|  |
| --- |
|  |
| Lab 5: Design Project |
| Acoustic Data Modem |
|  |
| **Greg Malysa (gjm76) and Arseney Romanenko (asr96)** |
| **5/5/2010** |

# Introduction

For our final design project, we designed and built a prototype acoustic modem to serve as a physical transport layer for digital communications. It converts between a digital communications scheme (RS-232) and an acoustically coupled communications scheme of our own design. Our project consists of a pair of such modems to operate as transmit/receive pairs and supports duplex communications. Although our modem operates in air, it is a proof-of-concept experiment in encoding, decoding, and data transmission techniques that will be used in the following years by the CUAUV team to design a system capable of communicating over some distance underwater.

# High Level Design

### Rationale

We chose this product because of its relevance to a student project team we are both members of, CUAUV (Cornell University Autonomous Underwater Vehicle team). Usually, remote-controlled vehicles use radio frequency (RF) wireless communications to transmit data between the device and its operator. However, due to the nature of water, electromagnetic waves do not propagate well, with an effective range of about a foot, depending on frequency. This makes them unsuitable for communicating with AUVs. Acoustic data, however, can propagate very far underwater and acoustic underwater communications is currently an important area of research at Woods Hole Oceanographic Institute (WHOI). Inspired by their success, our team has as a long term goal the creation of an acoustic modem that can be used to communicate with the vehicle while it is in the water without a tether. Our project serves as a prototype to help us develop our own algorithms and techniques for encoding and processing data acoustically.

### Background Math

Because we used cheap audio-range speakers and microphones, our project was limiting to transmitting within the range of human hearing, ~20 Hz – 20 kHz. Therefore, we selected an encoding scheme that minimized per-byte bandwidth utilization while retaining simplicity, so that it was possible to implement decoding. The two most intuitive and basic digital encoding schemes are frequency-shift keying, where data is transmitted on a pair of frequencies, each representing a distinct digital value, and on-off keying, where the presence or absence of a single frequency is used to encode a digital value. We chose the second (OOK) because it requires half the acoustic bandwidth (uses only one frequency per channel, rather than two) to transmit a particular amount of data. We also chose to use an asynchronous design in order to avoid paying for the overhead of having an additional clock frequency, instead breaking the transmissions into a series of known-width pulses. Transmitted data is broken into chunks of 64 samples so as to conveniently fit into a power of two size buffer so that it can efficiently be implemented as a circular buffer. Each physical frame consists of a start bit (S1, a one bit), eight data bits, most-significant-bit first, and a stop bit (S2, a zero bit). At a sampling rate of 40 kHz (chosen to avoid aliasing across the entire 20 Hz-20 kHz range), 64 samples corresponds to a pulse width of 1.6 ms, which limits each frequency channel to transmitting 62.5 complete frames per second as a theoretical max.

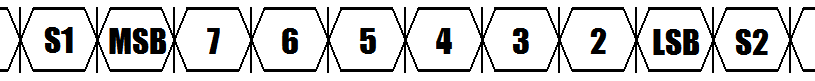


Figure - Frame Decomposition

Next, as we are limited by the computational power of our Atemga644 microcontroller, and we would ideally transmit and receive from a single microcontroller, we designed a hybrid algorithm for detecting transmitted data using FIR filters to estimate the magnitude of specific frequency components over time. For each frequency of interest (which will be derived afterwards), we resample at that frequency, exploiting aliasing to obtain a DC component corresponding to the amplitude of the frequency. Because the cosine of zero is one, the magnitude of the DC component can be estimated with a simple sum of those samples. However, because the real frequency component may have a non-zero phase shift associated with it, at DC it will have an analogous “phase shift” which requires that a second resampler to be used with a 90-degree phase delay—this pair together can estimate the Fourier Transform evaluated at DC for phase and magnitude information.

Figure - Flowchart for Magnitude Estimation

**Resampler**

**40 kHz A/D**

**90 Degree Delay**

**Resampler**

**Moving Average**

**Moving Average**

**Magnitude Estimate**

However, it is important to consider the effects of aliasing on frequencies OTHER than the frequency of interest. Usually, aliasing is a bad property to have in a digital system, as it can cause other frequencies to appear as the target frequency. For instance, when sampling at 6 kHz, the possible aliases that are within our analog pass-band and appear as a DC signal are true DC, 6 kHz, 12 kHz, and 18 kHz. Therefore, in selecting the frequencies for our transmission channels, if we choose 6 kHz as a transmit frequency, we cannot also use 12 or 18 kHz. Because the Fourier Transform estimator is implemented as a moving average, it can also be interpreted as an FIR filter, which has a known frequency response. An N-term moving average filter has a frequency response characterized by a maximum at DC and zeros evenly spaced around the unit circle. The number of terms in the moving average filter for each frequency is chosen by having the length of the filter match the length of each burst in time (1.6 ms). Because it is unlikely for these to match precisely, the filters are usually defined to extend one sample longer than the burst. In general, therefore, with sampling frequency fs and target frequency f, the length of the appropriate averaging filter is . For example, for 6 kHz, N is 10. Because the window is so short, it is important to minimize the interference between frequency channels, which is most easily done by having each channel fall match with a zero in the moving average for all of the other channels. Due to aliasing between the channels, a particular channel frequency may also fall on the alias of a zero rather than the normal interpretation of the zero.

Figure - Magnitude Estimation Flowchart

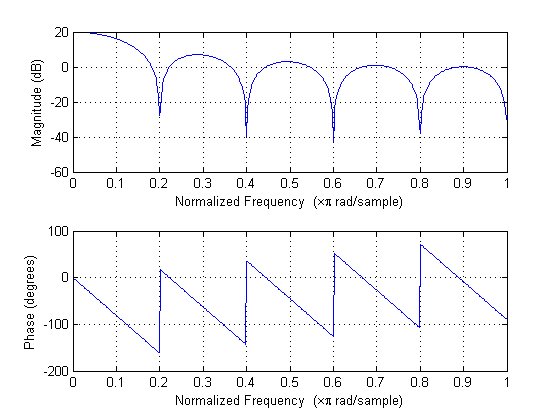


Figure - Frequency Response for a 10-Term Averaging Filter

Fortunately, it is possible to satisfy the requirement that each transmit frequency alias to a zero of all other transmit frequencies for at least four frequencies in the range 20 Hz – 20 kHz. Table 1 lists the relevant zeros for the frequencies we have chosen, with the ones in use highlighted:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Zeros for All Four Transmit Channels | | | | | |  | |  | |  | |  | |  |
| 4800 Hz | 600 | 1200 | 1800 | 2400 |  | |  | |  | |  | |  | |
| 6000 Hz | 600 | 1200 | 1800 | 2400 | 3000 | |  | |  | |  | |  | |
| 8400 Hz | 600 | 1200 | 1800 | 2400 | 3000 | | 3600 | | 4200 | |  | |  | |
| 10800 Hz | 600 | 1200 | 1800 | 2400 | 3000 | | 3600 | | 4200 | | 4800 | | 5400 | |