and Matlab code and check

for errors and omissions. Your findings should then be summarized in a 1-page report which will be submitted to both the instructor and the other group on the due date specified (details to follow).

Note: The grade you receive on the PDR report factors into **your** design project grade. Your findings have no bearing or impact on the other group's grade. What I am looking for is for you to demonstrate a thorough analysis and evaluation of the other group's design—your objective is to cast a critical eye on their progress to date, catch any errors/omissions, and provide them with some meaningful feedback.

Final Report

Your project report is the means for communicating to me what you know about the principles that you have learned from the project.

Projects will be documented in a formal written report (details to follow). The report is limited to a maximum of 10 pages including figures, graphs, tables, and references. You may consult and papers, texts, or confer with any faculty or student colleagues to complete this assignment. However, all work which is not your own must be properly referenced in standard IEEE format.

Communication skills are important in any career and engineering is no different. Use this project to refine your report writing skills. In my experience engineers (or Naval Officers) who cannot communicate their ideas are less likely to get promoted and are not taken as seriously. Part of our job as professors is to help prepare you for your career both in the Navy and in whatever career path you choose to follow after the Navy.

Demonstration

On April 21, instead of a lab, you will have the entire lab period to demonstrate and present a fully functional design to the instructor. You should be prepared to present the following information to the instructor for validation:

- Choices for sampling rate, quantization, and modulation scheme.
- Block diagram of the communication system with all elements fully documented.
- Verification that your signal meets the occupied bandwidth requirements (to include a plot of both the audio and RF spectrum).
- End-to-end demonstration of Matlab data file in and audio out using both the provided file and an instructor-selected audio file.

The instructor will be happy to provide advice and feedback on preliminary demonstrations any time prior to close of business April 17.

Grading

Peer Design Review Report (15%)

Project Demonstration (15%)

Final Report (60%) – Broken down as follows:

Presentation (20%) – How well was your report organized? How well did you express your ideas? This portion of the grade is based on the quality of the report in terms of communication skills. The report should not be a simple listing of the steps performed, it should be written as a standard technical report. One approach would be to have a major section devoted to the different aspects that were examined. In other words, one section could examine the impact of sampling rate; a second section could examine the impact of quantization, a third section could discuss an evaluation of the modulation schemes, and so forth. Figures must be labeled and referred to in the text.

Technical Content (20%) – Did you understand your results? How well did you tie in theoretical concepts to your results? This portion of the grade is based on the technical accuracy of your report.

Completeness (10%) – Did you include all of the things that were required? This portion of the grade is based on the completeness of your results/report.

Conclusion (10%) – What did you learn? What are your conclusions based on the experiments? You need to succinctly summarize your findings and what they mean.

Project Assignment

With the trend towards digital communications, one of the last vestiges of analog communications—the AM radio band—is still alive and kicking. In this project, you will design a replacement digital communications scheme to operate on the 1450 kHz AM radio channel. You will design this system by first evaluating different sampling/quantizing rates for an “analog” voice signal as well as the performance of various digital communication modulation schemes. You will then design and implement a communication scheme subject to the following two cost constraints:

Data Rate Cost: \$0.01/kbps/minute

Bandwidth Cost: \$0.40/kHz/minute

For example, if your system operates at 64 kbps and requires 100 kHz bandwidth, then the total cost would be:

$$C = (\$0.01 / \text{kbps} / \text{min})(64 \text{ kbps}) + (\$0.04 / \text{kHz} / \text{min})(100 \text{ kHz})$$

$$C = \$4.64 / \text{min}$$

In other words, it would cost the radio station \$4.64 per minute to broadcast in this format.

Phase 1

The goal of Phase 1 is to turn an audio signal (see the “Getting Started with the Audio Signal” section below) into a PCM modulated digital signal. Your objective is to maintain the highest possible signal quality for the lowest possible bit rate. In particular, you should investigate the following design questions:

- What is the impact of the sampling rate on the sound quality?
- At what sampling rate does the voice quality begin to suffer?
- How does this compare to the theoretical minimum sampling rate?
- Where do you think that the voice quality breaks down due to quantization (i.e., how many bits are necessary to properly quantize the signal)?

The end result for Phase 1 should be a PCM encoding scheme with the following parameters:

Low-Pass Filter Cutoff Frequency (pick any reasonable value)

Sampling Frequency (pick one of 2048, 4096, 8192, 16384, or 32768 Hz)

Bits of Quantization (anything from 2 – 16)

Phase 2

The goal of Phase 2 is to evaluate 5 modulation schemes (2-ASK, 4-ASK, BPSK, QPSK, and 2-FSK) for bandwidth efficiency and energy efficiency. Your objective here is to minimize the occupied bandwidth while maintaining an acceptable Bit Error Rate (BER Performance).

To evaluate the performance of the system, you will need to accurately simulate each modulation scheme and compare the results to the theoretical predictions. Simulating the performance of communication system is sometimes an arduous task, but it can basically be broken down into the following steps:

- Generate a random sequence of bits (1's and 0's)
- Input the bits to a modulator
- Add noise to the modulated signal based a specific SNR value
- Demodulate and recover the bitstream
- Determine how many bits were received in error
- Calculate the bit error rate

For this project, the necessary modulators, demodulators, and noise adder functions have been provided (see below for descriptions), and your job is to encapsulate them into a simulation. Most communications schemes are simulated for SNRs that range from 0 – 10 dB.

In order to compare the different modulation techniques, you will need to generate the following plots for each scheme:

- Bit Error Rate curves (theory and simulation)
- Power Spectral Density via the `get_spectrum` function

Additionally, you should create a comparison table that lists the Bandwidth and BER for an SNR of 10 dB for all modulation schemes, so that you can easily compare and contrast the different modulation techniques. Formulas for the theoretical BER curves are given below.

**BER and Operating Bandwidth for Various Digital
Communication Schemes**
(SNR is per bit in linear units)

Modulation	BER	RF Bandwidth
2-ASK	$P_E = Q\left(\sqrt{\frac{snr}{2}}\right)$	$BW = R_b$
4-ASK	$P_E = Q\left(\sqrt{\frac{snr}{4}}\right)$	$BW = \frac{1}{2} R_b$
BPSK	$P_E = Q\left(\sqrt{2 \times snr}\right)$	$BW = R_b$
QPSK	$P_E = Q\left(\sqrt{2 \times snr}\right)$	$BW = \frac{1}{2} R_b$
2-FSK	$P_E = Q\left(\sqrt{snr}\right)$	$BW = \frac{5}{2} R_b$

Phase 3

In Phase 3 you will implement your chosen design by taking the bitstream generated from Phase 1 and using that as the input to your simulation. You should also include a DAC conversion back to an audio signal at the output of your simulation so that the system can be demonstrated to the instructor. You should simulate the performance of your system over a range of 0-10 dB, and include a plot of the BER and power spectrum in the report.

Matlab Functions

The following Matlab functions can be found on the course Blackboard site:

```
DesignProjectAudio.mat
Analog2Digital.m
Digital2Analog.m
bandwidth.m
filter_audio.m
sample.m
uniformquantize.m
AddNoise.m
PhaseMod.m
PhaseDemod.m
AmpMod.m
AmpDemod.m
FSK2Mod.m
FSK2Demod.m
```

`Analog2Digital(x, f_s, bits, f_s_original)` - This function converts an 'analog' (i.e., highly oversampled continuous amplitude signal) to a PCM modulated digital signal (i.e., a vector of ones and zeros). x is the input vector, f_s is the desired sampling rate, $bits$ is the number of bits per sample (i.e., quantization levels = 2^{bits}), and $f_{s_original}$ is the original sampling rate of the 'analog' signal.

`Digital2Analog(x, NumBits)` - This function creates a numerical valued signal from a string of bits assuming that $NumBits$ per quantization level are used. Further, it is assumed that the levels are evenly distributed about zero.

`bandwidth(x, fs)` - This function calculates the bandwidth of a signal that contains 95% of the total energy in the signal.

`filter_audio(x, fco, fs_original)` - Implements a simple Low Pass Filter to prevent aliasing of the audio signal when sampled. Note that fco must be less than half of $fs_original$ and that fco is the -3dB cutoff point.

`sample(x, fs, fs_original)` - This function creates a "sampled" version of the input vector x using a rate of f_s samples per second. In order for this function to operate properly, $fs_original$ must be divisible by fs .

`uniformquantize(x, levels)` - This function creates a vector y which is a quantized version of the input x . Note that levels will be equal to 2^N , where N is the number of bits of quantization. The input is assumed to be normalized to be between -1 and +1.

`AddNoise(x, SNRperBit, BitsPerSymbol)` - This function adds noise to the vector x . The output SNR per bit of this vector is determined by the input variable $SNRperBit$. Note that $SNRperBit$ is *linear* not in dB.

`PhaseMod(x, k)` – This function uses an input vector of data bits (x) of length N to create an output vector of symbols of length N/k where $k=\log_2(M)$ is the number of bits per symbol and M is the number of possible symbols. The symbols are complex baseband representations of M -ary phase modulated signals.

`PhaseDemod(x, k)` – This function creates a vector of bits of length $N*k$ (where $k=\log_2(M)$ is the number of bits per symbol) from a length N vector of phase modulated symbols. The assumed modulation is M -PSK where $M=2^k$.

`AmpMod(x, k)` – This function uses an input vector of data bits (x) of length N to create an output vector of symbols of length N/k where $k=\log_2(M)$ is the number of bits per symbol and M is the number of possible symbols. The symbols are complex baseband representations of M -ary amplitude shift keying modulated signals.

`AmpDemod(x, k)` – This function creates a vector of bits of length $N*k$ (where $k=\log_2(M)$ is the number of bits per symbol) from a length N vector of ASK modulated symbols.

`FSK2Mod(x)` – This function uses an input vector of data bits (x) of length N to create an output vector of symbols of length N . The symbols are complex baseband representations of Binary Frequency Shift Keying modulated signals.

`FSK2Demod(x, k)` – This function creates a vector of bits of length N from a length N vector of 2-FSK modulated symbols.

Getting Started with the Audio Signal

First, go to the course page on Blackboard, navigate to the “Projects” folder, and download the project pack. Unzip the contents of the project file to a directory on your PC.

The .mat file `DesignProjectAudio.mat` contains an “analog” (i.e., highly oversampled with $f_s = 65,536$ Hz) audio recording of voice speech. For the first part of the project, you will need to determine the best combination of sampling and quantization to (a) minimize data rate of the PCM version of a voice signal while (b) obtaining acceptable voice quality.

To evaluate the audio signal, change the Matlab directory to the directory where you have saved the Matlab files, and run the following commands:

```
load DesignProjectAudio.mat
```

We will be using a highly over-sampled ($f_s = 65,536$) version of the voice signal with a very high number of quantization levels to approximate the analog signal. The following Matlab commands can be used to verify that you have loaded the signal properly.

```
sound(Original, 65536); (plays the sound file)  
[freq,power] = get_spectrum(Original,65536); (get the spectral data)  
plot(freq,10*log10(power), 'k-'); (plots the PSD with dB units, be sure to zoom in)  
plot(time, Original); (plots the time waveform)  
bandwidth(Original, 65536); (Calculates the Bandwidth of the signal – should get 1.4 kHz)
```

To evaluate the impact of sampling and quantizing the original signal, you will need to use the `sample.m` and `uniformquantize.m` files (see the help information associated with each to determine how to utilize them). You should listen to the signal for various sampling frequencies and quantization levels. Note that the voice should play at normal speed (i.e., it shouldn't sound as if it were going too fast or too slow). If it isn't, you may not be using the function `sound()` properly.

To convert the audio signal into a series of bits that you can use with your communication system, use the `Analog2Digital.m` and `Digital2Analog.m` files.

NOTE: In order for the `Analog2Digital` function to operate properly, you must select a sampling frequency which is a power of 2 (1024, 2048, 4096, etc.)

Code to verify the cutoff frequency of your filter

In order to verify the performance of your filter, you need only compute the impulse response, and then find the Fourier Transform of that impulse response. The following code snippet should help you evaluate your filter design.

```
fs_original = 65536;      % Original Sampling Frequency of 65,536 Hz
fco = 1000;              % Desired Filter Cutoff Frequency of 1 kHz

% The following code creates our delta-function in Matlab

test = zeros(1,100000); % Create a long vector of zeros
test(round(length(test)/2)) = 1; % Set a single, solitary value to be 1

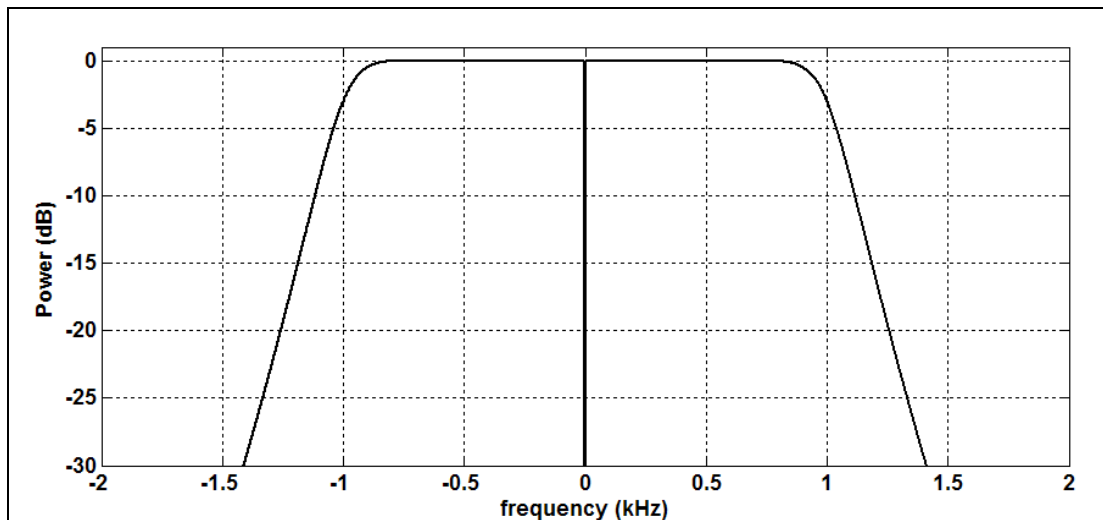
% Filter the signal

filtered_audio = filter_audio(test, fco, fs_original);

% Plot the result

[freq,power]=get_spectrum(filtered_audio,fs_original);
plot(freq/1e3,10*log10(power),'k-') % Plot frequency in kHz, Power in dB
xlabel('frequency (kHz)')
ylabel('Power (dB)')
```

You should see the following plot (after zooming in to the region you're interested in):



A Few Notes on Simulation

Simulations **must** (but all too often are not) be supported by analysis. If this is not done, the simulation results are not reliable since they cannot be properly verified. The results are therefore **useless** even if correct.

An important aspect of running simulations is to **sanity-check** your results. Before running a full end-to-end simulation, verify that each of the pieces are operating correctly. If you input a test signal to the ADC code, does it give you the results you expect? When you calculate BER, are you counting the number of errors properly? If your error rates are too high or too low, can you verify whether you're adding the correct amount of noise to the system (i.e., do you have the SNR set properly)? Before running the entire BER curve, run the code for a single SNR value and verify that the simulated BER matches the calculated BER (doing this can save you a significant amount of wasted time).

Simulating random numbers: In Matlab, you can use the `rand(1,N)` command to give you a vector of N random numbers that fall within the range of [0, 1]. To generate a random vector of bits, simply add the `round()` command: `round(rand(1,N))`.