Josko A. Catipovic
Arthur B. Baggeroer
Keith von der Heydt
Donald E. Koelsch

The Design and Performance Analysis of a Digital Acoustic Underwater Telemetry System

MIT Sea Grant
College Program

Massachusetts
Institute of Technology
Cambridge, MA 02139

MITSG No. 85-12
November 1985

THE DESIGN AND PERFORMANCE ANALYSIS OF A DIGITAL ACOUSTIC UNDERWATER TELEMETRY SYSTEM

Josko A. Catipovic
Arthur B. Baggeroer
Keith von der Heydt
Donald E. Koelsch

MIT Sea Grant
College Program

Massachussets Institute
of Technology
77 Massachusetts Ave.
Cambridge, MA 02139

Index No NA81AA-D-00069
Project No. R/V-5

MITSG No. 85-12

RELATED PUBLICATIONS

William H. Hanot. A Phased Array Sonar for an Underwater Acoustic Communcation System. MIT Naval Architecture/Ocean Engineering Dept. BS/MS thesis. August 1980. 153 pp. photocopy only available: $4.00

MIT Marine Industry Collegium. A New Underwater Communication System: Oportunity Brief #19. MITSG 80-6 NTIS: PB82-108-128. 1980. 17 pp. $3.50

MIT Marine Industry Collegium. Application of Marine Acoustic System: Opportunity Brief #33. MITSG 83-10. 1983. 52 pp. $4.00

J. Catipovic, A. Baggeroer, K. von der Heydt, and D. Koelsch. Design and Performance analysis of a Digital Acoustic Telemetry System for the Short Range Underwater Channel, IEEE Journal on Oceanic Engineering, vol. OE-9, No. 4, pp 242 - 252, October, 1984.

Copies of these reports may be ordered through:

Sea Grant Information Center
MIT E38-302
Massachusetts Institute of Technology
77 Massachusetts Ave.
Cambridge, MA 02139

617/253-7041

1. ACOUSTIC TELEMETRY IN THE OCEAN

Underwater data transmission is an important aspect of the technology being developed for more efficient ocean utilization. The underwater environment, particularly at significant depth, is hostile to man, and one has to use remote sensors and robots for exploring and utilizing it. Communicating with such devices is difficult, as electromagnetic radiation is severely attenuated in water. The choice for the system designer is between an acoustic data link and an electrical cable tether between the surface station and the underwater instrument. While the latter choice was traditionally favored for high-speed, short range applications, recent advances in electronic technology are making acoustic communication systems more suitable for such applications [1], [2], [3]. This work discusses the design and performance characteristics of a Digital Acoustic Telemetry System (DATS) which incorporates the current state-of-the-art technology and is capable of reliable data transmission at rates useful to a wide range of ocean exploration and development gear.

1.1 Demands For Ocean Communication Devices

The performance requirements placed upon underwater communication link such as the DATS may be grouped into three categories, each with a unique set of performance specifications:

1) Low-rate command and control of a remote industrial or military device requires a relatively modest data rate, with 10 to 100 bits/second usually considered sufficiently fast. The reliability of the data is often required to be extremely high, as drastic actions may result from relatively short command strings. Furthermore, such systems are often deployed for long durations, and the transmission path may be quite long or unknown. These constraints have often led to an acoustic low-frequency, high-redundancy digital data link incorporating either automatic repeat request (ARQ) or highly redundant forward error correction coding techniques.[4 - 8]

2) The technology developed for the exploration of the oceans and the ocean bottom shows a need for data transmission at moderate rates . Many of the presently available oceanographic sensors collect data while deployed at remote locations. Presently, extracting the data from the instrument means retrieving the entire package and reading the stored contents. A data transmitter operating at 100 bits/sec to several kilobits/second would allow accessing the data without disturbing the operation of the instrument. This application requires a higher data rate since an instrument may store many megabytes of

digitized data, and it is desirable to maintain the total transmission time at several hours or less. A data rate of 100 bits/second to several kilobits/second is considered adequate. This application does not require an extraordinarily low error probability; an occasional transmission error may be a small degradation compared to the Signal to Noise Ratio (SNR) and other performance limitations of the data-collecting instrument. A bit error probability range of 0.1% to 0.0001% may be considered adequate for many oceanographic applications.

3) Transmission of real-time video data comprises the third category. Such data is often transmitted using an analog carrier, but the limitations imposed by the ocean channel make transmission of digital data attractive. If a picture is digitized, a medium-quality black and white image requires approximately a megabit even if a moderately sophisticated image compression algorithm is used. Thus reproduction of a real-time TV quality image at the TV rate of 30 frames/second requires a rate of several megabits/second. This rate is presently unattainable by a realizable system; one can only expect to transmit frozen frames and a telemetry system operating at 10 kilobits/second could potentially transmit several video pictures per minute. Recent work on image transmission on 56 kilobits/second digital telephone lines indicates that continuous transmission may be achieved at this rate, but with extensive coding and data compression methods. The resulting algorithms are highly error-sensitive, but are expected to work over the commercially available digital line. [9]

Present capabilities of acoustic video transmissions are limited to "still" images which are adequate for many search and inspection applications. For an uncompressed image transmission, an error rate of as high as 1% might yield an acceptable quality, although image compression algorithms severely degrade the error rate for adequate performance.

It is worthwhile to note that all three above applications, and in fact most remotely operated ocean instruments, operate under tight power constraints with the amount of energy available to the communication system often severely limited. For example, if the device is a free-swimming unmanned submersible with a typical propulsor power requirement of 1 kw, the communication system might be supplied with 100 watts or so for intermittent operation, but an anchored oceanographic buoy might allow only 5 watts. It is becoming popular to use increasingly sophisticated data coding techniques, developed for other environments, such as space communication, to overcome the power limitations.[10,11]

The main limit to reliable underwater transmission is the need to use sound waves for information carriers. Acoustic propagation is currently the only viable way of transmission, but even acoustic waveforms are severely attenuated in water. The energy absorption is roughly proportional to the carrier

frequency, and this places an upper limit on the frequencies usable for a given transmission distance. For the ranges of interest to the DATS, typically less than 2 kilometers, a realistic high frequency limit is 100 kHz. The system bandwidth is further constrained by the coloring of the ambient water noise in the 0 - 100 kHz band, and the difficulty of manufacturing broadband underwater transducers. The DATS transmitter elements were designed for a broadband source, and yet have a Q of approximately 5 in order to maintain a 50% efficiency. Thus significant bandwidth is much more difficult to obtain under water than with other types of carriers.

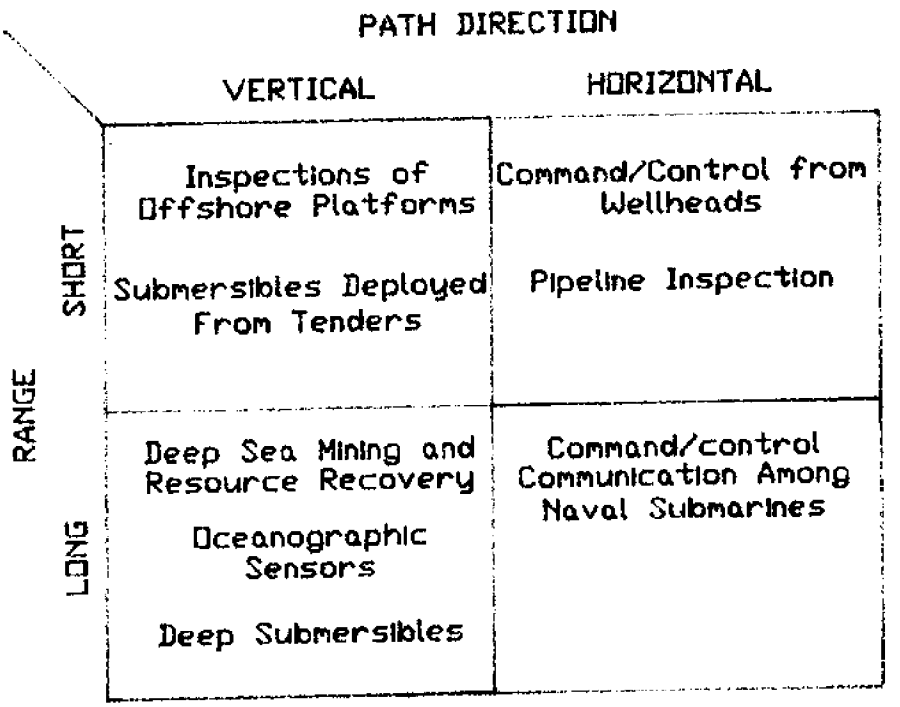


Figure 1: Acoustic Transmission Modes

Before discussing the design of a system to satisfy the above needs, a short discussion of the ocean as a communication channel is in order. Geometrically, acoustic communication can be broken down into four separate pathways through the ocean. It can be long range or short range, and may be propagated either horizontally or vertically. Figure 1 outlines the major uses of

each mode of data transmission.

The deep water vertical transmission path is noise-limited, as acoustic baffles may be used to block out the surface and bottom multipath, and data rates as high as 4800 bits/second have been achieved over this channel [12]. The other three channels are primarily reverberation limited, and the DATS is designed to operate in very reverberant channels. It adaptively monitors the time and frequency dispersion of the channel and uses the measurements to correct the demodulator and decoder. This yields a more complicated, but significantly more robust system which can operate in more places and with less attendant care than systems unable to cope with channel dispersion.

Regardless of the environment, transmissions from underwater sensors or vehicles necessarily involve a wide range of communication channels, and a system must be able to adapt to a wide range of conditions. For example, a "line of sight" is not always present when telemetering from an underwater vehicle. It may go behind a dock, down a crevasse, or otherwise disappear from view, and its data link must be able to cope with such interruptions. The authors feel that a more robust and adaptive system is preferable to a high speed one capable of operating only under contrived conditions.

The design of the DATS is based on a realistic model of the high-frequency acoustic channel generally accepted in the ocean acoustic literature [13 - 15, 17]. The model describes the received sound field as a sum of reflections from various scatterers, and the "line-of-sight" propagation path is not necessarily present.

We present measurements of the underwater acoustic channel used at at frequencies centered at 50 kHz and at ranges of up to 1 km in shallow water. An attempt is made to measure parameters relevant to digital data communication and classify the channel based on these measurements.

2 MODELLING THE TRANSMISSION CHANNEL

2.1 Analytic Model

Consider a communications channel where the received signal is a sum of many scattered replicas of the transmitted waveform. Then if

$$s(t) = e^{jw_1 t} \tag{1}$$

is transmitted, the received signal may be modelled as [17]

$$r(t) = \sum_{i=1}^{N} A_i e^{j[w_1 t + \theta_1]} = be^{j[w_1 t + \theta]} \tag{2}$$

$$be^{j\theta} = \sum_{i=1}^{N} A_i e^{i\theta i} \tag{3}$$

If several scatterers are substantially stronger than the rest and their amplitude and phase are at least partially known, it is customary to treat their contribution to the received field as a specular component and separate it in the analysis from the field due to the rest of the scatterers. A common example of this case occurs when the direct, i.e. non-scattered path constitutes a significant part of the received field. Alternately, one may consider several dominant scatterers as making up the specular path.

The received signal may be divided into the specular and random component:

$$be^{i\theta} = a_1 e^{i\phi_1} + a_2 e^{j\phi_2} \tag{4}$$

where the a_1 and ϕ_1 are deterministic (although possibly unknown) and a_2 is random, distributed as $N(0, v_2)$ and ϕ_2 is random and uniformly distributed. Then the density of b is Rician [13] (assuming ϕ_1 unknown and uniformly distributed).

$$P_b(b) = \frac{b}{v_2^2} \exp\left[\frac{a_1^2 + b^2}{2v_2}\right] I_0\left[\frac{a_1 b}{v_2^2}\right] \tag{5}$$

The channel behavior of interest in the design of DATS is the random modulation of a tone. A commonly used model for the fading behavior of a CW carrier uses the Rician probability density function to describe the instantaneous carrier amplitude. Since the quantity of interest to the DATS decoder is the square of a narrowband tone envelope, this work uses instead the noncentral chi-square density to model the behavior of the square of the tone envelope. The DATS is an MFSK system, and the soft decoding algorithm sums the squares of M received signal envelopes. Such a sum may be modelled by a noncentral chi-square density with 2M degrees of freedom.

Now we consider the processing of the received signal. The optimal receiver for a channel with a specular propagation component implements the following equation.

$$\ell = \frac{v_2^2}{N_0} \left| b \right|^2 + a_2 \mathrm{Re}\left[b e^{-i\theta} \right] \tag{6}$$

The DATS receiver does not track the coherent signal component for ease of implementation and because low specular signal levels were expected. As shown below, the specular signal level was found to be low for all but the very short ranges. The DATS demodulator thus performs the following operation, illustrated in Figure 2.

$$B(nt) = \left| \frac{2}{T} \int_0^T r(t)\sin(w_1 t)dt \right|^2 + \left| \frac{2}{T} \int_0^T r(t)\cos(w_1 t)dt \right|^2 \tag{7}$$

The performance loss due to this receiver implementation decreases as the channel becomes more saturated. Even when the specular strength is twice that of the scattered component, the performance degradation may be considered negligible. [16] To evaluate the receiver performance, we approximate the averaging integral:

$$B(nt) = \left| \frac{2}{T} \int_0^T b e^{j[wt + \theta]} \sin(w_1 t)dt \right|^2 + \left| \frac{2}{T} \int_0^T b e^{j[wt + \theta]} \cos(w_1 t)dt \right|^2 \tag{8}$$

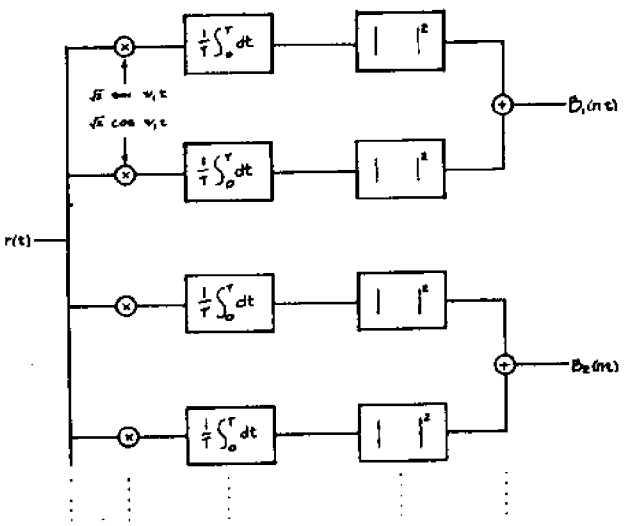


FIGURE 2: DATS Demodulator Processing Schematic

The expression is difficult to deal with with unless b is assumed constant during each integration. We conveniently assume that the bandwidth of b is much less then 1/T, the reciprocal of the integration time, and hence b may be taken outside the integral. For the data presented, this assumes the bandwidth of b to be much less than 100 Hz. Then, if T equals an integer number of carrier cycles:

$$B(nt) = \left| b(nt)e^{j\theta} \frac{2}{T} \int_0^T \sin^2(w_1 t)dt \right|^2 + \left| b(nt)e^{j\theta} \frac{2}{T} \int_0^T \cos^2(w_1 t)dt \right|^2 \qquad (9)$$

Where b(nT) is the averaged envelope at n^{th} receiver integration and has the same probability density function as b(t). Squaring and summing the two quadratures

$$B(nt) = b^2(nt) \tag{10}$$

Set $B(nT) = b^2(nT)$. Then the density of $B(nt)$ may be written as

$$P_B(B) = \frac{1}{\sigma^2} \exp\left[\frac{-(B + a_1^2)}{2\sigma^2}\right] I_0\left[\frac{\sqrt{B}a_1}{\sigma^2}\right] \tag{11}$$

or

$$P_B(B) = \frac{1}{\sqrt{\pi\sigma^2}} \exp \frac{-(B + a_1^1)}{2\sigma^2} \sum_{i=0} \frac{(a_1^2)^i B^i}{(\sigma^4)^i} \frac{r(i + \frac{1}{2})}{r(i + 1)} \tag{12}$$

which is the noncentral chi-square distribution with two degrees of freedom and the noncentrality parameter equal to the square of the specular component, a_1.

The DATS next adds M B(nT)-s (corresponding to M different tone frequencies) to produce a statistic corresponding to the total received energy for a particular code word. This yields a noncentral chi-square density if the individual B(nT)-s are independent and their variances are equal. It is reasonable to assume that the ratio of specular to scattered energy is the same for all tones, and that the additive noise is white. Then the equal variance assumption is valid. On the other hand, there is usually some correlation between envelope fading at different frequencies, and, in general, the envelope amplitudes may not be considered independent without examination of the carrier frequency separation and the channel behavior.

2.2 Experimental Results

To estimate the correlation functions for modelling channel behavior and the resulting DATS performance, a number of CW experiments were carried out. The measurements were collected in the harbor at Woods Hole, Mass., with the DATS system. The transmitter broadcast a "chord" of eight equally-spaced unmodulated tones. The tone spacing was fixed at 1000 Hz. The received and amplified data were quadrature demodulated, digitized, blocked, and fed to an array processor which performed 128-point complex FFT computations. The squared magnitude of the FFT was saved for further processing. Each data frame length was fixed at 10 msec for envelope measurements. The processing was performed in real time on the continuously-sampled input, so no time gaps appear in the data. Typically 3 to 5 minutes of data

were processed during each experiment, and the results saved on a floppy disk. The output consists of sixteen FFT output points from each frame. These include the eight points at the broadcast frequencies, and interspersed between those, eight data points representing the additive noise energy halfway between the broadcast tones. A time series of this data collected over a 20 meter shallow water path is shown in Figure 3. This plot represents the encountered fading behavior for most propagation paths, as detailed fading characteristics are difficult to distinguish from short records presented in this format.Note that the log of the squared magnitude of data amplitude is plotted. It is seen that even at this short a range, considerable fading is occuring. Also note the considerable frequency dependence of the received energy. The eight tones have equal energy before transmitting. The spectral shading is due to the nonideal transfer functions of the transmitting array, receiving hydrophone, and the demodulating and filtering electronics. Attempts to compensate for the system shading and measure the channel transfer function were unsuccessful because the transmitter array gain was frequency dependent along any direction but broadside. Transmitter pointing errors resulted in several dB differences in signal magnitude across the frequency range of interest.

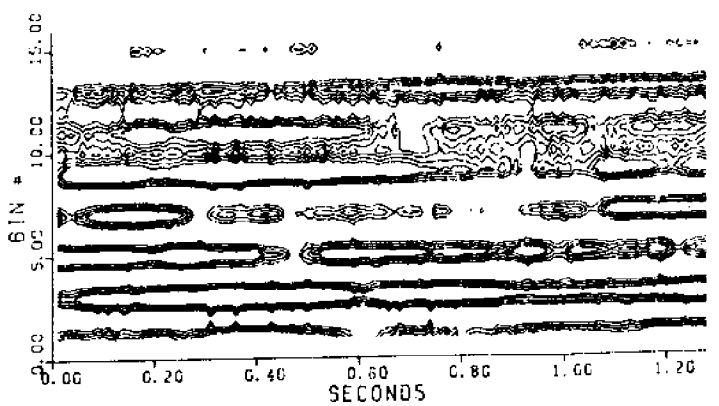


FIGURE 3: Fading Contour Plot

Histogram estimates of the probability density functions of the log of the 16 FFT outputs are shown in Figure 4.

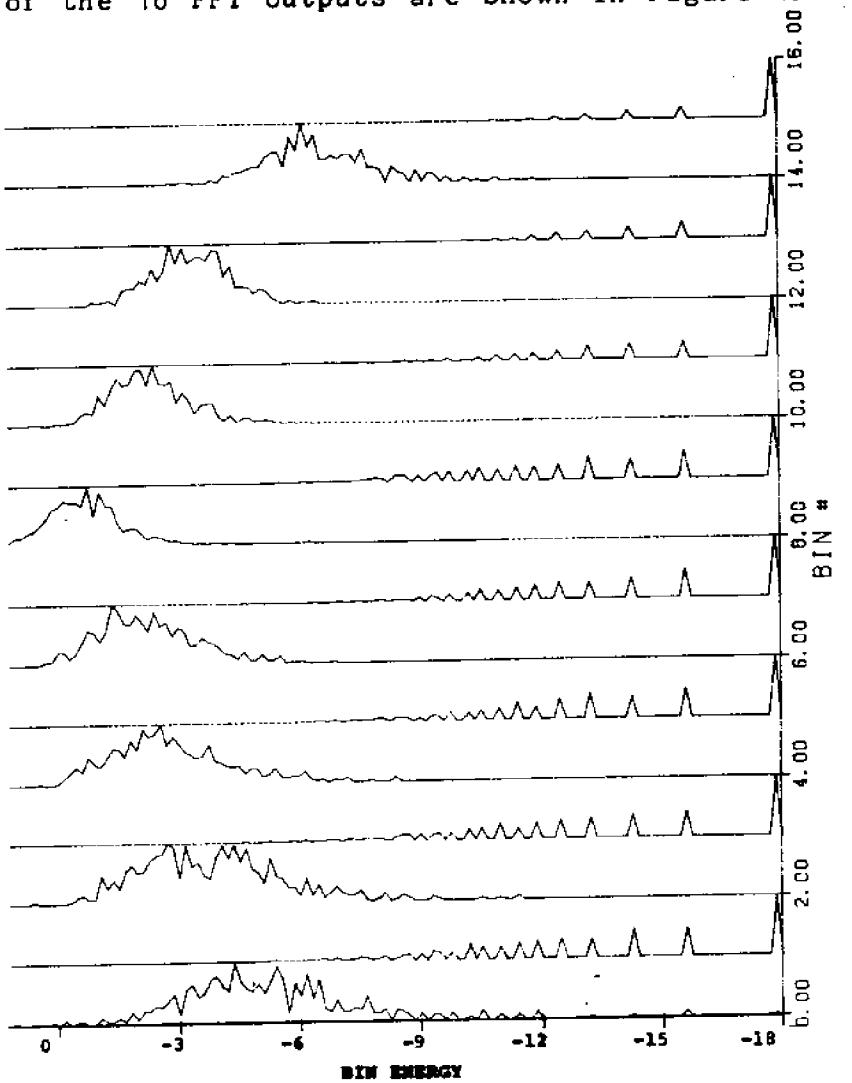


FIGURE 4: PDF Histograms - short path

Four thousand data points are processed for each channel. The plots are normalized by the most frequent (highest) value of each histogram to ease plot formatting, and the vertical scales for the 16 tones may thus be different. The abcissa range is the same for all 16 functions, so that the relative energy distributions in the 16 channels may be directly compared. The data in this format represents the density function of the squared signal envelope.

Figure 5 shows results of an identical experiment for an approximately 800 m long path across Woods Hole harbor. The propagation path was in water approx 20 to 60 ft deep, with the

transmitter fixed approximately 6 ft off the bottom in 30 ft of water and the receiver close to the surface. During the experiment the sea surface, agitated by a 20 mph wind, was generally foamy with a considerable number of breaking waves. Since the harbor is sheltered, the waves were seldom higher than 1ft.

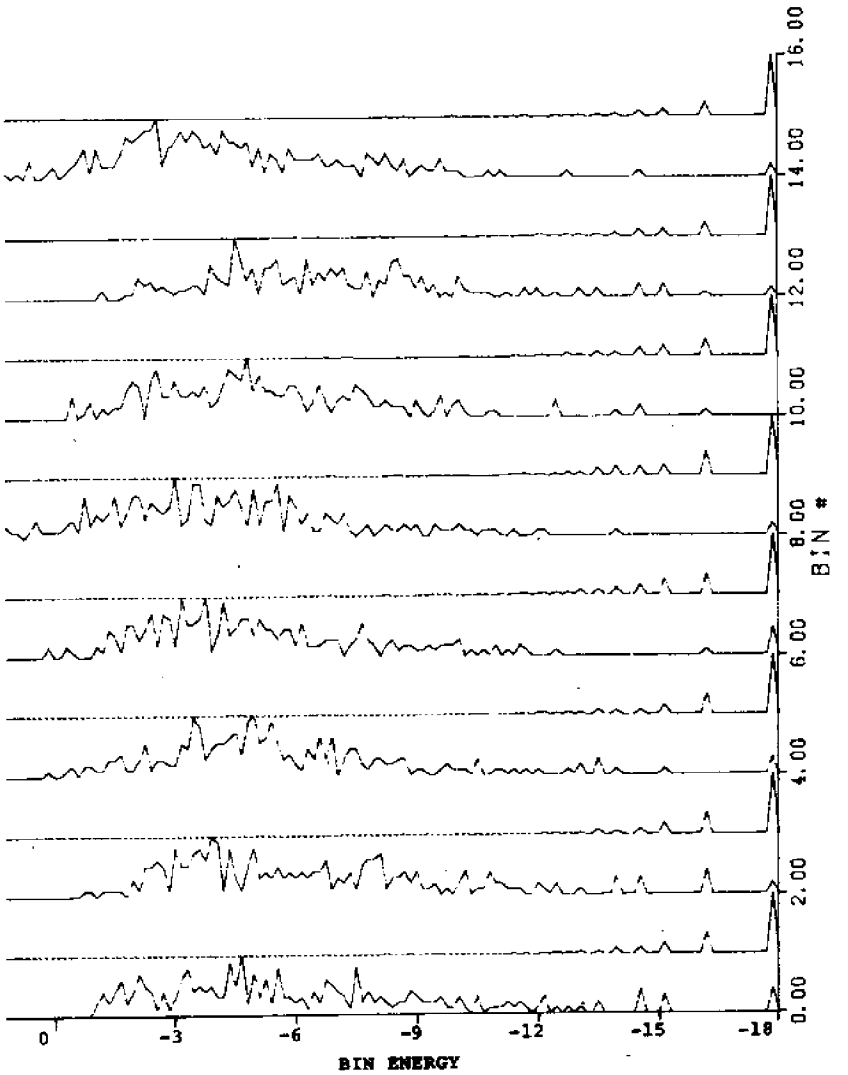


FIGURE 5: PDF Histograms - long path

As expected, the relative strength of the specular path decreased with increased source-receiver separation, but not as drastically as might be anticipated from the 20 meter results. The longer-range experiment was done with more open surroundings, and excited fewer secondary paths, but the primary path was also more attenuated, i.e. both the direct and the reverberant fields were lower for the second experiment.

Figures 6 and 7 show the time correlation functions of the 16 FFT outputs.

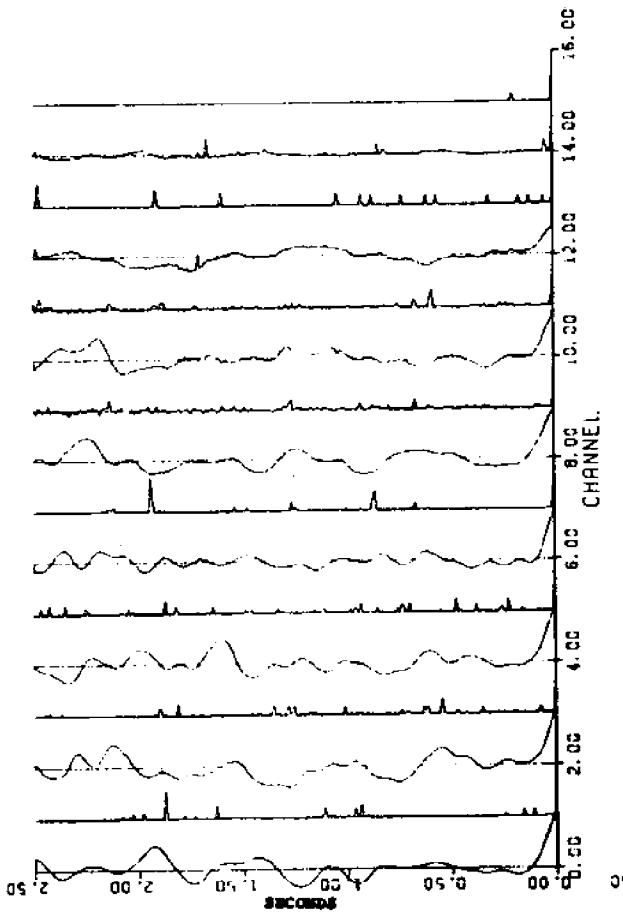


FIGURE 6: Time correlation short path

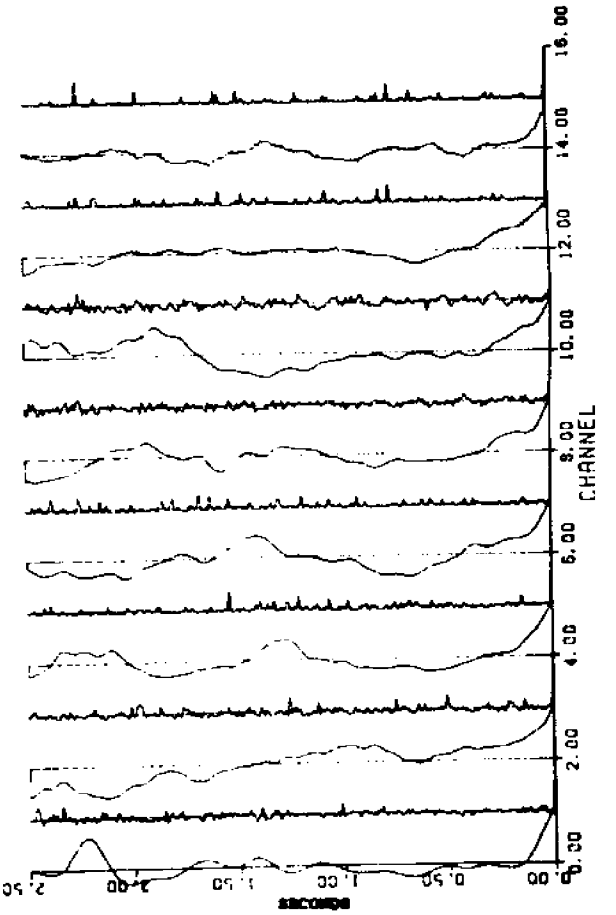


FIGURE 7: Time correlation long path

500 data points per channel were used to estimate the first 250 lags of the autocorrelation functions. The time series was assumed to be stationary. The 8 noise channels show no time correlation. As seen in Figures 4 and 5, their envelopes appear Rayleigh, and it is reasonable to conclude that the noise is well modelled by the white Gaussian assumption. The data channels exibit a characteristic correlation time of several tenths of a second, and it is seen that the longer-range signals fluctuate more slowly. If a deep fade should occur, the signal may be expected to stay "dropped out" for several tenths of a second.

The longer features (approx 1 sec) of the envelope correlations are believed due to the fluctuations of the specular path. In particular, the motions of nearby berthed ships were noted to be correlated with the tone amplitudes. In general, such slow fluctuations of the specular path are compensated by the receiver equalizer, although an extensive signal fade will cause loss of data.

Since a digital transmission error for a coded MFSK system requires a simultaneous dropout of several data channels, the frequency correlation of the tone fading is of interest. This data is useful for determining the distribution of implicit diversity paths in the channel [17].

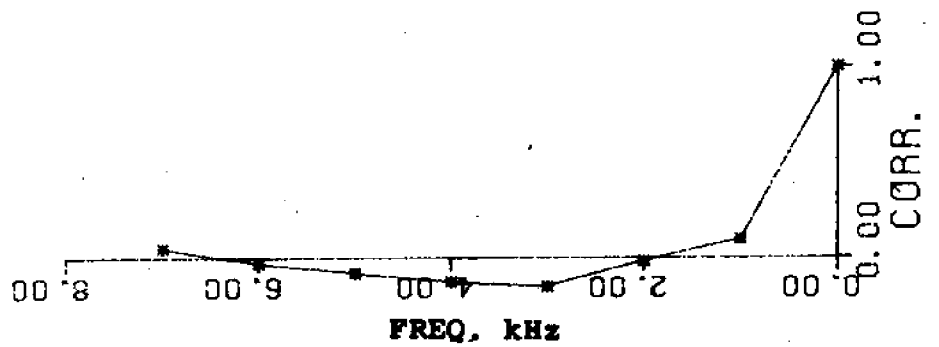


FIGURE 8: Freq. correlation - short path

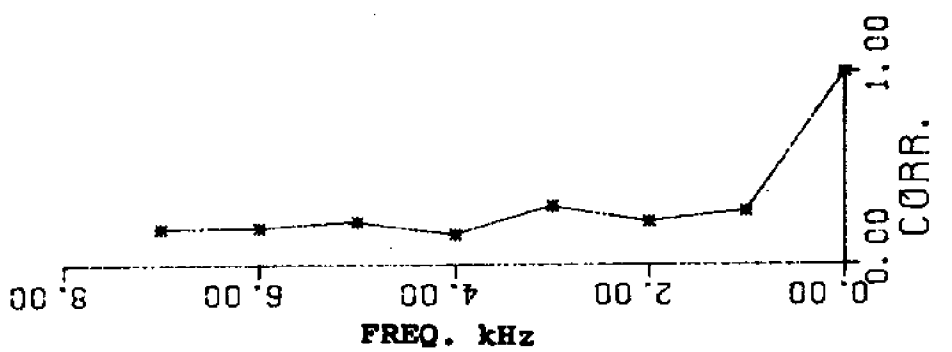


FIGURE 9: Freq. correlation - long path

Figures 8 and 9 show the frequency correlations of the eight data tones. 500 data points per channel were used to estimate the eight correlation lags for the eight data channels. Both plots

indicate some frequency correlation of the data tones, and suggest that a single diversity path occupies at least one kHz.

The DATS is designed for incoherent data transmissions over a fading channel. Such a channel may be considered as composed of many signal scatterers, each of which contributes a delayed and Doppler shifted signal replica to the received waveform. When the number of scatterers is large, it is convenient to consider the average of scattered energy received after a certain delay and Doppler shift. Such information is expressed in terms of the channel scattering function, which depicts the average received signal strength versus time delay and Doppler shift [17 - 20]. Other quantities of interest closely related to the channel scattering function are the two-frequency correlation function, F(dw,t), and the narrowband envelope time correlation function R(t ,t) . They are related by the following transforms of the channel scattering function.

$$R(t_0 t_1) = \iint S(\tau,f)\exp[j^{2\pi}f(t_0-t_1)]\, d\tau df \qquad (13)$$

$$F(\delta\omega,t) = \iint S(\tau,f)\exp[i\delta\omega\tau = 2^{\pi}ft]d\tau df \qquad (14)$$

where

is the channel scattering function.

The scattering function is a more complete channel description than either F() od R(), but its measurement in the DATS environment requires very high resolution processing. For use in system design, we found direct measurements of F() and R() more expeditious, particularly since the system configuration is based on rough estimates of the two functions and is insensitive to their detailed structure.[17]

The first moments of F(w ,w) and R(t ,t) represent the RMS spread of received energy in the frequency and time domains. They are important parameters in the design and performance of the DATS. The RMS frequency spread, B, influences the total received tone bandwidth. If the bandwidth of the transmitted tone is W, the received bandwidth is W+B, so the minimum tone spacing must be set at W+B.

If the mean Doppler shift is not corrected for, B represents the total Doppler drift and spread, but the DATS receiver effectively removes the mean of the Doppler drift. B then

represents only the RMS scatter around the mean Doppler shift, and is consequently much smaller, allowing the system to improve bandwidth use.

The RMS time spread of the received waveform, T, influences the temporal correlations of the received waveforms. If T is comparable to or larger than the tone duration, L, intersymbol interference results since each word becomes contaminated by delayed echoes of the previous transmissions. To reduce the effects of time spreading, the DATS modulator is capable of frequency hopping. This operation is illustrated in Figure 10. For this example, the tone frequencies are separated by increments of 250 Hz, and the modulator hops to a new set of frequencies for transmitting each word. The hopping partially clears the channel of multipath and reduces the degree of saturation and intersymbol interference at the expense of additional decoder errors while tracking the hop band sequence.

3 DATS DESIGN OVERVIEW

3.1 Coding for the Underwater Channel

The design of the DATS is based on the above channel model. Since it is designed for remote power-limited locations, the power consumption in the transmitter is minimized while maintaining the target 1200 bits/second data rate. Since power is strictly rationed on most underwater instruments, performance increases through coding are a promising alternate. Implementatins of such systems are greatly simplified by using the commonly-available microprocessors. Now significant amount of coding and pre-processing power can be packaged on data transmitters with minimal space and power overheads.

The DATS system utilizes an efficient coder and modulator based on an 8085 CPU chip; it is capable of coding an incoming data word into up to 16 tones and transmitting them simultaneously. A variety of coding algorithms may be used. During initial system testing, an (8,4) Hamming block code was used. Such a code is particularly well suited for a soft decoding algorithm, since a hard decoder could only correct single bit errors and detect 2 bit errors. The data was shifted in 4 bits at a time, and the 4 bit word used to generate a set of 8 tones transmitted into the water. The tone groups (or chords) were selected so that any one differed from all the others by at least 4 tones, i.e. the minimum code distance is 4. A set of convolutional codes with constraint lengths of 3 to 6 is available and is being tested, but the results are not reported in this work. The DATS was designed for efficient testing and evaluation of various coding algorithms, and it is relatively simple to modify the coding process even during a sea deployment of the instrument.

Now we consider implementing several coding techniques on the available telemetry system hardware and concentrate on the software/hardware structures required for real-time decoding of the DATS codes.

It is well known [18,21,22] that the error probability of a single path in a strongly fading channel decays only algebraically with increasing signal-to-noise ratio, and that faster exponential delay may be achieved by optimally exploiting the diversity structure of the channel. High error rates occur during deep signal fades, and one must design the signal to be insensitive to such a dropout, which extends over several tenths of a second and several kHz in the channel of interest. If a signal is to be transmitted and recovered effectively in the presence of such a fade, it must be larger than the fade in either bandwidth or time. Optimizing the signal bandwidth can be done by diversity techniques, where the number of independent data paths is determined as a function of available power,

bandwidth, and channel characteristics. The time duration of a signal may be extended by convolving the transmitted data stream with a preprogrammed code, thus "smearing" each signal over a longer time duration.

The optimum diversity tehniques have been applied to many communication channels, including atmospheric and deep space propagation, but not enough is known about the high frequency underwater acoustic channel yet to understand its fading behavior. The pioneering work at describing low-frequency acoustic fluctuations points to a unique set of mechanisms responsible for the signal fading,[14, 23] and there is reason to believe that ocean acoustic fading is sufficiently different from the previously-studied examples to warrant a different, currently unavailable approach. The DATS design and the coding used are designed for the fading channel as modelled in the previous section. Such a model is primarily applicable to a number of radio propagation channels, and our experimental results indicate that the ocean acoustic channel may be parametrized this way, although this may not be the optimal way of describing it.

The initial code implemented on the DATS relies on signal bandwidth and frequency hopping to overcome the effects of fading. It is an (8,4) Hamming code where each 4-bit data word is coded into 8 bits as per Table 1.

TABLE 1

input word	coded word	code weight
0000	00000000	0
0001	00011110	4
0010	00101101	4
0011	00110011	4
0100	01001011	4
0101	01010101	4
0110	01100110	4
0111	01111000	4
1000	10000111	4
1001	10011001	4
1010	10101010	4
1011	10110100	4
1100	11001100	4
1101	11010010	4
1110	11100001	4
1111	11111111	8

This code is linear with a simple distance distribution. Its implementation is perhaps best understood by examining Figure 10, which shows a spectogram of an actual transmitted sequence. The signal consists of isolated groups of eight FSK tones, or "chords", which are hopped in frequency to reduce the intersymbol echo interference. The spacing of tones within each chord is different for each valid codeword, and thus the chord duration is the same as the time required to transmit a single data block.

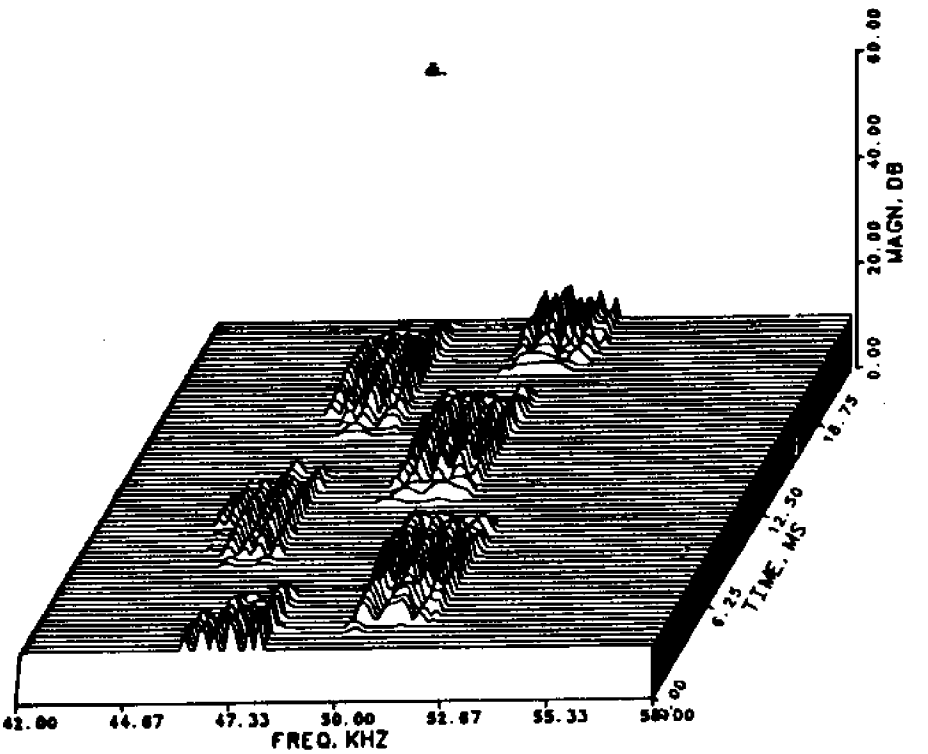


FIGURE 10: Frequency Hopping Spectogram

The tone corresponding to each bit is upshifted in frequency for a "one" and downshifted for a "zero". The center frequency for each tone is also varied from frame to frame according to several predetermined "frequency hopping" patterns. In other words , the signal may be represented as:

$$r(t) = \sum_{i=1}^{8} \sin[(w + dw)t] \qquad (15)$$

where $w_i - w_{i+1} = 4dw$ is the same for all the tones, The tone frequency separation may equal 50, 100, 200, 250, 300, 400, 500, or 1000Hz. w_{i+dw} is sent if the i-th encoded bit is a "one", and

w_i-dw corresponds to a "zero". w_o corresponds to the chord base frequency and may be varied in increments of 1 kHz from a base frequency around 45 kHz. In the example illustrated, $w_i - w_{i+1} =$ 250 Hz, and the base frequencies change with each data frame with the offsets 0,4,1,5,0,... kHz from the lowest utilized base band. The system is designed to transmit equal energy tones, and the spectral shaping evident in Figure 10 is due to the system (primarily hydrophone) nonideal transfer functions.

The DATS Receiver is a digital implementation of the canonical incoherent correlator receiver [18] illustrated in Figure 2. The algorithm computes the energy of each hypothesized tone during one frame and adds the energies of the tones corresponding to all possible words. The word corresponding to the largest sum is declared the output. This operation is illustrated in Figure 11.

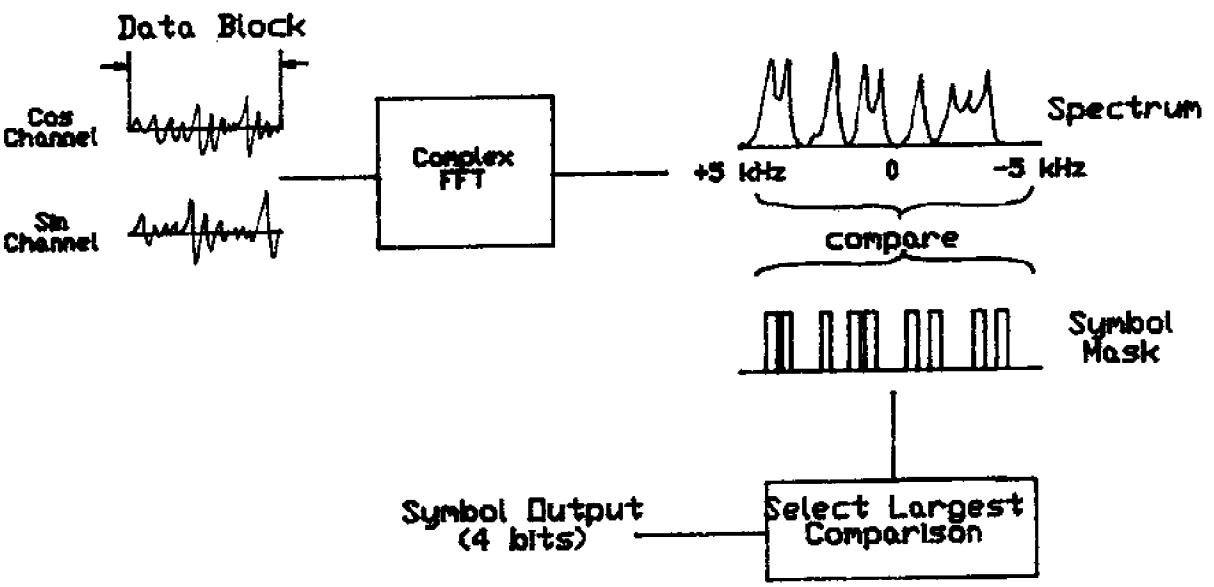


Figure 11: Signal Flow for Decoding

It is a standard implementation of the soft decoding method[24, 25, 26]. The decoding algorithm is written in Intel 8086 Assembler language, and the decoding algorithm is flowcharted in Figure 12. The emphasis was on time-optimized modular code suitable for pipelined or parallel execution on several processors. The numerically-intensive tasks, such as the data FFT and adaptive equalizer updating, are relegated to an array processor on the system. Although the numerical processing load

of the optimal receiver is significant, we were able to mitigate the computational problem by using a loosely-coupled set of processors for executing different parts of the decoding pipeline in Figure 12 in parallel. The resulting architecture allowed easy testing of code (on a single CPU) and at the same time removed any constraints on decoding speed arising from the computational complexity. The speed of the DATS as tested was constrained only by the available SNR per bit and the sampling duration required for the FFT resolution.

3.2 DATS DECODING SOFTWARE

The receiver tracks the frequency hopping pattern of the transmitter and matches the equalized spectrum of the received signal with the previously stored tone patterns. The four bits corresponding to the best-fitting (in the mean-square sense) tone pattern are chosen as the receiver output. The demodulator and decoder are both implemented digitally, and the flowchart of the algorithm to decode the (8,4) Hamming code is shown in Figure 11. Note that either four or five frames of data may be processed simultaneously.

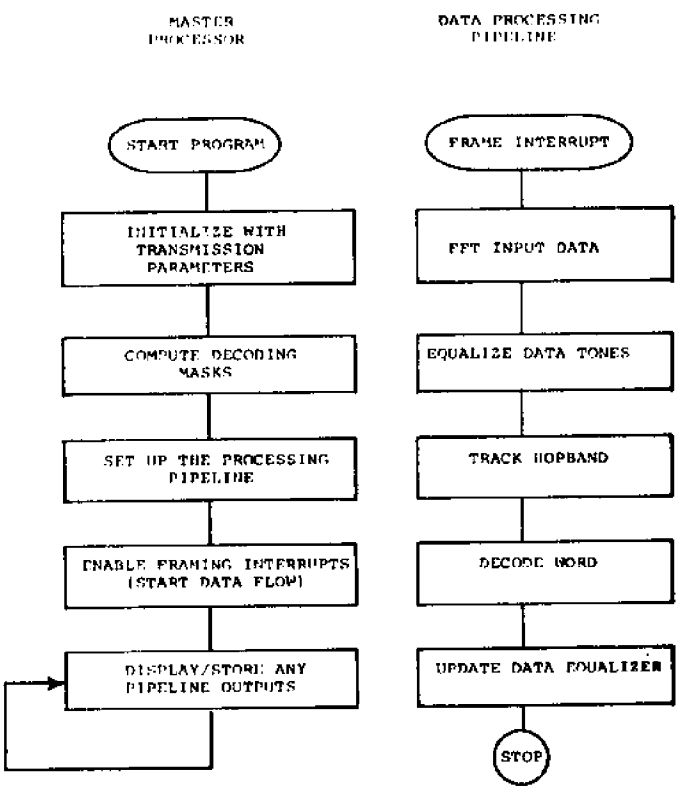


FIGURE 12: DATS Algorithm Flowchart

The data storage and display routines are not synchronized to the framing pulse, and operate whenever there is available time. They are prioritized, so if there is insufficient time to process all of them, lower priority routines are automatically deleted until there is time to process them again. For example, during normal system operation, the display routine often lags the decoder by more than 20 4-bit words. This occurs because the time required to decode and store a data word varies due to signal to noise ratios, error probability, and the instantaneous channel reverberation echoes. We now examine the Hamming code decoder software in detail. The implementation illustrates the capability of a microprocessor system used as a real-time decoder, and should be understood thoroughly before implementation of more complicated codes is attempted.

With the system properly initialized, the processing of a data frame is initiated by an interrupt to the main 8086 CPU from the frame tracking system. The system responds by initializing each of the components of the processing pipeline, which are shown in Figure 12, and are described below:

1) A new data buffer is initialized and passed to the control of a DMA processor, which is loaded with the buffer address and the number of 16-bit words to be transferred. The DMA processor inputting the previous data buffer is terminated, the number of words transferred is read, and the buffer is formatted (zero padded) to be available to the array processor. Typical execution time is 50 to 200 microseconds, depending on the amount of zero padding to be done.

2) The state of the array processor is checked. If the AP is busy, the system flags this condition and waits for the next interrupt cycle. This is a major system constraint. The time required by the array processor is largely fixed from frame to frame, varying only due to the differences in CPU => AP DMA transfer timing. Thus queing the data and waiting for a shorter AP execution cycle doesn't work.

If the AP is not busy, and has left no error messages from the previous frame, it is initialized with the address of the function control block (FCB) which contains detailed processing instructions. At this point the array processor begins work independently of the main CPU. The required processing consists of the following:

a) DMA of the frame data buffer from the main CPU memory

b) Complex FFT of the data buffer. The result is left in

bit-reversed order, and instead, the address offsets of the desired locations are bit reversed at system initialization. Thus bit reversal need not be done during each frame.

c) Magnitude squared of the FFT result data buffer is computed and optionally summed with the equalizer buffer. The equalizer buffer tracks any slow changes in the data buffer spectrum by averaging the FFT buffers and comparing the averaged spectrum to the equalizer coefficients. The averaged buffer is updated and corrected by the known energy distribution in the channel by a low priority asynchronous task which also modifies the buffer's dynamic range to keep the entire mask within the AP-s 24 bit range. A better version of the equalizer would track the energy in the required tone bands employing a dedicated processor, and we intend to implement one of the "standard" adaptive equalizer algorithms. The algorithm used was quite effective in correcting stationary variations in the frequency-dependent gain, but its performance on the time-variant components of the transfer function was at times suspect.

d) The FFT output is divided by the equalizer coefficients in an attempt to remove the effects of system shading. An equivalent routine operating in the main CPU exists and is used at slower frame rates and off-line because its operation and outputs are more easily observable.

e) DMA of the result data buffer back to the main memory. The result is a bit-reversed equalized spectral estimate of the input data stream.

Typical execution times are 20 to 50 microseconds for the main CPU and 1.5 to 3 milliseconds for the AP. In addition, the AP-initiated DMA may additionaly slow down the system.

3) If there is a coprocessor available on the bus, and the transmitter is frequency hopping, the main CPU checks whether the coprocessor has completed determining the most likely hopband of the previous data frame (one which was processed by the AP two frames ago).There are 8 possible hopbands. The hopping sequence is tracked by first summing the energy corresponding to the 16 available within a given hopband. The resulting 8 numbers represent the likelihood ratios for the eight hopbands. Spectral overlap of adjacent hopbands is roughly corrected for by multiplying each number by a coefficient which depends on the spacing between the tones. Since channel memory, i.e. non-zero channel impulse response, is anticipated, each number is then scaled by a decision-feedback algorithm which estimates the residual energy due to energy transmission at previous hopbands More recent hopbands contribute more energy. The algorithm does not determine the channel impulse response adaptively; this information is entered by the operator at start-up time. It is realized that these coefficients should be optimized with respect to the channel behavior and the probability of loosing synchronization with the transmitter, but the following method was found to give satisfactory performance and is easy and quick

to implement: The hopband corresponding to the largest adjusted sum is then picked as the current hopband and the algorithm is updated.

Since the time required to determine a hopband varies with the number of hopbands used, and may change during the transmission, the main CPU will wait for the program to complete before proceeding. Although this could pose a potential botleneck, this hasn't happened since the hopband algorithms used took less time than the AP to process a frame. If a problem does occur, it would be solved by adding another coprocessor or transferring some of the work to the AP. Typical execution times are 20 to 50 microseconds for the CPU, and 1 millisec for the slave processor if only two hopbands are used. There are some DMA transfers to slow up the system.

4)If there is no coprocessor on the system, the main CPU determines the hopband and then decodes the data block. The decoding algorithm used is a standard soft decoding algorithm which compares the energy of all the 16 8-tone chords within a given hopband. The largest-energy chord is declared as the decoded output. Note performance would improve if the decoder determined the energy of all the 16xn possible chords possible with n hop bands used, but the execution time would be prohibitive. Although the predicted performance degradation is beyond the scope of this text, it was observed that hopband errors due to the hopband-tracking system were substantially less frequent than other decoding errors, and occurred significantly often only in the presence of synchonization errors.

If another framing interrupt occurs before the above algorithms complete executing, the CPU abandons the data frame, after optionally marking it as a "did not finish" or "blank" data frame. If the word is decoded in time, desired decoding parameters such as SNR, chord energy, and the decoded word are saved in appropriate buffers for later processing and storage. The additional time required to decode a word is approximately 1.5 millisec if frequency hopping is not used.

5)The CPU returns from the frame interrupt to the background processing shell. Among the programs which may be executing in the background are:

a) Save the AP output buffer on disk - this takes a considerable amount of time - about 1.5 milliseconds (msec) for a 128-word block - and will generally execute only if no other option is available, at the frame rate of 250 hZ. In addition, the system must shut down frequently for bulk disk transfers. It is a useful option when checking hopband tracking algorithms and overall spectral shaping.

b) Save the FFT output points corresponding to the 16 possible tones within a given hopband. This routine was used to record the time series displayed in Chapter II. It is much faster than a) because only 16 data words per frame are stored and has been found more useful, but the decoder must still shut down

every 256 frames as only 4k words may be output to the floppy disk at a time.

c) Save the decoded word stream on disk. This routine is useful for manual error checks and decoder behavior during synchronization errors and decoding time overflows. It consumes little time, and large blocks of data may be stored, so that less probable events and failure modes may be recorded.

d) Save the average probability of error and SNR during a transmission. This often-used routine operates on a repetitive data transmission, and attempts to match the received data stream to a stored data pattern. It operates sucessfully at error rates of up to 20 %. If more than a certain programmable (usually 8) number of errors is detected in a row, the system attempts to synchonize to a different data sequence, assuming that a synchronization error or time overflow has occurred. The resynchronization algorithm may require up to 10 msec, and largely determines the speed and the usability of the routine. The data stream is buffered in a 2k data buffer, and the routine often operates with several hundred decoded words in the buffer. During the low-error periods, it only uses 300-500 microseconds and catches up rapidly. The SNR is computed by dividing the energy found in the tones corresponding to the decoded word by the energy in the eight other possible tones. This sets the upper limit on the actual SNR, since the SNR computed for any data block is greater than one. It is, in fact, the correct SNR given 0 error probability, since the largest energy word chosen is not necessarily the one transmitted. The SNR computation proceeds asynchronously from the error probability routine. The data is saved on disk as average SNR for each 256 frame block and the number of errors and resynchronizatuion attempts in each 256 frame block Each two words stored correspond to the same 256-frame block. The routine will operate at 1 frame pulse per 10 msec at error rates of up to 20 %, and at error rates of up to approx 5% at 1 pulse/4 msec. At this rate, it is dependent on the type of errors occurring, as each synch error consumes more time than a decoding error.

e) Display the decoded word on the system screen. This is primarily used for a visual check of the decoder. When used in the lowest priority, it indicates the load on the decoding system. It consumes little time, and was usually enabled. Certain actions, such as a error probability buffer overflow, may automatically disable it to indicate a serious error event.

3.3 DATS Capabilities

It is appropriate now to consider the ultimate performance capabilities of this system. Most of the above-mentioned performance criteria may be significantly improved, although at considerable cost. For example, most of the 8086 code can be speeded up twofold by substituting a 10 mHz processor (in place

of the 5 mHz CPU used) and a memory card fast enough to operate without wait states. The real limits of the system, then, lie in the SNR and the data sampling/resolution constraints. Decoding of even moderately complex codes need not present a problem with the parallel processor structure described and the modern computer hardware.

The chord frame duration determines the data rate. Consider a frame 2 msec long. With the (8,4) code used, this translates into a 2 kbit data rate. At a typical sampling rate of 16 kHz complex 8-bit samples, the receiver operates on 32 complex data points, and the FFT output consists of 32 independent samples spanning the frequency range fom 42 to 58 kHz, i.e. a sample every 500 Hz.

While the number of available FFT outputs seems much larger than necessary for the eight data tones used, frequency hopping of the transmitted data requires more matched filters than transmitted tones at any given time, as the receiver does not know a priori the base frequency of the data tones. Up to eight hop bands are available, spaced 1kHz apart. The bit band spacing, as mentioned above, is programmable. If the eight tones are spaced in increments of 1000 Hz, then the entire processed bandwidth is used by the decoder, as the data band is 8 kHz and the guard bands for the other possible hopbands take an additional 8 kHz. Thus the maximum data rate attainable with this tone configuration is 2 kbit/sec. This should not be viewed as an absolute limit, but as a useful practical performance bound. The rate may be doubled by either doubling block size (to (16,8)) or doubling the bit band and the frame rate. In either case frequency hopping is now precluded. If the bit band is decreased to other separations programmable on the DATS system, the maximum data rate drops further because of additional resolution required from the FFT. With any rate 1/2 code, the maximum data rate can not exceed 2 kbits/second for any tone separation used. The channel is thus bandwidth-limited, or rather limited by the channel dispersion time of the transmitted data, which dictates the maximum "duty cycle" with which a carrier tone may be used.

During the system testing, no significant Doppler shifts were observed. If communication with a moving vehicle is desired, the system is equipped with a pilot tone to correct for any mean Doppler shift, and the Doppler scatter of more than several hundred Hz is unlikely. A more significant constraint on the tone spacing is the channel coherence bandwidth, Figures 8 and 9 show that for the two experiments conducted, the frequency spacing of independently propagating paths must exceed 1 kHz. The bit bands available on the DATS are generally considerably narrower than this. One gains no advantage by spacing signals closer than approximately 2 kHz other than increasing the signal power present in a single fading path. A discussion of the optimal diversity techniques may be found in [17]. One finds that, for the available DATS operating environment and SNR per tone

available, the number of independent paths should be maximized, that is, that the widest possible tone separation should be used. For this reason the data presented in Chapter 5 was collected with the largest possible tone separation (500 Hz).

4 DATS HARDWARE DESIGN

The detailed hardware design of the DATS system is described in the following section. It is instructive to view the system as an implementation of a canonical communication system, illustrated in Figure 13.

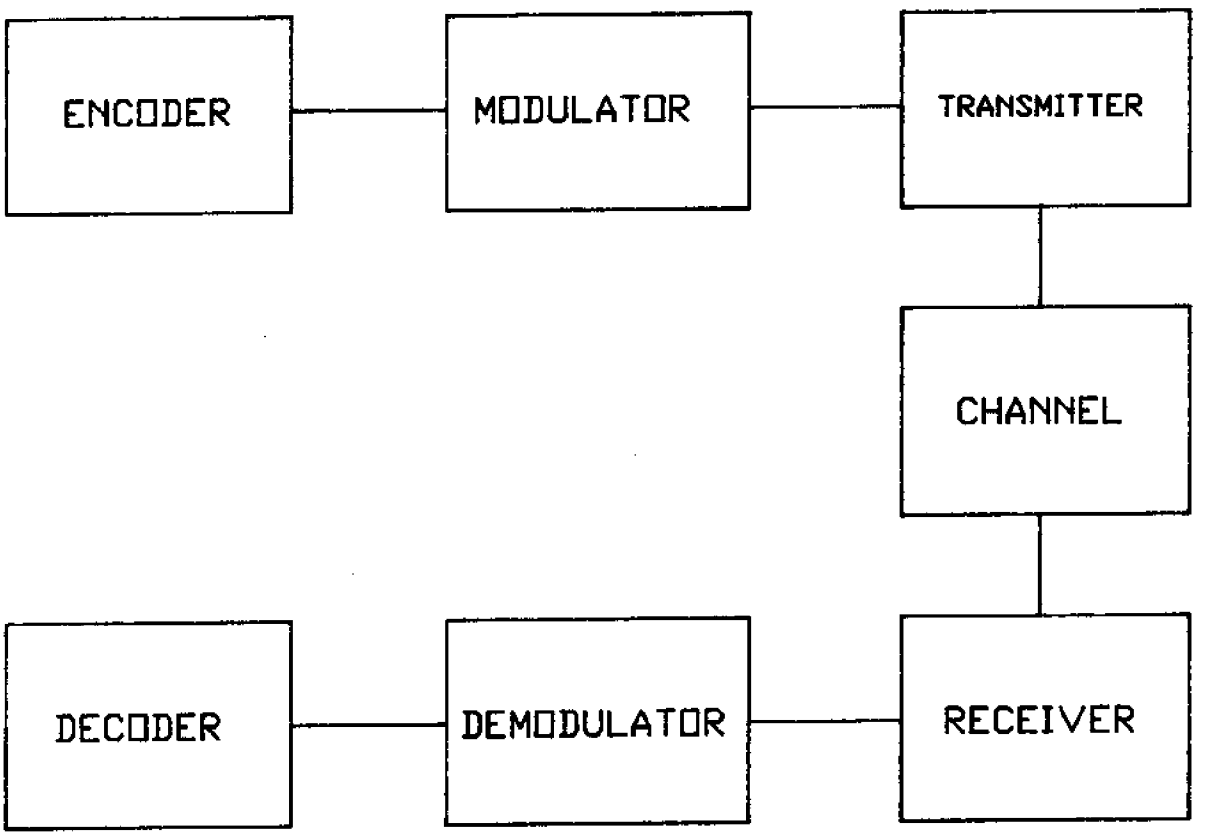


Figure 13: Canonical Communication System

The task of any such system is to send information from the data source to the data user across a noisy, (perhaps partially) unknown channel. Data is sent through the channel using a transmitter, which converts electrical signals into waveforms capable of propagating through the medium, and a receiver, which inverts the transmitter operation. The conversion of a data stream into a set of electric waveforms is accomplished by the encoder and modulator; the demodulator and decoder attempt to reconstruct the data stream from the received waveforms. Each unit may be considered separately although close parallels often exist between the various components.

The DATS encoder is implemented in software on an 8085 CPU. The encoding system processes the input data stream (provided by the system user for and end-user application or generated by the instrument itself in case of a performance test of the system), and outputs a sequence of digital data words to the modulator. The code words are generated either by a look up table for short block codes or by synthesizing a digital binary convolver when convolutional codes are used.

The modulator is a digitally programmable frequency synthesizer which generates a multi-tone "chord" corresponding to the input data word. The chord presently consists of up to 16 data tones, although the system may be easily expanded to more tones if necessary. Each tone is either upshifted or downshifted from a known center frequency; an upshift signifies a "one" and a downshift a "zero". These bit center frequencies are equally spaced and together comprise a chord which is transmitted through the water. Considerable care was taken to ensure the tone frequencies fall exactly on the computed FFT points when the received time series is demodulated so that the minimum possible FFT resolution may be used in the demodulator. To reduce the effect of channel reverberation, the chords may be "hopped" in frequency. Thet is, the chord center frequency changes periodically. The frequency band of the data tones extends from 45 to 55 kHz in the present system configuration.

The tones are generated by a bank of CMOS Phase-Locked Loops (PLL-s) capable of rational division of a common master clock. The master clock is implemented with a simple crystal oscillator at 2 MHz. The divisors are loaded into the PLL-s by the control processor. To avoid the sometimes lengthy loop settling times when the PLL-s are loaded with new frequency divisors or multipliers, all needed frequencies are generated and then mixed digitally to select those corresponding to the desired word. At each frame, the decoder selects:

1) The coded word to be displayed. In the configuration tested, this was an 8 bit word derived from a 4 bit input nibble.

2) The hopband used. To reduce reverberation effects, the center carrier frequency for the tone group may be changed periodically, as seen in Figure 10. The modulator steps through a deterministic sequence of carrier offsets. The hopband may be varied in increments of 1kHz, and may be changed after any chosen number of data words. Under normal operation, the hop band is changed after every transmitted word, or not at all. The hopband sequences are designed to spread out the energy as evenly as possible over a given bandwidth. We have also used some simpler "training" sequences for bringing the hopband tracking and decoding algorithms on line.

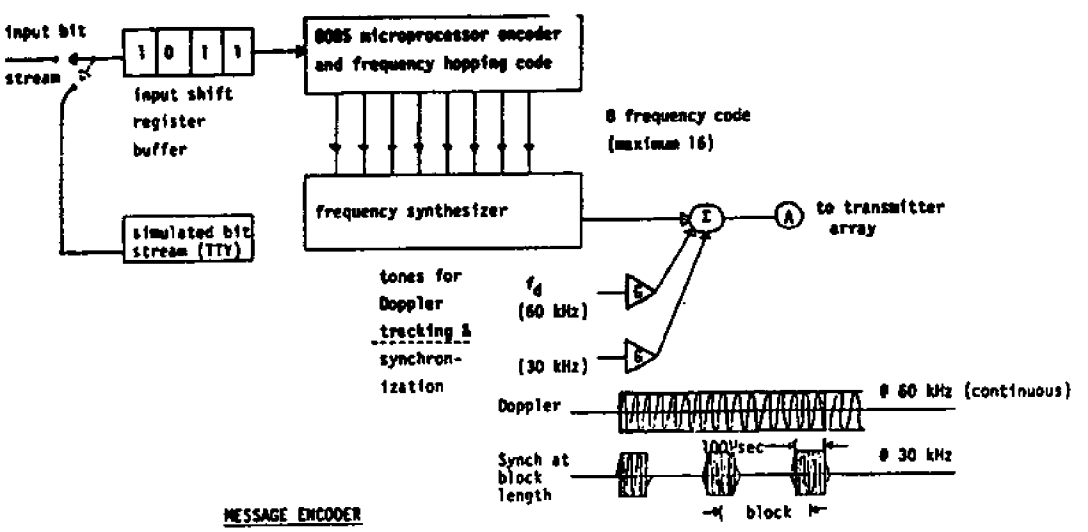
3) The base frequency offset of the tone group. Since the data demodulator at the receiver employs a fixed quadrature demodulating frequency, this feature may be used to align spectrum of the received word with the computed FFT coefficients.

The feature could also be used to avoid any particularly noisy parts of the spectrum or transfer function nulls, but we did not use adaptive spectrum selection in this way.

The following features are also under software control, but are variable only between transmissions, as the decoder is unable to track the changes:

1) Bit band selection. This is the incremental spacing between the tone pair bands of the transmitted word. The available spacings are: 50, 100, 200, 250, 300, 400, 500 and 1000 Hz. This change involves reloading the PLL-s with new frequency divisors, and the required settling time may be as long as several seconds. This feature is useful in determining the properties of the channel frequency correlation function and the distribution of diversity paths, and was used extensively during our channel tests. For data transmission the widest bit band was generally the most useful, as it allowed use of coarser FFT resolution and increased the number of diversity paths.

1) Frame rate selection. The frame rate is essentially the data transmission rate. One frame corresponds to one data word, which, with the (8,4) code used, corresponds to 8 transmitted tones and 4 data bits. The beginning of each frame is marked by the transmission of the synchronization pulse, which is a short (adjustable from 50 to 500 microseconds) tone burst centered at 30 kHz. The frame rate is varied by loading the time duration into a countdown timer which then interrupts the CPU periodically.



Examples: 1) 2 ms/4 bit block → 2 kbit/s
8 tones → 8 kHz total bandwidth with +/- shift (bandwidth expansion = 4)

2) 1 ms/4 bit block → 4 kbit/s
8 tones → 8 kHz total bandwidth with 1/0 keying (bandwidth expansion = 2)

3) 4 ms/4 bit block → 1 kbit/s
8 tones → 8 kHz total bandwidth with +/- shift and frequency hopping 4 kHz/block

FIGURE 14: DATS Modulator

The modulator also generates a continuous 60 kHz pilot tone which is used to correct for the mean Doppler shift in the channel. The tone is present whenever the instrument is powered up. It is generated by frequency division of the base crystal reference. By tracking this frequency reference, the receiver is able to correct for any clock drift, as well as for the channel Doppler offset. The modulator is diagrammed in Figure 14.

The transmitter is a phased array. It consists of a 4 X 8 hydrophone array eith the -3dB beamwidth of approximately 7.5 and 15 degrees. The array elements were purchased from the International Transducer Corporation in Goeta, California. The initial design called for the one-half wavelength spacing at resonance (50 kHz) to preclude any grating lobes and provide a reasonable degree of side lobe control as well. The required physical design of the transducers precluded the use of this spacing. Efficiency and frequency response needs resulted in a transducer element diameter of 2.2 cm, larger than the 1.5 cm 1/2 wavelength at 50 kHz. The final spacing of 2.41 cm was the minimum that could be obtained while still preventing contact between the transducers.

Each transducer is a standard longitudinal vibrator design, with piezoelectric ceramic drivers mass-loaded by quarter-wavelength resonators. In this case the head mass located at the top of the transducer is made of aluminum and forms the radiating face of the driver. The taper attempts to negate some of the directionality inherent in a piston type sound source whose ka value (where k is the wave number and a is the diameter), is much larger than that of a unidirectionally radiating element. While the ultimate objective of the array is to focus the radiated acoustic energy in a very narrow beam, for steering purposes it is desirable that the individual transducers closely approximate a uniform hemispherical response. Two lead zirconate titanate (PZT) ceramic rings make up the active part of the transducer; they are wired in a center driven mode which places the acoustic center of the transducer at the joint between the cylindrical pieces of ceramic. The heavy tail mass located beneath the PZT rings is mild steel. The entire assembly is cemented together and then precompressed by a stress bolt located through the center of the stack. The prestressing is needed to avoid shattering the array in tension by overdriving it.

The array housing is shown in Figure 15. The transducers were mounted with noncompliant epoxy under pressure, forming a uniform thin joint to a thin plate of mild steel, with holes provided to channel the wiring down into the lower part of the housing. The plate serves to acoustically uncouple the transducers. The assembly is then mounted to the backplane of the upper array housing at the distance of 1/4 resonant wavelength to cancel out sound reflections from the backplane.

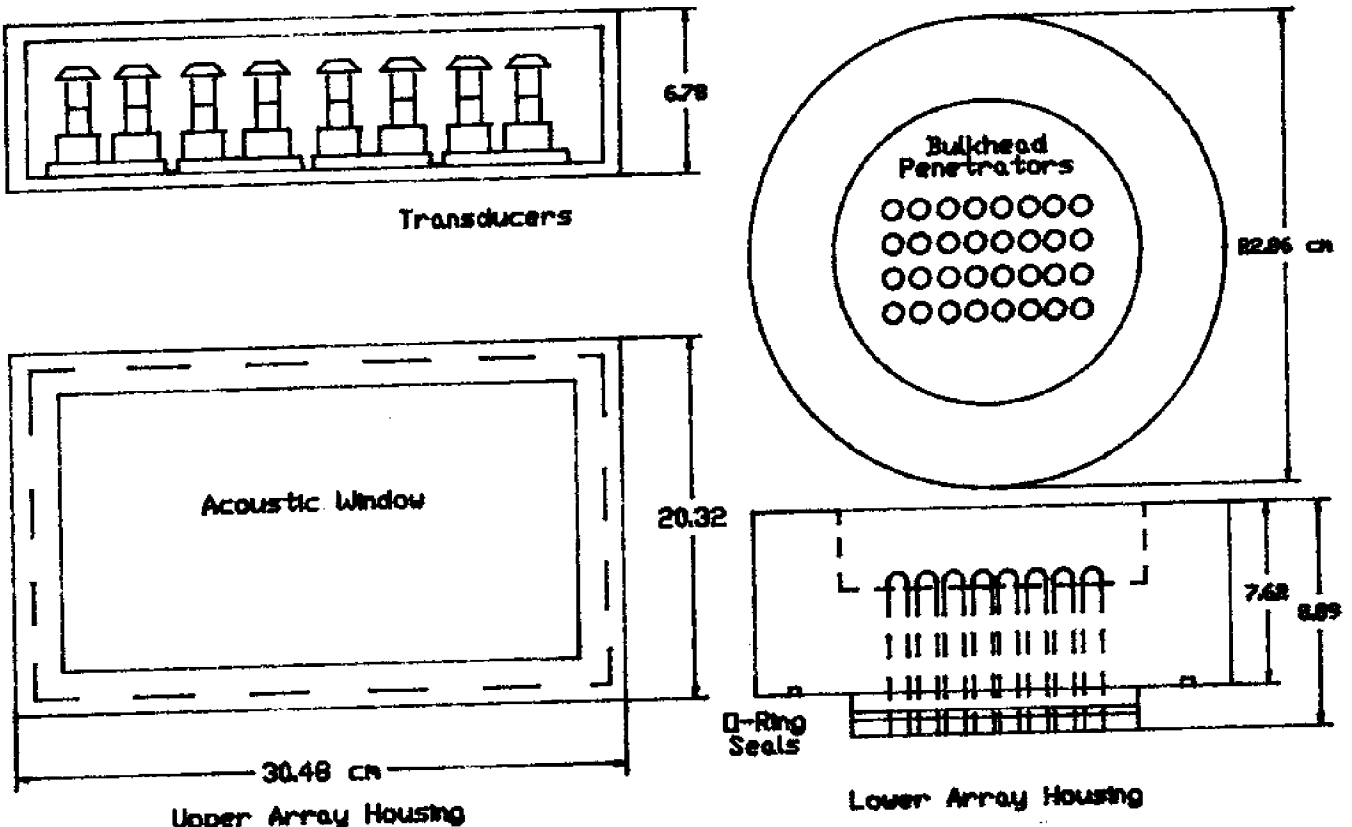


Figure 15: Array Housing

The dimensions of the upper array housing were dictated by the need to provide a clear acoustic path for any degree of steering up to 45 degrees. It is constructed of 0.95 cm thick 6061 aluminum alloy plates. Holes drilled into the backplane provide access to the lower array housing, and those located at each end serve as inlet and outlet for oil during the filling process. The top cover is fitted with an acoustic window composed of polyurethane, whose acoustic impedance closely matches that of water. The lower array housing contains the 32 watertight connectors for the array element electrical input. It is fabricated out of a solid piece of 6061 aluminum, and 32 "mini-mecca" connectors are embedded in it. The array housing is designed as an independent unit intended to be compatible with a number of instrument cases.

The array beam pattern may be computed as 15 by 7.5 degrees at the resonant frequency of 50 kHz. The 4X8 element pattern was selected to study the effects of spreading out transmtted energy horizontally and vertically, and as the most efficient layout

from the number of available transducers.

The array was calibrated and its performance measured in the MIT swimming pool.[27]. The tests consisted of checking the frequency response and the beam pattern characteristics of the array. Figure 16 shows the measured frequency response of the array. Figure 17 shows the observed array directivity along the narrow axis. A detailed analysis of the array performance is found in Hanot [27].

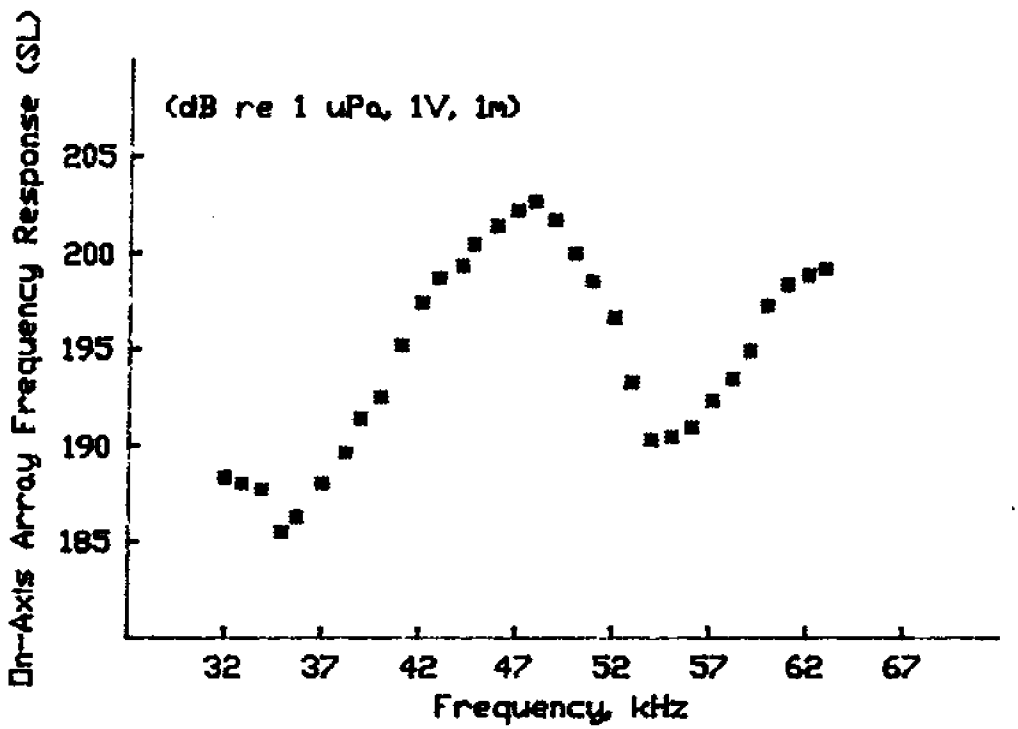


Figure 16: Array Frequency Response

The instrument is contained in two very similar pressure cases. One holds all the batteries that make up the power source for the system, and the other contains all of the processing electronics and the power amplifiers. Connections between the two are made through two 12-contact commercial watertight bulkhead connectors, located through the lower end cap of each pressure case. the transmitting array forms the upper end cap of the

electronics housing, and the two cylinders together form an entirely self-contained telemetry transmitter. The body of the case is constructed from extruded 7075 alloy. It has an outside diameter of 17.78 cm, a wall thickness of 1.27 cm, an an overall length of 109,22 cm. A ring is pressure fitted and pinned to each end of the tube to provide a means for attaching the end caps. The end caps feature O-ring seals to the instrument cases. they are made from 2.54 cm thick 7075 alloy plate. The pressure casing of the system is designed for a depth of 2000 m, as possible deep-water tests were envisioned.

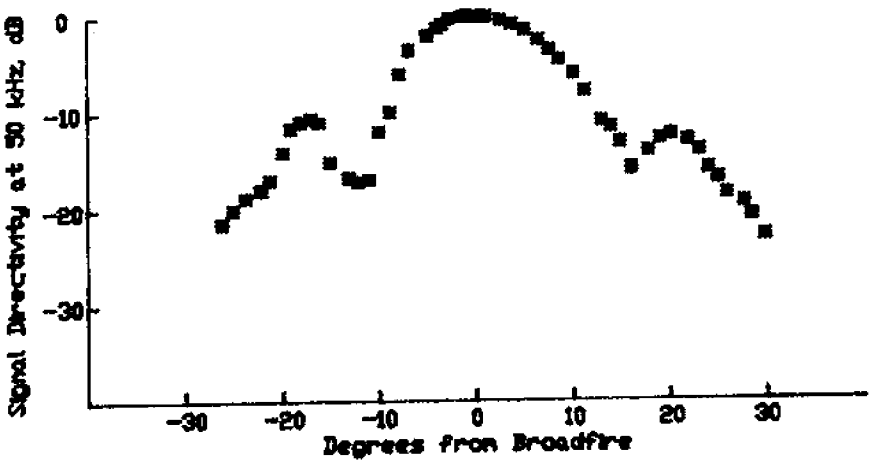


Figure 17: Array Beam Pattern

The array elements are driven individually by op-amp based amplifiers. Individual drivers are needed for array steering operations. The amplifier transfer function is flat +- 1 dB up to 75 kHz, and the phase distortion is < 2 degrees up to 70 kHz. The total power consumption for the system under peak power operation is approximately 30 Watts to produce a measured Sound Pressure Level (SPL) of 181 dB re 1 uP at 1 m. The power consumption

could be decreased by up to 10 Watts by using a single amplifier to drive all 32 transducers, but the array could not be steered electronically.

The array is steerable independently along the short and long axes. The steering is accomplished by delaying the input signal through a network of charge-coupled devices (CCD-s). The CCD-s are controlled by an RCA 1802 microprocessor which can point the array in any direction up to 45 degrees from broadside with a maximum pointing error of 2 degrees. The delay to an array element is a sum of the delays along the short and long axes. For example the delay of the (6,2) element is: 6dX + 2dY where dX and dY are the incremental delays between two adjacent elements along the X and Y axes. The steering circuit consists of two banks of CCD-s connected in series. The first bank computes the dX delays and feeds the signal into the bank which computes the dY delays. The microprocessor controls the CCD input clock rates and multiplexes the CCD outputs to obtain any of the 32 possible values of dX and dY and thus steer the beam to any point in an octant.

Input to the processor consists of four interrupt-like data lines to steer the array up, down, left and right. An active input on any of the data lines steers the array 3 degrees along the respective direction. The input lines contain energy detectors operating at 8.5, 9.5, 10.5 and 11.5 khz so the array may be steered by a remote sonar link. The desired steering angle may also be directly loaded from a serial interface contained in a waterproof box attached to the transmitter by a 100 ft cable. We have encountered problems with both of these steering techniques because the initial orientation of the instrument is often unknown. A magetic compass and a set of inclinometers to provide an absolute reference on the array direction will be a part of any future modification of the steering network.

During shallow water operation, the interface cable is buoyed off at the surface. It is terminated with a watertight box containing a set of switches used to change any transmission parameters or steer the array. This surface control unit was very useful during system testing.

The DATS receiver hydrophone is a simple omnidirectional unit in order to minimize pointing problems which might occur when two directional receivers are both attempting to point towards each other.

The front end of the receiver begins with a battery-powered switchable 20/40 dB voltage gain and low output impedance preamplifier located within 10 m of the receiving hydrophone. The preamplifier is a standard op-amp design which, in addition to boosting the power output of a piezoelectric hydrophone, reduces the EM noise pickup by the hydrophone cable. This cable is often required to be longer than 100 m, and pickup of stray EM transmissions, prevalent around ships or marine worksites is a serious nuisance requiring careful attention to cable shielding

and grounding.

Inside the receiver housing, the preamp is followed by another 20 dB of gain and a 0 to 60 dB automatic gain control (AGC) amplifier. The AGC brings the signal to the several volt range for a wide range of channel distances and geometries. The AGC integration time is chosen longer than 1 sec to make the circuit less sensitive to impulsive disturbances (jackhammers, waves, ship engines, ...) which do not otherwise affect the system performance significantly. More rapid channel variations are compensated by the digital equalizer implemented on the array processor.

The demodulator consists of an analog quadrature demodulator preceded by a bank of filters which separate the data from the synchonization and Doppler pilot tones. The received Doppler pilot tone frequency is multiplied by 5/6, and the resulting 50 kHz (nominally) carrier used to quadrature demodulate the data band onto a complex $\pm$ 5 kHz bandwidth signal. The resulting complex data channel is digitized with 8 bits of precision for further processing. The digitizing rates typically range from 24 to 36 kHz. The data demodulator is shown in Figure 18.

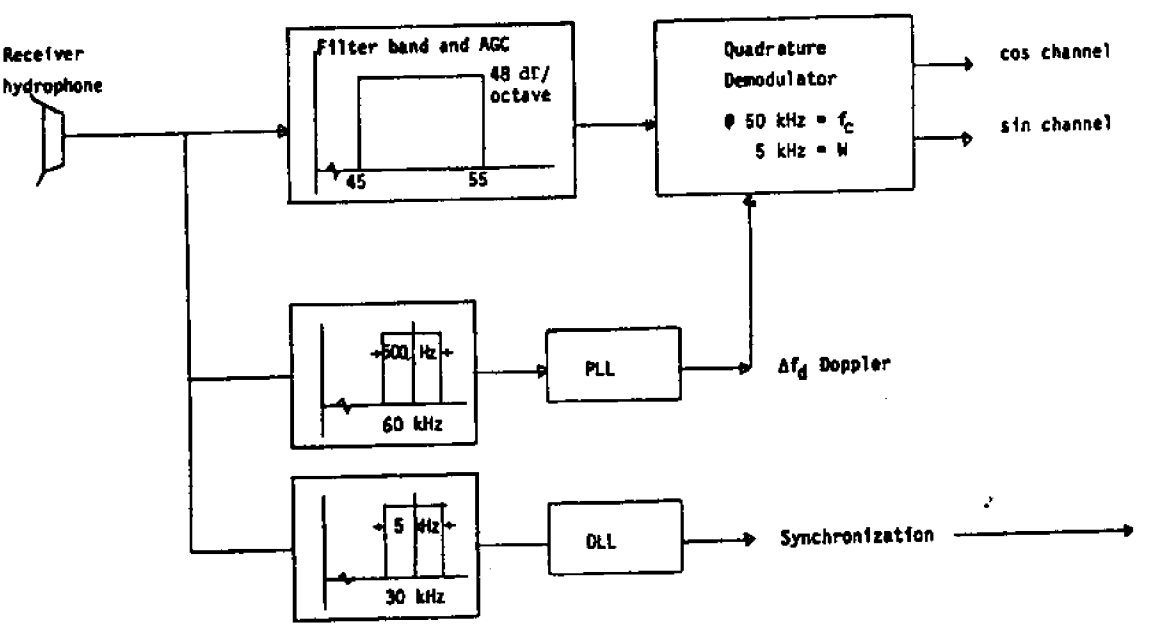


FIGURE 18: DATS Demodulator

The synchronization pulse is received and provided to the decoding computer for a start-to-decode interrupt. Upon receipt of a synch pulse, the processor initiates processing of a new data frame. The sampled data from the modulator is saved in memory, and the data collected during the previous interval is made available to the attached array processor to perform a complex FFT.

The FFT coefficients indicate the amount of energy present at the corresponding frequencies during the last frame. They are input to the decoder, which executes any of a number of available decoding algorithms to attempt to reconstruct the original data sequence. The decoded data is either made available to the end user or is stored on disk along with data and channel behavior descriptions for later performance analysis. The decoder is fast enough to operate in real time for most realistic data rates, and is capable of multi-CPU parallel processing operation to further speed up execution or operate on more complex codes.

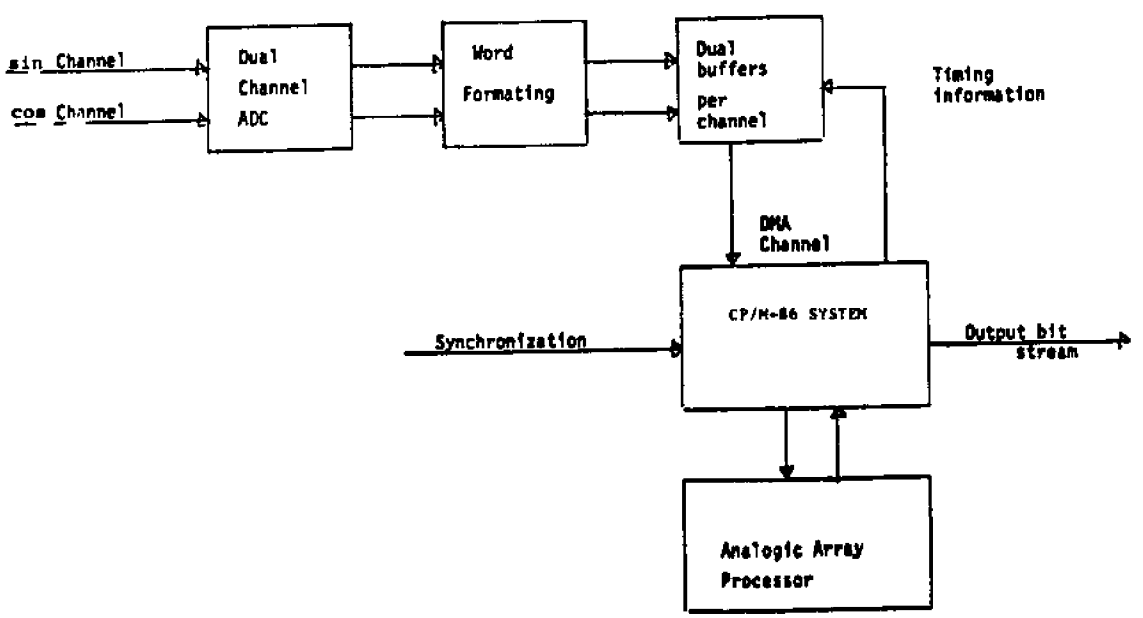


FIGURE 19: DATS Decoder

The processing unit itself is best defined as a loosely-coupled parallel-processing architecture. It is based around an off-the-shelf S-100 CP/M-86 operating system extended to allow for parallel processing. An Analogic AP-400 array processor is available on the system bus for FFT and other numerically intensive computations, and the software operating program allows for coupling other 8086-based slave processors by merely plugging in the slave boards into the S-100 connectors on the bus board. The hardware is diagrammed in Figure 19.

The decoder is interrupt driven. A decoding cycle is initiated on receipt of a synchronization pulse. The main (master) processor responds by initiating the digital data input from the A/D, initializing the FFT processor with the data collected during the previous frame pulse, and, if implemented, sending messages to the slave processors that a task is ready to be processed. The main processor then checks whether all tasks started during the previous frame pulse were completed. If they were, it begins its own processing task; if they weren't, it may either attempt a soft fail by deleting some less important tasks, such as screen output when data is also being saved on disk, from the next frame to be processed, or it may mark the error and stop.

The data are passed to processor memory under control of a programmable DMA transfer device which comprises a single S-100 bus compatible card and is treated by the operating system as an IEEE 696 protocol-compatible temporary bus master. Any such device must be capable of requesting and receiving control of the system bus according to the specifications outlined in Ref [28]. It comprises the priority arbitrating system proposed in [28] and a collection of latches to assert the proper bus master identification and control lines. The detailed requirements and descriptions of those lines are outlined in [28] and are not discussed here. It is worthwhile to note that the transfer card is capable of 16 bit transfers only, as the CPU and the array processor are both 16 bit machines, and as possible expansion of the 8 bit analog to digital converter to 16 bits was anticipated. Presently the least significant bits of the data word are grounded, and the DMA card asserts the bus as a 16 bit device.

The DMA transfer circuitry is centered around an Intel 8237-5 DMA transfer chip which is programmed to transfer the data words (one by one) to main memory as soon as they become available at the ADC following a valid frame interrupt. Although some performance is lost by not utilizing an 8089 I/O processor designed to operate with the 8086, the 8237 allowed a simpler, "off the shelf" design. Also, an 8089 assembler was unavailale at the time. The performance lost with the 8237 is approximately 30 microseconds per frame for the DMA initialization routine plus a harder to measure increase in DMA transfer cycle duration, which is approximately 300 to 400 nanoseconds per cycle.[28] For a typical frame duration of 4 millisec and considering 256

sampled points per frame, this computes to less than 130 more microseconds of bus access time required for the data gathering, or approximately a 3% performance degradation over an optimized 8089 system.

A second generation design dedicates a slave processor to data collecting, and offers up to a 50% throughput increase by transfering the whole data block to the master bus at one time, so that all DMA arbitration is only done once per data block.

During FFT computations, it is necessary to distinguish between the sine and cosine quadratures of the incoming data. The DMA circuit is set up to start the transfers at the first sine quadrature word following a transfer initialization. When ready to accept data, the processor outputs a pulse to the data digitizer. This pulse is used as a conditional along with a "sine quadrature ready" line to enable data ready pulses. A data ready pulse is set by the A/D converter. Its rising edge is recognized by the DMA circuitry and reset by the DMA acknowledge line after the DMA transfer to memory. The first A/D pulse arriving after the last DMA acknowledge cycle sets the data ready line into an inactive mode to await the next processor pulse and the "sine data ready" condition. The DMA circuit transfers a preprogrammed number of words and then waits for reinitialization. It is possible to reinitialize before the preset number of words has been transferred, and the number of words transferred may be read from the DMA chip by the CPU at any time.

The extended address port on the DMA card is used to initialize the upper 3 bits of the 8086 1 megabyte address space, as the DMA chip only drives the lower 17 lines. It is good practice to set the extended address port to the upper three bits of the 8086 data segment register since any "clean" program will transfer all data via the data segment.

Four DIP switches determine the DMA access priority of the board. The S-100 bus allows up to 16 slave processors temporary access to the main system bus. Since such requests must be taken as asynchronous, a device to synchronize them to the bus memory cycle exists on each slave card. In addition, a priority network allows sooner bus access to the higher priority requests Since the DMA board makes frequent (once per sample period) but short (single transfer only) requests and has no data buffering capability, it is desirable to give it a high bus access priority. Otherwise a data byte may be lost while the board waits for the bus to become available. The loss of a single data byte from the DMA card is a serious system error since it disrupts the entire complex data buffer and all subsequent processing of that buffer is impaired.

The other slaves all request the buffer for passing blocks of input and output data or communicating with the main processor. The most essential slave is the system array processor, which does FFT-s and magnitude squared operations on

the received data blocks. It is essentially a convenient implementation of the incoherent correlator detector. The AP operates on the data block placed in memory by the DMA board. It is initialized and loaded with processing routines by the main processor, with which it communicates during execution by a message depositing system. The AP interface was adapted for use with the CP/M-86 system from its original HP RTE-IV interface with which the machine was mistakenly purchased. The design and debugging of the required interface consumed almost one full man-year. The design details and operation method and instructions are given below; they are meant as an extension of [29] which describes the detailed operation of the AP-400 and the HP RTEIV interface.

The Analogic AP-400 array processor is designed for relatively autonomous operation. It consists of a control CPU and a data-processing pipeline. Its only interface to the operator, however is via another computer, which must serve as the software storage/development system as well as a device to load the AP memory and initialize a processing routine. During operation, the AP is generally executing its own "housekeeping" program, except when the host CPU requests it to execute one of the previously loaded programs. The AP responds to the host CPU requests with a series of acknowledging messages and/or interrupts as it executes parts of the program. It may also perform a data transfer to or from the host system.

The AP must initialize the DMA operation; there is no similar mechanism for the host to transfer large amounts of data to the AP. If the host wishes to transfer data, it may either call up the appropriate routine in the AP, (if one is loaded and available), or load the data via direct processor control. The latter technique is essential for the initial AP loading, since the latter contains no permanent memory, and any software used must be reloaded after the power is applied.

To load the AP, or to transfer information to it, the host CPU views the AP as a collection of several programmable registers. The CPU has access to:

AP command register - used to send a software execution command to the AP. The commands are listed in Ref. [2] and need not be repeated here. A command is always first in a string of values sent to the AP.

AP data register - a bidirectional port used to send data to and from the AP. It is primarily used for loading AP program memories or CPU and interface registers.

AP interrupt acknowledge port - provides a convenient method for responding to AP interrupts. A hardware host interrupt response is in some cases too fast for the AP to handle.

AP reset line - used for the complete software and hardware reset of the AP. This line is or-ed with the CP/M-86 system reset so he AP is reset and placed in a predictable state together with the rest of the system. The AP may thus be reset by

a software command.

The extended address port is used to allow the AP, which may address a maximum of 1/4 megabyte of memory directly, to address the full 1 megabyte of 8086 address space. The host uses this port to initialize the upper two bits of memory address. It is convenient to set them to the upper two bits of the data segment, since a "clean" 8086 program will transfer all data through the memory segment allocated to data.

Since the AP must be loaded by an external host processor, any software must be developed and linked on the host machine, and an extensive set of software must be developed to handle the editing, assembling, linking, and downloading the AP-400 code, which, of course, is different from that of a host CPU. The array processor was purchased with the attendant software set up for an Hewlett Packard-RTE IV system, and substantial portions of that code had to be rewritten to execute on the CP/M-86 system. The final software design represents somewhat of a compromise, between development effort and ease of use since all off-line software development and assembly is done on the HP system. The resulting load modules are then downloaded to the CP/M-86 system, where they are linked with any code already prepared for or loaded into the machine to produce a memory map which is downloaded to the array processor. The method is admittedly clumsy but usable since the AP code is seldom rewritten, and any changes may quickly be downloaded to the CP/M-86 and stored in an AP-400 code library. The work is further simplified by a modem program which allows the CP/M-86 computer to emulate a HP RTEIV terminal and greatly simplifies file transfer between the two machines. Thus all work may be done from the same console, and the calling and transfer protocols are largely transparent to the user (or rather, the program author).

While the code is formatted into a load map on the CP/M-86 system, the AP is actually loaded by the code being formatted. This method is used to keep track of the memory requirements of the downloaded code, and facilitates adding an additional module to already set up memory contents. The entire memory map may be stored on the CP/M-86 system disk, and thus most often, only one link file need be downloaded to the AP. Of course, modules may also be combined by the HP linker and the combined load module downloaded to the CP/M-86 machine for conversion into a load memory map.

After the code is loaded into the AP and the latter is initialized, it will attempt to interrupt the CPU to say that it is ready to accept a task. The response to that interrupt and all other host- AP interactive code is lumped into a CPM/86 routine library. The library must be linked with any program which attempts to use the AP. Its contents conform to the documentation laid out in the AP-400 user's manual; however, since that documentation assumes the host CPU is a PDP 11, some variations in parameter passing methods exist, and the listing of the

routine library should be consulted for detailed use. Its detailed documentation is not within the scope of this document.

The invoking of an AP-400 function after it has been loaded together with all the required AP executive and housekeeping code is accomplished through a set of function control blocks (FCB-s). The FCB-s are short memory segments containing sufficient information about the function to be executed and the input and output data. They are formatted in host memory by a particular formatting function. The base of the memory adress is then passed to the AP, which fetches an FCB via DMA, and decodes it to process a function. Upon completing the function, a single-word DMA marks the appropriate spon in the host FCB with a done code, which tells the host whether and how sucessfully the function was executed. The formatting function may be treated as the mirror image of the actual AP processing function. Its task is to reformat the calling parameters from an 8086 convention into that compatble with the AP. A different formatting function is needed for each AP routine, but only a handful were prepared for the DATS code. Writing a new one is not difficult, as these fuctions are only superficially different, and several exist as examples.

The AP should be seen as an independent processor, operating in parallel with the main CPU with only occasional communication between the two. This concept generalizes easily to a set of parallel processors operating on the same bus, and weakly controlled by a bus master. Each processor has an operating system not accessible to the master CPU and an interface for information and control transfer with the master.

The AP is a highly specialized machine and thus requires its own unique instruction set and microcode, but a slave designed to relieve the 8086 CPU of some of the decoding burden may (and should) consist of a CPU identical to the master, so that code as well as data may be transferred between the two. This allows partitioning of the decoding program into several concurrent modules, with each module executing on a different processor. The overall speed increase is significant, since the execution time for an algorithm may be replaced by the duration of access to the master bus for data and control transfer. This scheme is sucessful since relatively small amounts of data need be passed between the modules, and the system bus is not overladed. Bus arbitration is under control of the DMA units (one on each card), and the 8086 processors are not hindered. As long as a block of data is input and output at any time during each frame, the operation is not bottlenecked.

Slave processor boards for the S-100 bus are presently commercially available and are particularly well-suited to this task, which is naturally organized into a set of pipelined steps. A slave processor task is be loaded into its memory, and the master processor initialize its processing for every data block. This effectively removes the CPU processing time constraint from the total decoding time without complicating prohibitively the

required software.

A slave processor designed to emulate the AP interface and communications protocol may be thought of as an "AP" whose processing unit is identical to the master CPU and consists of a minimal 8086 computer system. The computing system may be very simple since its only interface to the outside is via a single port to the master bus controller. The key to such a design is in the communications interface. The slave must be directly controllable by the master, and also capable of sending data back to the main bus. Since the DATS decoder operates on the S-100 bus, such communication has to conform to the IEEE 696 protocol; the interface is based around a programmable DMA chip, the 8237-5. The chip comprises up to four DMA channels, and the design utilizes three: one is available to the processor for transfers to/from the slave bus, another is available to the slave CPU for transfers to/from the master bus, and a third is used for external DMA to the slave bus. (The last channel is particularly useful for buffering DATS input data before passing it to the AP-400.) Thus all arbitration and transfer protocol is concentrated on the already familiar 8237 chip and its peripherals. Note that the 8237 must be programmable by both the master and the slave. This requires an access-arbitrating circuit which gives the chip control to the first requester, and forces the other to wait until the first CPU completes the transfer. The third DMA channel requires very little hardware overhead.

Software for the parallel system is based heavily on the CP/M-86 - AP-400 interface and is designed to resemble it as much as possible. The core of the master CPU supervisory program is the task distributor, which essentially formulates a FCB for each task given to a slave and monitors the execution progress via the FCB-s.

Thus the time required to process a block of data is largely governed by the time for the match filtering, or the FFT computation. Decreasing FFT computations beyond the 2.5 msec required the necessary AP-400 computations (DMA data input, 128-point complex FFT, complex magnitude of the FFT result, DMA data output) to be done by dedicated hardware. This requires, at this writing, quite expensive hardware. This FFT constraint, however, allows one to transmit a large N-ary alphabet (up to 128 tones/block) per frame. A well-designed code of this block size may be capable of data transmission rates and fidelity equaling that of presently-marketed telephone modems.

5 DATA TRANSMISSION RESULTS

Together with the CW transmissions, each harbor experiment included a 6 hour data transmission. The data transmission rate was 400 bits/second. 1200 bits/second experiments were also carried out, but the reduced bit SNR at the higher data rate caused higher error rates and error logging problems. For all data transmission tests, a pseudorandom word sequence was used, so the receiver could compare the received word stream with a stored replica of the transmission to identify transmission errors. It also computed the bit SNR by dividing the energy levels of the eight detected tones by the energy levels in the eight unused frequency bins. The SNR was averaged for 256 data frames, then stored on disk together with the number of transmission errors during the same period.

Before the in-water transmission experiments, the additive white Gaussian noise (AWGN) channel was simulated in the laboratory by summing the transmitted signal with the output of an analog white noise generator This served as a system calibration run and a benchmark for fading channel performance tests. The noise power was varied to span a range of SNR conditions, and the test was conducted for a total of 12 hours. The SNR and error logging procedures were the same as during the water tests and synchronization errors were avoided by directly connecting the transmitter and receiver. It was found that again the synchronization system performed substantially worse than the data decoder at low SNR. However, DATS is not designed to operate over the AWGN channel, and the data is presented because the AWGN is a benchmark and to illustrate the relative performance levels achieved over the two channels.

The in-water data errors consisted of transmission errors, and timing synchronization errors due to errors in tracking the begining of each data chord. The synch. system performed acceptably as long as deep signal fades were infrequent. The two types of errors were detected and tallied separately. Any presented data reflects only the transmission errors, which at times represented but a small part of the overall error rate. The problems with the synch system result from the fading of the pulses. The receiver is implemented with a delay-lock loop which is able to track the pulses across a short (~100 msec) dropout. If the fade is substantialy longer, the receiver begins to search for the pulses, and does not lock again for up to several seconds. As the data transmission during the synch search is garbled, a very large increase in overall error probability results. The present synchronization system is probably inadequate for a strongly fading channel.

To illustrate the difference between the fading and the AWGN channel, Figure 20 shows the predicted and observed DATS

performance over the Rayleigh channel, and Figure 21 shows the same information over an AWGN channel. Both the uncoded and the (8,4) Hamming coded predicted system behavior is shown. On the AWGN channel, both conditional densities are still Chi-square as in the fading case, but the noncentral parameter of the true hypothesis depends on SNR, and the performance improves more quickly with SNR.

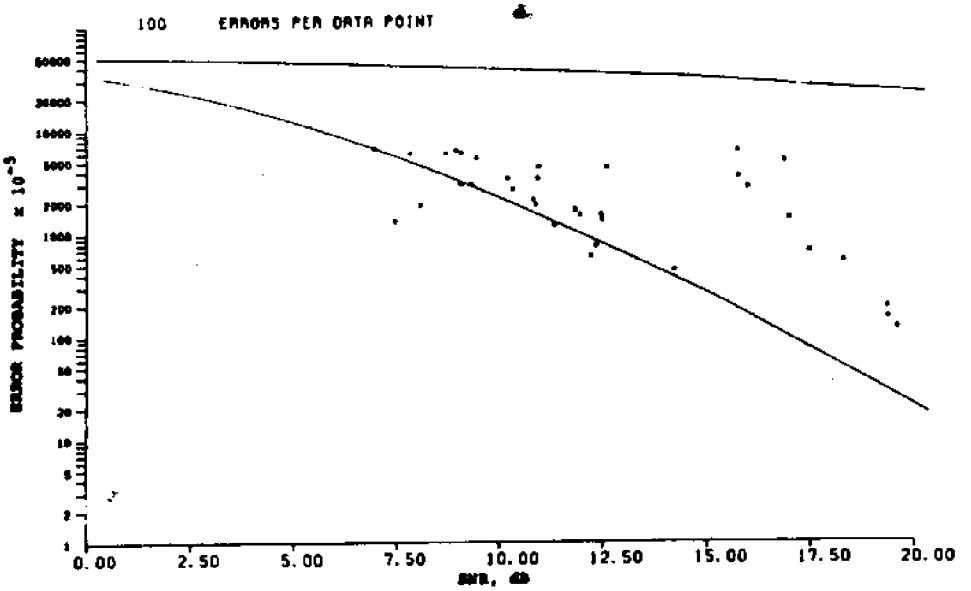


FIGURE 20: DATS Performance - Rayleigh channel

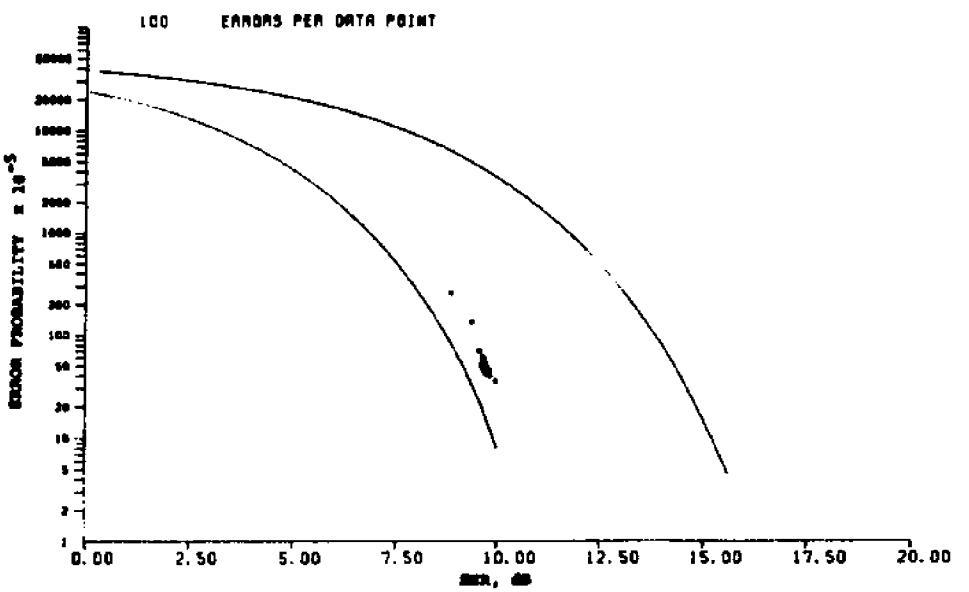


FIGURE 21: DATS Performance - AWGN channel

The receiver performance may be modelled by considering it as a hypothesis test between the tone groups. Consider first a hypothesis test with one tone; the generalization to n tones follows directly. If H1 is true, there is a fading tone at f1, and white noise plus leftover reverberant energy at f2, and if H2 is correct, there is a tone at f2 and noise at f1. The receiver computes the square of the averaged complex envelope in each of the two frequency bins. Then the hypothesis test consists of distinguishing between two Chi-square random variables. If the channel allows a specular propagation path, one of the variables has a noncentral Chi-square distribution. The probability of error for this hypothesis test is given by Turin, [16] for both the coherent and incoherent receiver. The probability of error for an incoherent receiver is given by:

$$\text{Pr. error} = \frac{1}{2} \frac{N_0}{(v_2^2 B(nT) + N_0} \exp\left[\frac{B(nT)a_1^2}{2v_2^2 B(nT) + N_0}\right] \qquad (15)$$

where

is the noncentral parameter discussed in section 2.

Now consider a test with four tones present under each hypothesis, relevant because the code distance is 4. In other words, the two words differ by four tones, so to decide between them the receiver compares the sums of four tone energies. The test consists of comparing the sum of squares of the four tone envelopes, namely:

where a_1

$$\sum_{i=1}^{4} B_i(nT) \gtrless^{?} \sum_{i=1}^{4} B_i(nT) \qquad (16)$$

One may assume that the R-s are independent variables, or equivalently, that the different tones fade independently. In the communication channel encountered, this would be justified if the tone separation is greater than 2 kHz, and the assumption would provide an upper performance bound if the tone separation is narrower. Then the sums are Chi-square random variables with 8 degrees of freedom. The variable corresponding to the true hypothesis has a noncentral distribution.

If there is no intersymbol interference in the time domain, i.e. if there is no "leftover" specular energy from a previous transmission at the frequencies which are not being used for the

current word, the other variable is Chi-square with 8 degrees of freedom. Intersymbol interference may be modelled in two ways. Correlated fading reduces the number of degrees of freedom of the statistic, and reverberation increases the energy level of the "false" hypotheses. Both of these effects tend to overlap the two conditional densities more, and thus increase error probability. Temporally correlated fading would make the error probabilities of successive words dependent, and thus control the "burstiness" of the errors. It would not influence the error probability of a single word.

In all instances the system performance is somewhat worse then predicted. A substantial part of the degradaton is due to the spectral shaping of the transmitted signal by both the transmitting and the receiving equipment, as is evident in Figures 6 and 7. Equalizing the transmitted waveform is difficult because of the frequency hopping and other rapid frequency variations of the transmitted waveform. The transmitter hydrophones, in particular, have a peaked transfer function. The time synchronization pulse, at 30 kHz, and the Doppler pilot tone, at 60 kHz, are also attenuated severely, but the amplitudes are somewhat compensated before transmission.

At the receiver the signal is further shaped by the receiving hydrophone and the front end electronics. This spectral shaping is easily corrected by the digital demodulator, which attempts to whiten the input spectrum, and may be neglected if the equalizer is performing properly. Equalizing a waveform shaped before transmission, however, introduces SNR variation from path to path and thus colors the additive noise. Thus the decoding consists of choosing between sums of Gaussian variables whose means and variances differ. Computing error performance then becomes substantially more difficult, and seems universally avoided in the literature. The uniform path case serves as an upper performance bound.

1 kHz tone separation, as used in the above experiments, is not sufficient to guarantee an independent fading path for each tone. Thus the test statistics have fewer degrees of freedom than considered in the performance prediction, and this, too, is expected to degrade observed performance.

The most significant problem inherent in the high-frequency acoustic communication channel is the paucity of independent propagation paths. A single path for several kHz of carrier frequency in an environment already bandwidth-constrained by attenuation makes for a situation worse than that expected when the system was designed. There have been several attempts at coherent communication which used either the non-fading deep ocean path or concentrated on baffling and system beam patterns for the reduction of secondary returns. The results and data rates attained are superior to those described here, but DATS probably fails more gracefully and is able to operate under more

adverse conditions, particularly those encountered around marine worksites which are typically noisy and cluttered with strong scatterers. Although the DATS performance reasonably reflected the theoretical predictions, the system does not operate with a usefully low error probability. Both the ultimate achieved performance and the level of "tweaking" and system maintenance required to achieve the described performance indicate that increasingly sophisticated coding and a mechanically simpler system need to be employed for potentially useful underwater communications systems.

TABLE 1: FIGURES

REFERENCES

1. Vine, A., "Present and Prospective Needs of Submersibles for Underwater Telecommunications", in Present and Future Civil Uses of Underwater Sound, NAS Book 309-01771-8, 1970

2. Blackington, J. G.and Odegard, M., "An Ocean Bottom Seismograph Using Digital Telemetry and Floating-Point Conversion", IEEE Trans. On Geoscience Electronics, Vol GE-15, No. 2, April, 1977, pp 74 - 82

3. Walsh, G. M., Alair, A. P. and Westneat, A. S., Raytheon Corp, "Establishing Message Reliability and Security in an Underwater Command Link", Proceedings of the Offshore Technology Conference, OTC Paper #1095, 1969

4. Campbell, E.C.: "Helicopter Borne Acoustic Command Control System for Sub Sea Installations" Proceedings of the Offshore Technology Conference OTC Paper #1495, 1977

5. Hearn, P. J.: "Underwater Acoustic Telemetry", IEEE Transactions on Communication Technology, Vol COM-14, No. 6, December 1966

5. Chase, J. V. Jr., "A Tracking and Telemetry System for Severe Multipath Acoustic Channels", in Oceans '81 Conference Proceedings, (Boston, Ma.), Sept 81, pp35 - 39

6. Birdsall, T. G., "Acoustic Telemetry for Ocean Acoustic Tomography" IEEE Journal of Oceanic Engineering, Vol OE - 9, No 4 pp 237 - 241, Oct. 1984

7. Jarvis, F. J., "Description of a Secure Reliable Acoustic System for Use In Offshore Oil Blowout Preventor (BOP) or Wellhead Control", IEEE Journal of Oceanic Engineering, Vol OE-9, No. 4, pp 253 - 258, Oct. 1984

8. Shevenell, M. P. and A. L. Winn, "Acoustic Transmission of Images Using a Microprocessor-Based Video Bandwidth Reduction Technique", IEEE Journal Of Oceanic Engineering, Vol OE - 9, No. 4, pp 259 - 265

9 Szabo, Bernard, MIT Cognitive Information Processing Group, Personal Communication

10. Pieper, J. F., J. A. Proakis, R. R. Reed, and J. K. Wolf, "Design of Efficient Coding and Modulation for a Rayleigh Fading Channel," IEEE Trans. Information Theory, Vol IT-24, No. 4, pp 457 - 468, July 1978

11. Viterbi, A. J., and I. M. Jacobs, "Advances in Coding and Modulation for Noncoherent Channels Affected by Fading, Partial Band and Multiple Access Interference, in Advances in Communication Systems, Vol 4, Academic Press, New York, pp 279 - 308, 1975

12. Mackelburg, G. R., S. J. Watson, and A. Gordon, "Benthic 4800 bits/second Acoustic Telemetry," in Oceans '81 Conference Proceedings (Boston, MA). Sept '81, p.72

13. Rice, S. O., "Mathematical Analysis of Random Noise", Bell System Technical Journal, Vol 23, pp 283 - 332, 1944 and Vol 24, pp 46 - 156, 1945

14. Flatte, S., editor: "Sound Transmission Through a Fluctuating Ocean", Cambridge University Press, 1979

15. Mikhalevsky, P., "Envelope Statistics of Partially Saturated Processes", JASA, Vol 72(1), pp. 151-158, July, 1982

16. Turin, G. L., "Error Probabilities for Binary Symmetric Ideal Reception Through Nonselective Slow Fading and Noise" Proceedings of the IRE, Vol. 46, pp. 1603-1619, 1958

17. Kennedy, R., "Fading Dispersive Communication Channels", Wiley-Interscience, 1969

18. Van Trees, H. L.,"Detection, Estimation, and Modulation Theory", Wiley, 1968

19. Bello, P., "Characterization of Randomly Time-Variant Linear Channels", IEEE Transactions on Communication Systems, Vol. CS-11, pp. 360-393, December 1963

20. Root, W. L., "On the Measurement and Use of Time-Varying Communication Channels", Information and Control, Vol 8, pp. 390-422, 1965

21. Wozencraft, J. and I. Jacobs, "Principles of Communication Engineering", Wiley, 1967

22. Proakis, J. G., "Digital Communications" McGraw-Hill, 1983

23. Flatte, S., "Wave Propagation Through Random Media:

Contributions From Ocean Acoustics", Proceedings of the IEEE , Vol. 71, No. 11, November 1983

24. Gallager, R. L., "Information Theory and Reliable Communication", Wiley, 1968

25. Viterbi, A. J., "Principles of Digital Communication and Coding", McGraw-Hill, 1979

26. Clark, G. C. and J. B. Cain, "Error-Correction Coding for Digital Communications", Plenum, 1981

27. Hanot,W. H., "A Phased Array Sonar for an Underwater Communications System", M.S. Thesis, MIT Department of Ocean Engineering, August 1980

28. Elmquist, K. A. et al, IEEE task 696.1/D2, "Standard Specification for S-100 Bus Interface Devices", IEEE Computer magazine, pp 28 - 52, July 1979

29. Analogic Corporation, Wakefield, Mass. "AP 400 Array Processor Handbook", 1980