IMPLEMENTATION OF VOICEBAND MODEMS ON A DIGITAL SIGNAL PROCESSOR

A. C. Salazar
D. N. Sherman
S. P. Verma
J. J. Werner
Bell Telephone Laboratories, Incorporated
Holmdel, New Jersey 07733

ABSTRACT

This paper presents some considerations on the implementation of data sets on a programmable digital processor. The main points in this paper have resulted from our recent experience in programming voiceband data modulators and demodulators on the Programmable Multiple Data Set (PMDS). As an illustration we discuss the implementation of two 2400 bps data sets which have been successfully implemented on the PMDS.

I. INTRODUCTION

With the decrease of cost of digital circuitry, the use of digital processors for generating and receiving data signals has become more and more attractive. The Bell System New Family of data sets described in Reference 1 makes extensive use of digital processing to provide substantial cost and size savings compared to strictly analog processing methods. Until very recently, it has been customary to provide the digital processing functions with hardwired logic circuitry designed for the specific functions at hand. However, in the experimental Programmable Multiple Data Set (PMDS), all the functions that a data set has to perform are programmable. Using the PMDS, the designer has a great many freedoms in the choice of modulation formats and demodulator structures.

In addition to its programmability, the PMDS has also a multiple data set capability. The time multiplexing of different data sets is done under the surveillance of a microprocessor. The number of data sets that can be accommodated by the PMDS is limited by the computation power of the high-speed processor and the ingenuity of the programmer to reduce the computation time needed for each data set. The present design of the PMDS limits to eight the number of different ports through which data signals can be transmitted or received. Also, on each port, the PMDS is capable of a large number of asynchronous or synchronous modem speeds. Timing control is obtained through the D/A and A/D converter sampling intervals which are under program control.

In this paper we will not be concerned with the multiple data set capability of the PMDS but rather concentrate our attention on the implementation of data sets on the high speed processor. Due to the exclusive use of digital, real-time processing it is often impossible or ill-advised to copy, say, modulation or adaptive equalization schemes which have proved successful for other types of implementations. Our goal has been to design data sets which are compatible with existing data sets, notably the new high-speed family of Western Electric data sets, and which provide a comparable performance at a minimum cost in computation time. The reader is referred to Reference 2 for a description of the high-speed processor of the PMDS. The implementation of different functions such as modulation, demodulation, scrambling, equalization and so on is discussed in Section II. The design of two 2400 bps data sets which have been successfully implemented on the PMDS is studied in Section III.

II. IMPLEMENTATION OF DATA SETS ON THE PMDS

One of the crucial choices that have to be made in the design of a digital signal processor is the choice of the sampling rate, that is, the basic rate of processing of the signals. For voiceband types of signals the highest frequencies of significance that are encountered are equal to about 3 kHz. Therefore, according to the sampling theorem, a sampling rate of 6 kHz is enough to characterize and process linearly all voiceband signals. However such a low sampling rate presents some drawbacks. Nonlinear operations have to be considered carefully because of the risks of aliasing. Many operations are difficult if not impossible to perform at this low sampling rate; for example, zero-crossing detection of differential phase information. Also, low sampling rates impose very stringent requirements on the low-pass analog filter which is necessary at the output of each D/A converter.

On the other hand, a low sampling rate allows the most efficient use of the high-speed processor since the larger the interval of time between samples, the more computations per sample may be obtained. In the design of the PMDS it was decided that the increase of computation power outweighted the annoyance of not being able to perform certain nonlinear operations and the added cost of sharp low-pass filters. Therefore, a variety of relatively low sampling rates has been chosen for the implementation of various synchronous data sets. The required difference of sampling rates between different data sets is due to the necessity of implementing simple programmed counters to derive the baud clocks needed for the transmitter and the receiver. This is easily achieved if the sampling rate is a multiple of the baud. Such a choice is also useful for the implementation of delays since most of the delays used in data sets are simply related to the signaling period.

As far as the programming effort of a data set is concerned, the different operations which have to be performed can be classified as filtering on one hand and the rest of the operations on the other. It has been our experience that filtering is the costliest operation on the PMDS. Up to 80 percent of the computation time allocated to a data set can be used for filtering. On the other hand, filtering uses very few different subroutines since the same subroutines (forming a second-order section for example) can be time multiplexed the required number of times to produce the desired filters.

We have developed several digital filter design programs which make available a wide range of filters.[3,4,5] As a rule we found it convenient to use recursive filters for the receivers and nonrecursive filters for the transmitters. It is well known that recursive filters use less computation time than nonrecursive filters. However nonrecursive filters can exhibit perfect linear phase characteristics and can be more convenient tools to work with than recursive filters in certain applications. Nonrecursive filters can also be used for the implementation of adaptive equalizers. In this case, the filter coefficients are stored in random access memories and are periodically updated.

The updating and filtering are usually done once per baud. The PMDS has a branching capability that allows it to execute different programs in the consecutive computation frames, so that updating and filtering can be done in different frames which occur within the same baud interval. This results in an optimum use of the computational elements of the processor and reduces the computation time needed for an equalizer.

Operations other than filtering, although less expensive in computation time, tend to use a large number of different subroutines. Since the total number of subroutines that can be made available on the PMDS is limited, it is the programmer's interest to reduce the number of subroutines needed for each function. Usually this requires a serious microprogramming effort and an efficient use of the computational elements and the temporary storage is always the key to success. Scrambling, descrambling, serial to parallel and parallel to serial conversions are some of the functions that have to be implemented with micro-code designed for efficiency. Just as is the case for the automatic equalizer, the possibility of processing the required nonfiltering operations at the baud results in a saving of computation time.

III. EXAMPLE OF TWO 2400 BPS DATA SETS

Two medium-speed synchronous data sets have been successfully implemented on the PMDS which operate at 2400 bps and use differently coherent phase-shift keying (DPSK) modulation. Although the two experimental PMDS data sets have significantly different structures, they produce line signals compatible with existing Western Electric 2400 bps data sets and yield about the same error performance. Each data set uses a fraction of the total computation time available on the PMDS. They both operate at sampling rates less than 10 kHz. Both receivers use the same timing algorithm and employ incoherent demodulation. One of the transmitters emulates the transmitter of an existing data set and the other one uses a new modulation structure that has been found attractive for implementation on the PMDS. The flowcharts for the transmitters and receivers are shown in Figures 1 and 2 respectively.

A DPSK signal has the general representation

$$s(t) = \sum_n g(t-nT)\cos(\omega_c t+\phi_n) \qquad (1)$$

where $g(t)$ is a Nyquist pulse, $\frac{\omega_c}{2\pi}$ is the carrier frequency and $\phi_n - \phi_{n-1} = \Delta\phi_n$ is the symbol to be transmitted. For four phase DPSK, the symbol $\Delta\phi_n$ represents one of the four dibits 00, 01, 10 or 11. The mapping from the dibits to the symbols is done by a serial to parallel conversion which requires mostly the use of the Arithmetic Logic Unit (ALU) and temporary registers. We will not go into the details of the functioning of transmitter A. Basically what is done is to compute the phase $(\omega_c t+\phi_n)$, feed it to the sine look-up table and shape the amplitude of the output of the look-up table in order to obtain a well-behaved spectrum. The nonlinear operations that have to be performed introduce aliasing which is nonnegligible at low sampling rates. This transmitter shows a degradation of about 1.5 dB in received signal-to-noise ratio (SNR) for a given error rate when compared to an implementation which does not produce aliasing interference.

Transmitter B generates the line signal in a linear fashion and has a performance which is always within .5 dB of a nonaliasing implementation. There are

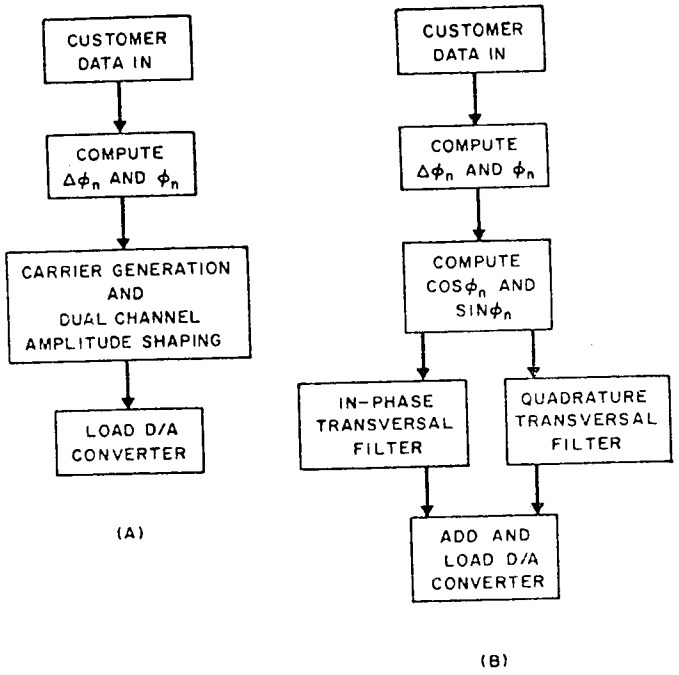

FIGURE 1 — FLOWCHARTS FOR A) TRANSMITTER A B) TRANSMITTER B

several ways of linearly generating a DPSK signal. We found that the least-expensive structure is obtained by rewriting Equation (1) in the following way:

$$s(t) = \sum_n g(t-nT)\cos\omega_c(t-nT)\cos(\phi_n+\omega_c nT) - \sum_n g(t-nT)\sin\omega_c(t-nT)\sin(\phi_n+\omega_c nT) \qquad (2)$$

By inspection of equation (2) we see that $s(t)$ can be obtained by using two bandpass filters with impulse responses $g(t)\cos\omega_c t$ and $g(t)\sin\omega_c t$. The inputs of the two filters are trains of impulses with strengths $\cos(\phi_n+\omega_c nT)$ and $-\sin(\phi_n+\omega_c nT)$ respectively. The implementation of this structure is straightforward. Once per baud $\Delta\phi_n$ is computed and fed to a first order digital filter to yield ϕ_n. A quantity $\omega_c T(= 3\pi$ in our case) is then added to ϕ_n and proper use of the sine look-up table then gives the inputs to the filters. For the bandpass filters we use a transversal filter with a multiplicity of taps. The coefficients of the taps were obtained by starting with the sampled values of the 100 percent excess-bandwidth pulse of the raised-cosine family. These values were then weighted by using different windows and the performances were compared on different telephone lines to determine the optimum set of coefficients.

The performance of both receivers is equivalent to the performance of existing data sets over a large number of lines. For each receiver, the output of the A/D converter is first passed through a recursive digital shaping filter.

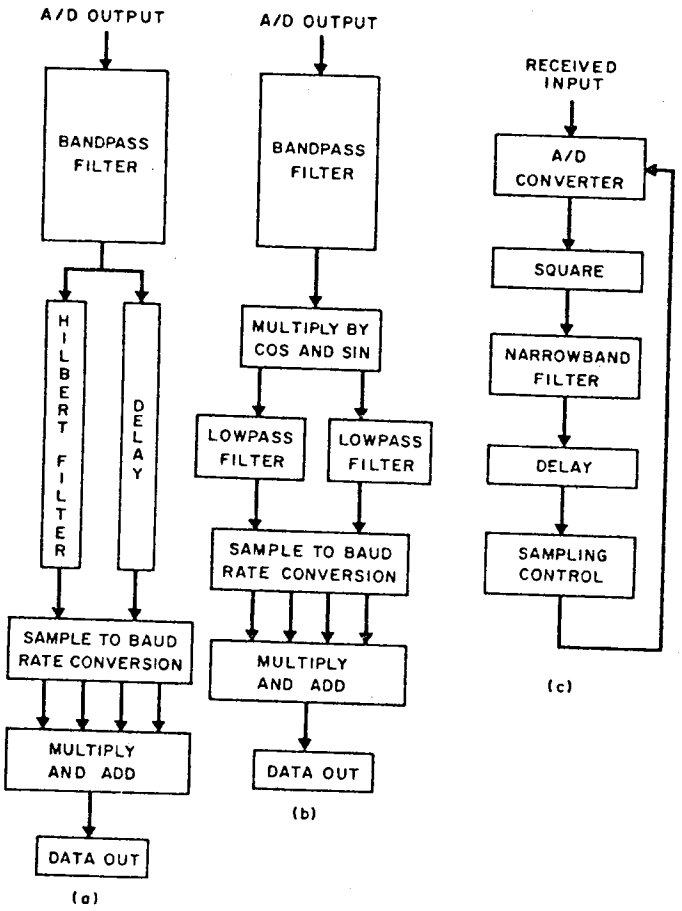

FIGURE 2 — FLOWCHARTS FOR RECEIVERS:
a) DEMODULATOR A, b) DEMODULATOR B,
c) TIMING LOOP

After shaping, each receiver uses a different method of obtaining the in-phase and quadrature components of the transmitted signal. In receiver A, the quadrature component is obtained directly with a 90° degree phase-shift filter. In receiver B, the signal is first multiplied by the in phase and quadrature versions of the derived carrier (phase incoherent) with the frequency shifted result of each multiplication then being low-pass filtered to eliminate the higher frequency sidebands resulting from the multiplications.

The multiply and add routine which recovers the transmitted symbol from the in phase and quadrature components is the same for the two demodulators. We will derive it for receiver A. The received symbol $\Delta\phi_n$ is one of $\pm\frac{\pi}{4}$, $\pm\frac{3\pi}{4}$ and is uniquely defined by the sign of $\cos\Delta\phi_n$ and $\sin\Delta\phi_n$ or, equivalently, by the signs of the real and imaginary parts of the complex number

$$C_n \equiv \cos\Delta\phi_n + j\sin\Delta\phi_n \qquad (3)$$

Consider the analytic signal z(t) defined by

$$z(t) = s(t) + j\tilde{s}(t) \qquad (4)$$

where $\tilde{s}(t)$ is the Hilbert transform of s(t). From Equation 1 it can be shown that z(t) is equal to

$$z(t) = \sum_n e^{j\phi_n} g(t-nT) e^{j\omega_c t} \qquad (5)$$

Multiplying z(t) by its delayed and conjugated version we get

$$z(t)z^*(t-T) = [s(t) + js(t)][s(t-T) - js(t-T)] \qquad (6)$$

$$z(t)z^*(t-T) = \sum_n \sum_m e^{j(\phi_n - \phi_m)} g(t-nT) \cdot g(t-(m+1)T) e^{j\omega_c T} \qquad (7)$$

Sampling at $t = nT$ and using $\omega_c T = 3\pi$, $g(0) = 1$ we get

$$z(nT)z^*[(n-1)T] = -\cos\Delta\phi_n - j\sin\Delta\phi_n = -C_n \qquad (8)$$

A simple change of sign restores C_n which in turn allows us to make a decision about $\Delta\phi_n$. The operations that must be performed are seen by inspection of Equation (6). To obtain the real part of C_n, for example, we have to multiply s(t) by s(t-T) and $\tilde{s}(t)$ by $\tilde{s}(t-T)$, add the two quantities and sample. The order of performance of the operations of multiplying and sampling can be reversed with no penalty to the demodulation process so that all the multiplications can be done at the baud. Therefore, in both receivers, all operations prior to the derivation of the quadrature component are performed at the sampling rate and all subsequent operations are performed at the baud. Note that, in receiver A, the in phase component must be delayed prior to entry into the multiply and add demodulator to account for the delay introduced by the Hilbert filter.

Proper sampling of the digital eye is obtained by synchronizing the sampling clock of the PMDS with the baud clock of the received signal. This is done in a very inexpensive way for the 2400 bps data sets. First the input signal is squared. A tone which is synchronous with the baud clock appears at 1.2 kHz. This tone is recovered by passing the squared signal through a recursive digital bandpass filter centered at 1.2 kHz. The tone is observed once per baud and the timing algorithm locks the sampling time to the 1.2 kHz tone by either advancing or retarding the sampling time of the A/D converter. More sophisticated algorithms have been implemented for data sets with higher speed in order to reduce the jitter in the received clock. However the simple algorithm presented here has proved to be quite satisfactory for use in 2400 bps data sets.

IV. CONCLUSION

In this paper we have only touched some of the problems associated with the implementation of data sets on the PMDS. We think that most of the comments that we have made will remain valid for the

implementation of data sets on other digital signal processors. The success achieved in the programming of synchronous data sets on the PMDS is an encouraging sign for the future of digital signal processing in data communications.

REFERENCES

1. E. R. Kretzmer, "The New Look in Data Communication" Bell Laboratories Record, p. 258-265, October 1973.
2. D. N. Sherman, S. P. Verma, "Programmable Multiple Data Set (PMDS), General System Description," Conference Record, NTC 75, New Orleans.
3. F. Brophy, A. C. Salazar, "Recursive Digital Filter in the Time Domain," IEEE Trans. on Acoustics, Speech and Signal Processing, February 1974.
4. F. Brophy and A. C. Salazar, "Synthesis of Spectrum Shaping Digital Filters of Recursive Design," IEEE Trans. On Circuits and Systems, March 1975.
5. F. Brophy, A. C. Salazar, "Two Design Techniques of Digital Phase Networks," BSTJ April 1975.