ECE 380

Mini-Project: Frequency-Shift Keying

Introduction

Perhaps an apt title for this quick introduction to frequency-shift keying (FSK) modems would be "Everything you wanted to know about FSK modems but were afraid to ask!"

The goal of this mini-project is to help you understand a simple modem, the FSK modem, referred to by the International Telecommunications Union (ITU) as V.21. The V.21 modem communicates 1's and 0's by sending either a 1650 Hz tone or an 1850 Hz tone, respectively, for 1/300 second. Thus the overall data rate is 300 bits/second. Even though 300 bps is quite slow compared with the theoretical maximum of 56 kilobits/s over a phone line, the V.21 format is still used in almost every modem call. This is because receiving and decoding this format is so simple. A V.21 modem call can be received without using difficult techniques such as equalizers, echo cancellers, and matched filters. Furthermore, it can be received accurately even in the presence of a significant amount of noise. For these reasons, V.21 is used as an initial *handshake* between two modems. You can hear the V.21 modem tones at home when your V.34, V.90, or V.92 phone line modem or fax machine starts a phone call. V.21 is also used to transmit caller ID information over the phone line.

Preliminaries

Continuous Phase FSK

The FSK signal is the concatenation of sinusoidal tones at 1650 Hz or 1850 Hz. There is no silence between the tones. Simply concatenating bursts of sinusoid can create a waveform with discontinuities when the frequency shifts. The concatenated signal will look smoother if the phase of each successive burst is adjusted to avoid these discontinuities. In LabVIEW, the DSP First Function Generator has a "reset signal" input. When properly set, the function generator will remember the phase from one sinusoidal burst to the next, and will produce a smooth waveform.

Converting from ASCII to FSK

An FSK modem is a digital communication system, so it transmits binary information. In order to transmit messages expressed in an alphabet, there must be a binary representation for each character in the alphabet. The standard for this representation is ASCII (American Standard Code for Information Interchange), where eight bits are used to represent each character, number, or punctuation mark. For example, the upper-case 'A' is represented in ASCII with the number 65 (decimal), which has an 8-bit binary form of 01000001; lower-case 'b' is 98 in ASCII, or 01100010. Therefore, if we want to encode a message, such as 'Hello World' for the FSK modem, we must turn the eleven characters of the message into zeros and ones. Since spaces are counted as characters (00100000), we would end up with 88 bits. Once we have a vector of bits, we can generate the appropriate sinusoids by agreeing to the convention that 1650 Hz is used to represent a "1" and 1850 Hz is used for "0."

LabVIEW has some useful VIs for converting a string of characters into an array of binary data. The String to Byte Array VI will convert a string into an array of bytes, where each array element contains the numerical value of the ASCII representation of one of the string characters. The Number to Boolean Array VI will convert a number to an eight-bit array of TRUE/FALSE values. You will have to experiment to find out which end of the boolean array contains the most significant bit. Your FSK modem should transmit characters in the order in which they appear in the English text. The bits that make up each character should be transmitted *least* significant bit first.

Synch Bits and Prefix

In order to send a message, the modem needs additional bits to indicate the beginning and end of the message, and also some bits to initialize the modem so that the bits are synchronized. This is accomplished by sending a preordained pattern of bits. For the implementation in this miniproject, the preamble sent prior to the message bits will be a sequence of six "01" pairs followed by eight ones.

0101010101011111111

The alternating zeros and ones are used by the receiver for bit synchronization, but we will postpone a discussion of that topic until the demodulator and decoder have to be designed. The choice of eight ones comes from the fact that 255 in ASCII is not a normal character, so the pattern will not appear as a part of any text message. For the end of the message, another sequence of ones is used to indicate the end. For this miniproject, the message will be terminated with a sequence of 16 ones. All of the "message" bits are to be placed between the preamble and the termination.

Filter Design

When you implement the FSK decoder, it will be necessary to have filters to detect the presence of the 1650 Hz and 1850 Hz tones. A very useful GUI for designing filters is available on the DSPFirst website (www.rose-hulman.edu/DSPFirst). Click on Demos, and under LabVIEW look for Digital Filter Design Utility. In the utility you can enter your filter specifications in the Main Setting and Filter Specification blocks. The frequency response and pole-zero diagrams appear immediately. Once you have a satisfactory design, click the Save Filter Coef button to save the filter coefficients to a text file.

The filter coefficients are saved in a "second-order cascade" form, that gives better numerical performance than the direct form. Fortunately LabVIEW understands this form. In LabVIEW, find the IIR Cascade Filter VI. One of the inputs is called "IIR Filter Cluster." If you "create constant," LabVIEW will create an empty cluster containing two arrays. Open up the text file containing your filter coefficients, and copy and paste the coefficients into the arrays. The reverse-coefficient array ("a" array) is the one on top. If you prefer an FIR filter, use the FIR Filter VI. Follow the same procedure, but there will be only one (long) array.

The Project

FSK Encoder

In LabVIEW create a complete FSK encoding system that will take a text message and produce the correct sequence of sinusoidal signals as would be generated by a V.21 modem. The

message has to be converted from a text string to a stream of bits. In addition, the standard preamble bits must be placed at the beginning of the stream and sixteen ones must be placed at the end to indicate termination of the message. The modem output signal must be continuous in phase. Use a sampling rate of 8000 samples/second (a telephone industry standard). Allow the bit rate to be adjustable, with a front panel control.

Encode the message "Don't give up the ship!" at 50 bps, and then show part of the spectrogram to illustrate that the encoder is functioning properly.

Filter Design

Design a bandpass filter that will pass the 1850 Hz FSK tone and reject the 1650 Hz tone. The passband should extend from 1750 to 1950 Hz, with no more than 1 dB of ripple. The lower stopband should extend from 0 to 1650 Hz and the upper stopband should extend from a frequency of your choice greater than 1950 Hz to 4000 Hz. The gain in the stopband must be no greater than -40 dB. You may use any filter type that the filter design utility supports. Show a plot of the frequency response magnitude (generated in LabVIEW, not in the GUI) in the region 1400 Hz to 2600 Hz to confirm that the filter meets the specs.

Generate an FSK signal with a bit rate of 10 bps, and then process a one-second interval of the FSK signal to see if the filter works correctly. On a plot of the time signal show how the amplitude of the output changes for different bits.

Now change the bit rate to 300 bps and process a short section of the signal containing between 10 and 15 bits. Comment on how well or poorly the bandpass filter works at this bit rate. For example, on a plot of the processed output indicate where the 1850 Hz sections should be found. Comment on whether your filter would be suitable for use in an FSK decoder.

What's Due

For this mini-project you are to work on your own. You may discuss your ideas with others, but you may not share LabVIEW files. The policy stated on the Homework Procedures handout under "Responsibility" applies to this and other mini-projects.

What is due:

1. A brief memo describing what you did. Be sure to have a sentence or two intro and conclusion. Include all of the results required under The Project above.
2. Print out your LabVIEW code. Selecting File→Print will start a print wizard. When you reach the "Print Contents" window, select "Icon, description, panel, and diagram." This will print out the whole works in a compact format.

A hard copy of the memo and LabVIEW printout are to be handed in at the start of class on **Tuesday, February 6**.