MICROPROCESSOR SIMULATION OF SPEECH-ENCODING

by

ROMAN CONTRERAS-DAVILA

This Thesis is submitted in partial fulfil
ment of the requirements for the degree of

MASTER OF PHILOSOPHY

The University of Aston in Birmingham
The Department of Electrical and Electronic Engineering

MARCH, 1982

ACKNOWLEDGMENTS

I am indebted to Dr. Martin H. Ackroyd, my research supervisor, for his constant encouragement, support and advice.

MICROPROCESSOR SIMULATION OF SPEECH-ENCODING

by

Roman Contreras-Davila

A thesis submitted to the
University of Aston in Birmingham
for the degree of
Master of Philosophy
March 1982

Summary

This work describes the implementation of a programmable microprocessor for real time speech processing. Linear and adaptive delta modulation systems for speech encoding and decoding are presented. The systems are based on the Signetics 8X300 microprocessor evaluation board. As this device operates under software control, this feature allows for making as much modifications as required during development at almost no delay or expense.

A PDP-11/03 minicomputer has been used for writing and generating object programs for the 8X300 in a Cross Development System (CDS) arrangement, and a commercial cross-assembler, the MCCAP, was used for program development.

The algorithm used for the adaptive delta modulator was the SONG algorithm[1].

The delta modulators described were tested with speech as the input and also with square and sine waves. Construction details for the system are provided as well as graphical results.

Speech has been band limited to 3.6 KHz and the encoding was performed at sampling rates of 20, 30, 40, 50, 56 and 70 Kbits/s. For further use in subjective tests, an audio tape was generated containing speech as the different sampling rates and using phonetically balanced sentences[2]. Subjective analysis of the encoded speech was performed by two methods, direct comparison and category judgment method. The results of this evaluation are presented.

Software simulating the linear and the adaptive (SONG) delta modulators has been developed and details of these are given.

Salient features of the 8X300 evaluation board are presented, for ready reference.

Twelve-bit resolution was used.

SIGNAL PROCESSING - SPEECH ENCODING - DELTA MODULATION
MICROPROCESSOR SYSTEMS - MICROPROCESSOR DEVELOPMENT
SYSTEMS

CONTENTS

FIGURES AND TABLES

Figures and Tables cont.

CHAPTER I

SPEECH CODING

1.1.-Introduction

This chapter is intended to present a review of the current techniques used to encode voice into digital format. A brief presentation is made for the different methods and techniques trying to underline the main features, the bit rate required to obtain a known quality standard and code efficiency. The main goal for researchers in the area is to be able to encode speech with the highest possible quality and also be able to transmit it over the least possible channel capacity, keeping the overall cost low. Unfortunately, code efficiency and channel utilization are positively correlated with coder complexity, and complexity in turn, is positively correlated with cost. However, recent rapid advances in Large scale integrated circuits (LSI) are dramatically changing the cost/perfomance ratio, particularly with the advent of program controlled devices such as microprocessors and microcomputers. The software aproach in designing speech coder devices permits the designer to include improvements such as making the algorithms more efficient and methods for reducing the data rate further, at almost no additional delay or expense , which could not have been made in a hardwired hardware design without insuring large costs and long lead times. It is in most cases a matter of only modifying the software to incorporate the changes or to try out new systems.

Figure 1.1 shows a spectrum of speech coding transmission rates currently of interest and the associated quality.

Digital coders that achieve telephone toll quali-

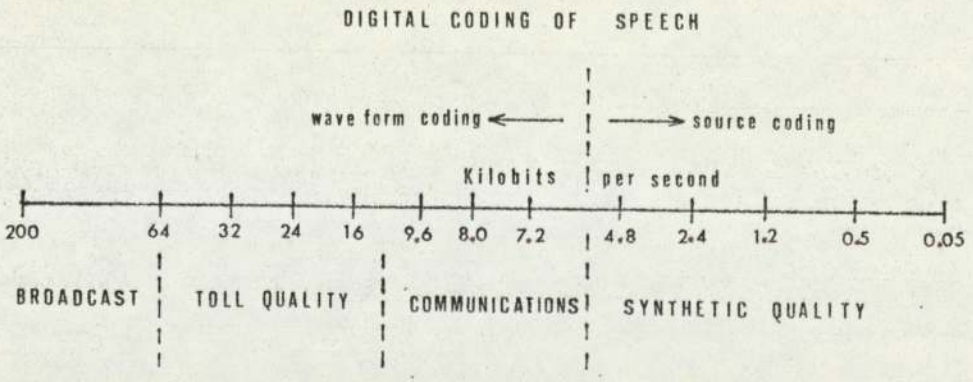


Fig.1.1.-Spectrum of speech coding transmission rates and associated quality.

ty for speech signals at coding rates of 16 kbits/s and above are well known. It has not been the case for coders that operate at rates much below 16 kbits/s. Low-bit-rate speech-communication systems have been available for many years now, but their application areas have always remained extremely specialised. Speech coding at rates within the range 9.6 to 7.2 kbits/s is still intelligible but has some detectable distortion, noticeable quality reduction, and lessened talker recognition (3). Coders which work in the range of 4.8 kbits/s and below provide synthetic quality of speech where the signal usually has lost substantial naturalness. Within the last few years, however, low-bit-rate speech communication systems interest has been stimulated by the demand for digital communications. Efficient data-compression techniques are necessary since the increased bandwidth inherent in digital coding is contrary to the philosophy that bandwidth conservation is requisite.

1.2.-Waveform Coders and Source Coders

1.2.1.-Waveform Coders.-These form a broad class of speech co

ders which essentially strive for facsimile reproduction of the signal waveform. Voice-waveform digitization methods take samples of the speech waveform and represent the sampled waveform amplitude by digital binary-coded values. At the receiving end, the digital signals are converted back to analog form in an attempt to reconstruct the original speech waveform. Waveform coders can code equally well a variety of signals, speech, music, tones, voiceband data, and for the case of speech , they tend to be robust for a wide range of talkers' characteristics and for noisy environment.

Moderate economies in transmission bit rate are achieved for waveform coders of low complexity but can be optimized for greater coding efficiency by tailoring them on the basis of a statistical characterization of speech waveforms.

1.2.2.-Source Coders.-Also known as Vocoders (Voice-Coders) when intended for speech. This class of coders use a priori knowledge about how the signal was generated at the source. The signal is filled into a speech specific mould and parameterized accordingly. Vocoder methods make no attept to preserve the original speech waveform. Instead the input speech is analized in terms of standarized speech features each of which can be transmitted in digital-coded form. At the receiving end these features are reassembled and an output speech signal is synthesized. Ideally, the synthesized signal, as it is perceived by the ear, closely resembles the original speech(4). Vocoders can achieve very high economies in transmission band width, but generally the perfomance is often talker-dependent

and the output speech has a synthetic quality(3). Source coders work in the range of 4.8 kbits/s and below.

1.3.-Waveform Coding

There are several speech properties that can be utilized in an efficient waveform coder design(5). These include, distributions for waveform amplitude and power, the non-flat characteristics of speech spectra, the quasi-periodicity of voiced speech and the presence of silent intervals in the signal. The use of these properties requires different amounts of coder memory, from zero-memory quantizers for amplitude and power distribution up to several seconds of encoding delay for coding that utilize the silent intervals in speech waveforms.

Waveform coding systems may be also based on different strategies for amplitude discretization. In this case the strategies exploit a hierarchy of waveform properties that are best decribed in a formal statistical framework. The probability density function (pdf) of speech amplitudes, the correlations that exist among amplitude samples of a speech waveform, ie, an autocorrelation function (ACF)(long-time and short-time autocorrelation functions), the average probability of different frequency components, ie, the power spectral density (PSD), and the spectral flatness measure (SFM)(long-time and short-time values).

1.3.1.-Time-domain speech waveform coder algorithms

Speech waveform coder algorithms are categorized into time-domain and frequency-domain classes. In some cases, coders within these two classes can be equivalent in terms of

the properties of speech that they exploit.

1.3.1.1.-Pulse Code Modulation (PCM).-A Pulse Code Modulation coder is basically a quantizer of sampled amplitudes of a signal waveform. PCM coders quantize amplitude samples by rounding off each sample value to one of a set of several discrete values and then represent these values by a coded arrangement of several pulses. In a B-bit quantizer, the number of these discrete amplitude levels is 2^B (a 7-bit quantizer implies 128 discrete amplitude levels). As a result of the quantization process, there exists an irremovable error known as quantization error the power of which is proportional to the square of the quantizing step-size. Since the step size is inversely proportional to the total number of levels for a given total amplitude range, a Signal-to-Quantization error ratio SNR can be defined that is proportional to 2^{2B}.

The quantization distortion can be minimized by choosing non-uniform spacing of the amplitude levels to suit the statistical properties of the signal. Non uniform quantization uses the fact that average distributions of speech amplitudes are decreasing functions of amplitude. Fine quantizing steps are used for the frequently occurring low amplitudes in speech, while much coarser quantizing steps take care of the occasional large amplitude excursions in the speech waveform.

There are, in the main, two methods of achieving non-uniform quantization(6). One is to compress the amplitudes of the analogue pulses in a non-linear amplifier (companding) followed by a linear encoder. The second method is that in

which the compression of the signal is carried out as an integral part of the encoding process (non-linear encoding). A compression curve that is reasonible flexible and relatively easy to implement is the logarithmic characteristic. The normalized output and normalized input are related by the logarithmic and linear expressions,

$$y = \frac{1 + \ln Ax}{1 + \ln A} \quad \text{for } 1/A \leqslant x \leqslant 1 \tag{1.1}$$

and a linear expression

$$y = \frac{Ax}{1 + \ln A} \quad \text{for } 0 \leqslant x \leqslant 1/A \tag{1.2}$$

These two expressions are continuos at x = 1/A as seen in Fig. 1.2, and the compression coefficient, A, is a constant chosen to suit the amplitude distribution of the signal. It is customary to approximate the desired compression curve with a multilinear segmental characteristic.

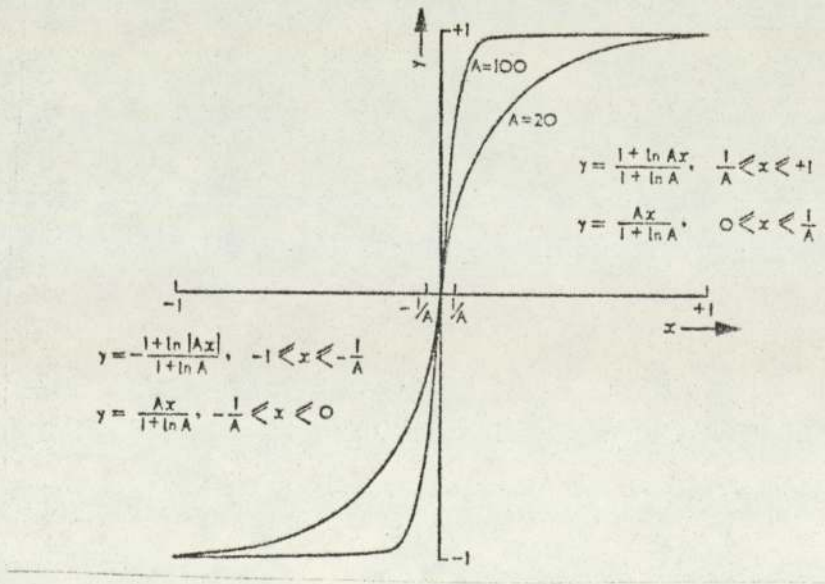


Fig.1.2.-Logarithmic encoding characteristic

1.3.1.2.-Differential Pulse Code Modulation (DPCM).-This coding scheme takes advantage on the redundancies or predictability present in the speech. (adjacent amplitudes in speech waveforms are often highly correlated). In DPCM, speech is represented not in terms of waveform amplitudes, but in terms of the differences between waveform amplitudes. These differences are quantized and then recovered as an approximation of the original speech amplitudes by essentially integrating the quantized difference samples.

Quantization error variance in DPCM tends to be proportional to quantizer input variance, then SNR is inversely proportional to the variance of the coding errors. By reducing the variance at the quantizer input by a factor G, the variance of the coding errors is also reduced by G, and thus the signal-to-noise ratio SNR is increased by a factor G.

Fig. 1.3 represents a practical predictive DPCM. For any number of bits B, the SNR for DPCM shows a gain over that of PCM.

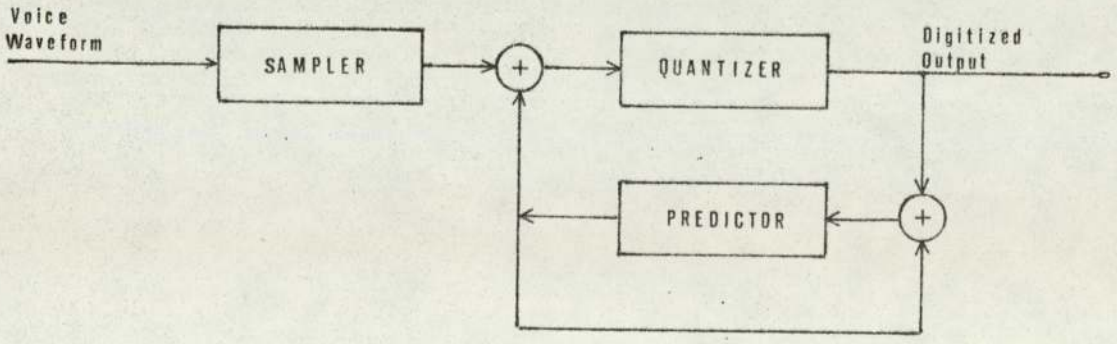


Fig.1.3.-DPCM quantizer

What a DPCM system actually encodes is the difference between a current amplitude sample and a predicted am-

plitude value estimated from past samples (Fig.1.3). This approach represents an step toward the goal of redundancy reduction and implies a significant change compared to that of basic PCM. A DPCM converter that uses one past sample requires approximately one fewer binary bit per sample than an equivalent PCM coder (4), and one employing three past samples can generally eliminate $2\frac{1}{2}$ to 2 bits per sample over PCM. Lower data rates are possible if the weights of the predictor are changed with the speech statistics.

1.3.1.3.-Adaptive Predictive Coding (APC).- Due to the nonstationary nature of speech signals, a fixed predictor cannot predict the signal values efficiently at all times. Adaptive Predictive Coding features a time varying characteristics to cope with the changing spectral envelope of the speech signal as well as with the changing periodicities in voiced speech.

Close observation of speech signals reveals that there are two general causes of redundancy in the voiced speech waveform. The fundamental frequency of the quasiperiodic pulses of air called the pitch of the voice signal (a line spectrum). The air pulses in turn excite the vocal tract establishing resonance conditions. These resonance conditions concentrate the acoustic energy into specific areas of the frequency spectrum known as formants. In terms of the short-term power-frequency spectrum of speech, periodic excitation creates a line spectrum while the vocal tract determines the envelope of this spectrum. Z-transforms can be used to characterize the prediction for both, the prediction method based on short-time spectral envelope and the prediction based on

spectral fine structure.

Fig.1.4 shows the block diagram of an adaptive prediction system and Fig.1.5 shows the block diagram of the predictor of eight order circuit.

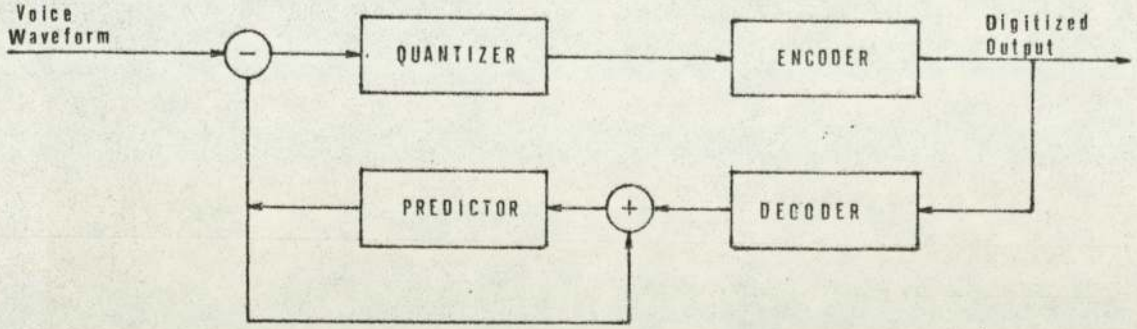


Fig.1.4.-Adaptive prediction system

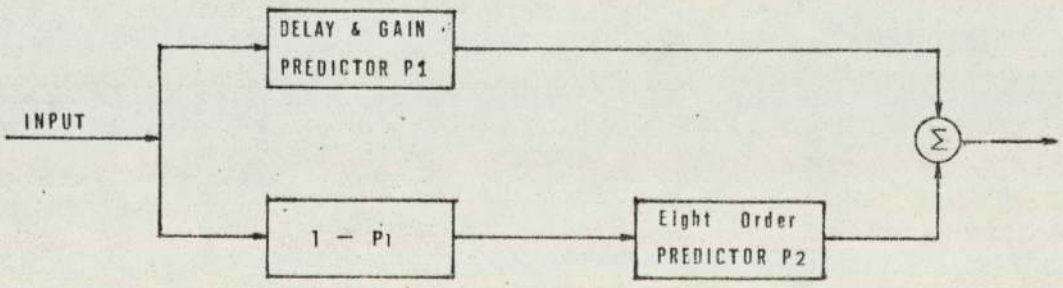


Fig.1.5.-Adaptive predictor. Circuit P_1 predicts quasi-periodic speech components. P_2 predicts short-term voice spectrum structure.

A real-time computer implementation of a fourth-order adaptive predictive coder (APC) has been reported(7). This coder implementation operating at 6400 bits/s is said to produce speech with acceptable voice naturalness even in the presence of acoustic background noise. It has also been referred (4) that computer simulation of APC at a transmission rate of 7.2 kbits/s, yields speech quality equivalent to that of a PCM system operating at 35-40 kbits/s.

1.3.1.4.-Delta Modulation.-A delta modulation encoding system has been chosen for the present work, for this reason an entire chapter has been devoted to this encoding technique. For the sake of continuity some lines will be written here about delta modulation.

Delta modulation is an special case of DPCM with one bit quantizer. In delta modulation systems the quantizer and predictor have been reduced to a very simple form. The predictor in the feedback loop consist of a gain circuit and an integrator.

Various methods of gain control have given rise to different forms of delta modulation. For instance, Continuous Variable Slope Delta (CVSD) modulation adapts to vary the gain A over a continuous range. Variable Slope Delta (VSD) modulation adapts on the error signal, changing the gain in steps of 2^{a}. Digital Controlled Delta (DCD) modulation uses a digital comparator operating on the error sequence to control the gain function.

Because delta modulation technique uses only a 1-bit quantizer, its implementation can be straight forward and inexpensive. This simplicity, combined with good voice quality, has generated considerable interest in using delta modulation for commercial telephone systems, military communication systems,spatial vehicles communication systems, as well as for using it in other areas.

Other coders such as Delayed encoding, Aperture coding and Gradient-Search coding are more sophisticated versions of DPCM and DM.

1.3.2.-Frequency-domain speech waveform coder algorithms

In the time domain coders described in the previous pages, speech is treated as a single full-band signal.

In the frequency domain coders, the approach is to divide the speech signal into a number of separate frequency components and to encode each of these components separately. The number of bits used to encode each frequency component can be varied dynamically and shared with other bands, so that the encoding accuracy is placed where it is needed in the frequency domain. In fact, bands with little or no energy may not be encoded at all.

A large variety of frequency domain algorithms, from simple to complex are available in the frequency domain technique and the main differences are usually determined by the degree of prediction that is employed in the technique.

1.3.2.1.-Sub-Band Coding (SBC)(8)(9).-In the sub-band coder the speech band is subdivided into typically four to eight sub-bands by a bank of band pass filters. Each sub-band is then low pass translated to zero frequency by a modulation process. It is then sampled at its Nyquist rate and digitally encoded with an adaptive-step-size PCM (APCM) encoder. On reconstruction, the sub-bands signals are decoded and modulated back to their original location. They are then summed to give a close replica of the original speech signal.

In the sub-band coding process, each sub-band can be encoded according to perceptual criteria that are specific to that band. Separate adaptive quantizer step-sizes can be used in each band. Bands with lower signal energy will have

lower quantizer step-sizes and therefore contribute less quantization noise. The shape of the quantization noise can also be controlled by properly allocating the bits in different bands. In the lower frequency bands, where pitch and formant structure must be accurately preserved, a larger number of bits/sample can be used, whereas in upper frequency bands, where fricative and noise like sounds occurs in speech, fewer bits/sample can be used.

Fig.1.6 illustrates a basic block diagram of the sub-band coder. Newer filter technologies such as Charge Coupled Device (CCD) filters and digital filters provide marked advantages in the digital implementation of this coder.

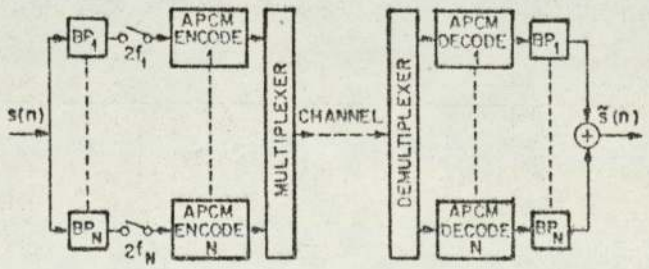


Fig.1.6.-Block diagram of a sub-band coder

The perceptual advantages of sub-band coding are put in evidence by the perceptual data of Fig.1.7 (3). Fig.1.7a shows the relative preference of a 16 kbits/s sub-band coder versus that of an ADPCM coder at various ADPCM coder bit rates. Fig.1.7b shows similar results for a 9.6 kbits/s sub-band coder compared against an ADM coder at various ADM bit rates.

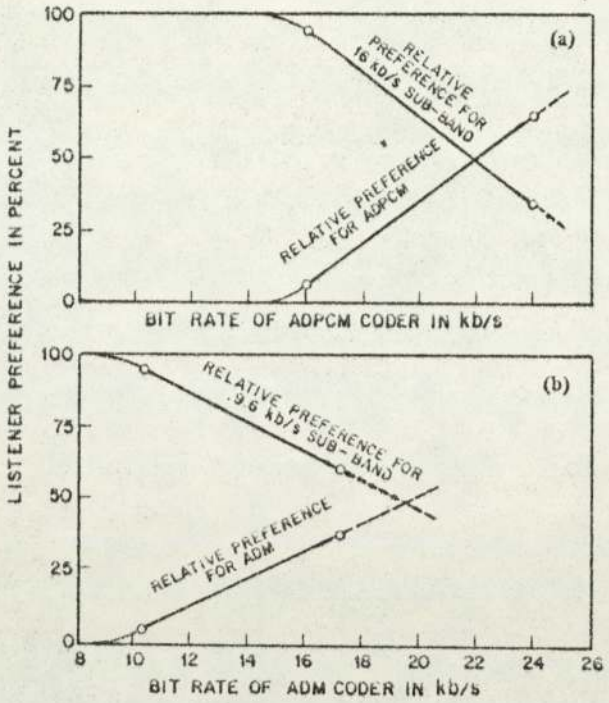


Fig.1.7.-(a)Relative comparison of quality of 16-kbit/s sub-band coding against ADPCM coding. (b)Relative comparison of quality of 9.6-kbit/s sub-band coding against ADM (From ref. 3).

1.3.2.2.-Adaptive Transform Coding (ATC)(10)(11)(12)(13).-

Transform coding involves block transformation of windowed input segments of the speech waveform. Each segment is represented by a set of transform coefficients, which are separately quantized and transmitted. At the receiver, the quantized coefficients are inverse transformed to produce a replica of the original input segments. Succesive segments, when joined, represent the input speech signal.

Fig.1.7 illustrates a basic block diagram of the algorithm. The input speech signal is blocked into frames of data (typically N = 128 to N = 256 samples long at 8 KHz) and transformed by an appropiate fast-transform algorithm. The transform coefficients X(k), k = 0, 1, 2, ..., N-1 are then

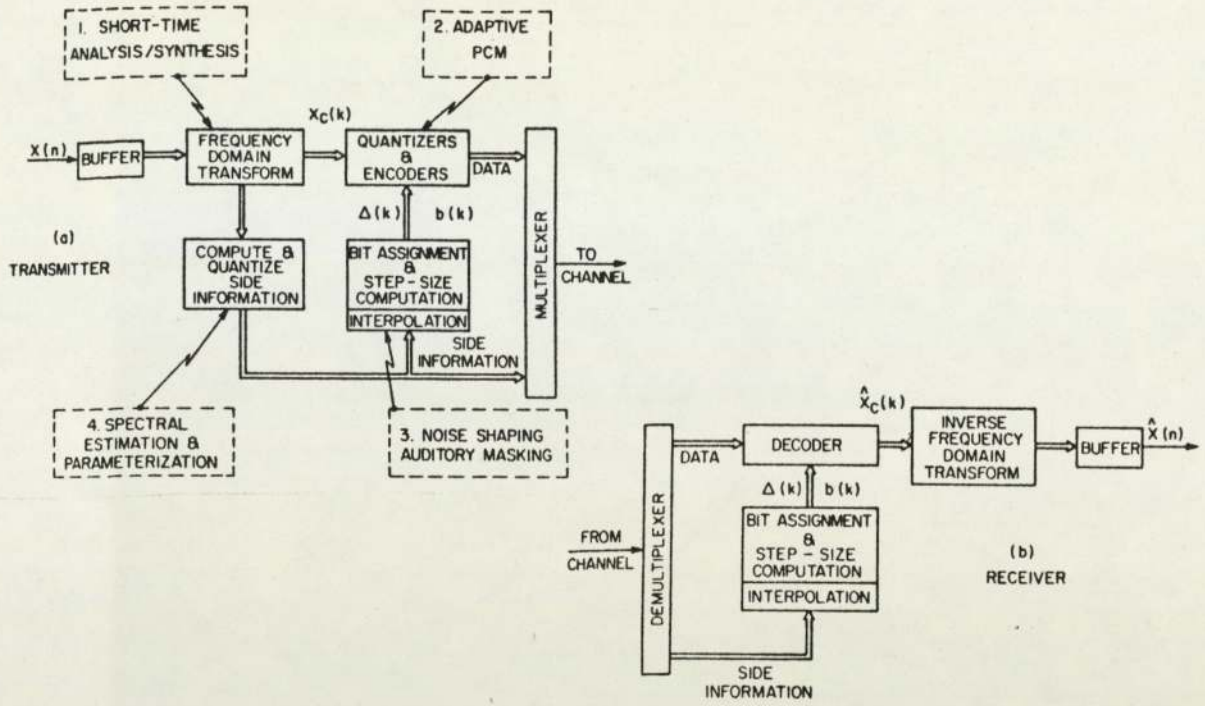


Fig.1.7.-Block diagram of adaptive transform coding

adaptively quantized by a set of PCM quantizers whose step-sizes (k) and number of bits b(k) are dynamically varied from block to block to take advantage of known spectral properties of speech production and perception.

In the quantization strategy, the transform coefficients are usually quantized individually using a uniform step size. The choice of the step size and the number of bits for a given transform coefficient is of fundamental importance and it is determined through the aid of a separate "side information" channel which models and parameterizes the local spectral characteristics of speech in each block. Ideally, the step-size must be large enough so that overloading of the quantizer does not occur. The "side-information" is also quantized and transmitted to the receiver for use in decoding.

Two basic adaptation techniques for ATC of speech

heve been proposed. A technique referred to as 'Non-speech specific'(10)(11), since it does not take into account the dynamical properties of speech production and a more complex adaptation technique referred to as 'speech specific'(12)(13). The first adaptation technique is appropiate for speech transmission at or above 16 kbit/s and the second one is more appropiate for lower than 16 kbit/s bit rates.(9.6-16 kbits/s).

1.4.-Source Coders, Vocoders.-

Speech signals are known to be highly redundant. The most effective method to reduce this redundancy and thus reduce the channel capacity required for transmission of speech signals is to extract and transmit only the major characteristics of speech at a regular interval (typically 10 to 30 ms).. Unlike Dm, PCM, DPCM and APC techniques, vocoders do not try to preserve the original voice waveform. Instead, the goal is to preserve the perceptually significant properties of the waveform. By analysing the input waveform on its short-term spectra, most vocoders compute parameters that describe a simplified model of the speech-production mechanism. The traditional model is that shown in Fig.1.9.

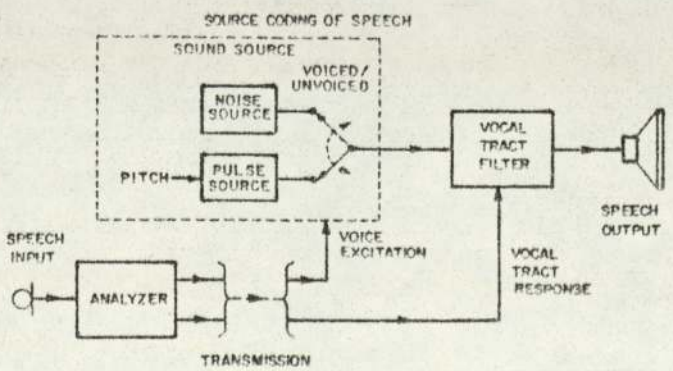


Fig.1.9.-Speech generation source-syetem model.

The source for voice sounds is represented by a periodic pulse generator, and the source for unvoiced sounds is represented by a random noise generator. The sources are normally considered mutually exclusive with a parametric signal indicating switching between voiced and unvoiced sources. The intensity of sound excitation for each source is also represented parametrically by an amplitude or gain signal. In addition the periodicity or pitch of the voiced pulse source must be specified by a parametric pitch signal.

In general, the analyzer section of a vocoder determines the resonant structure of the vocal tract, estimates the pitch, and decides whether the speech segment is voiced or unvoiced. The synthesizer section uses these speech features to reconstruct a new time waveform that sounds much like the input. Various vocoders differ in their methods for extracting speech features as well as their methods for reconstructing speech using these features.

1.4.1.-Frequency-domain Vocoders

As in waveform coders, vocoders have also a broad division of frequency-domain and time-domain vocoders.

1.4.1.1.-Channel Vocoder.-The oldest method for speech analysis-synthesis employing a parametric description of the short-time speech spectrum is the spectrum-channel vocoder(14). In it, the spectral envelope is represented typically by 10 to 20 samples spaced along the frequency axis. The spectral fine structure is represented by one additional parameter which measures the fundamental frequency f_o of voiced sounds and is equal to zero for unvoiced sounds or silence.

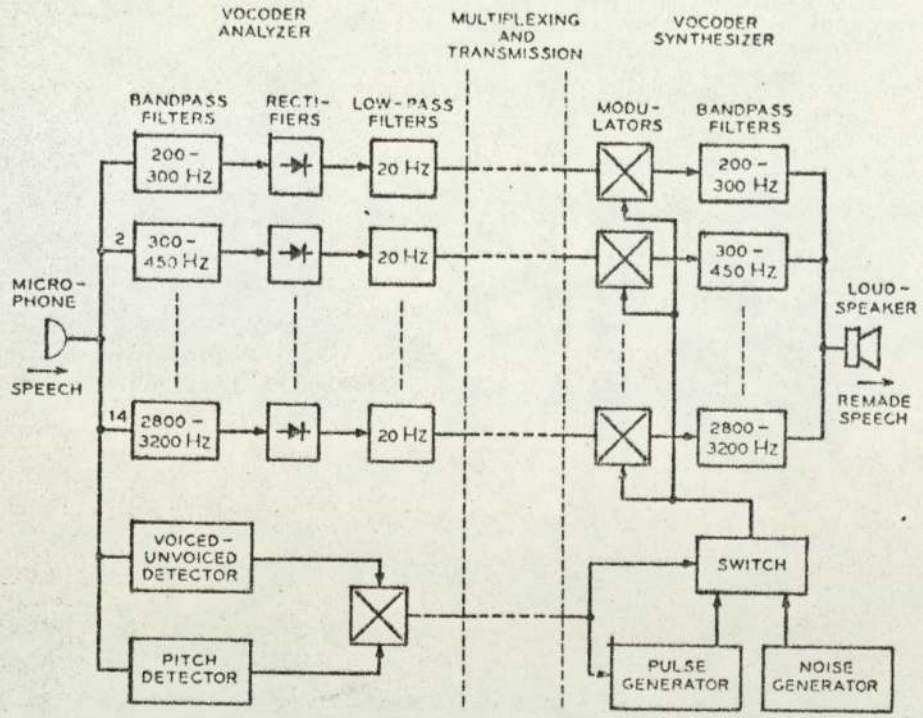


Fig.1.1.10.-Block diagram of channel vocoder.

A block diagram of a channel vocoder is shown in Fig.1.10. The speech signal is separated into 14 contiuous spectral bands with bandwidths between 100 and 400 Hz covering the frequencies from 200 Hz to 3200 Hz. The output of each filter is connected to a rectifier and low pass filter whose output represents the time-varying average signal amplitude for each frequency band . Together, these 14 channel signals represent the envelope of the short-time spectrum of the speech signal.

Also shown in Fig.1.10 are a voiced-unvoiced detector and pitch detector which determine the fine structure of the speech signal and produce a corresponding narrow-band signal. These 15 narrow-band signals are combined into a single signal with a total bandwidth of 15x20 Hz = 300 Hz. Thus,

the transmission bandwidth is only one-tenth of that required for the original speech signal.

At the synthesizer, the original channel signals are recovered and utilized to control the frequency response of a time-varying filter (modulators and narrow band-pass filters) to correspond to the spectral envelope measured at the analyzer. The input of this time-varying filter is supplied with a flat spectrum excitation signal of the proper spectral fine structure, quasi-periodic pulses for voiced speech sounds or white noise for unvoiced sounds.

At present, channel vocoder is a established system for low-bit-rate (2400 bit/s) speech compression. Recent advances in analogue signal processing may produce a channel vocoder attractive in terms of engineering premiums. One technology which offers high-density low-power analogue signal processing is the charge coupled device (CCD)(15).

A recent paper(16), reports a very low rate channel vocoder. It is the authors opinion that at the lowest rate value of 1200 bits/s, although degradation is readily apparent, the quality and intelligibility would be adequate for many applications.

1.4.1.2.-Formant Vocoder.-The formant vocoder is similar in concept to the channel vocoder. The major advantage of a formant vocoder is that, unlike other vocoders, intelligible speech quality can be obtained at a transmission rate as low as 800 bits/s. This is possible because fewer transmission parameters are required than for other vocoders.Instead of sending samples of the power spectrum envelope the formant voco-

der attempts to transmit the positions of the peaks of the spectral envelope. These peaks positions are called formant frequencies and correspond to the poles in the transmission response of the vocal cavity.

The formant vocoder is ideally suited for transmission of speech over an adverse channel where only low bit rates can be supported, or in a situation where bandwidth compression is of prime importance. The price for the low bit rate is of some degradation of speech quality. However, no other vocoder is known to produce better speech quality than the formant vocoder at a transmission rate below 1800 bits/s .

The hardware for vocoders has normally been complex, bulky, and expensive. However, recent advances in Large-scale integrated circuits (LSI) and in microprocessor technology have made it possible to build much smaller hardware at a cost that is competitive with other speech digitizers. A paper concerned with an all-digital formant vocoder system(17) reports a reasonable good speech quality at 1200 bits/s, in which most of the utterances are intelligible and speaker-recognizable.

1.4.2.-Time-domain vocoders

1.4.2.1.-Autocorrelation and cross-correlation vocoders(18)(14)

A speech sound can be characterized by a time-varying short-time autocorrelation function instead of its short-time spectrum.

A block diagram of an autocorrelation vocoder is shown in Fig.1.11 In the analyzer section, an equalized speech signal is multiplied by itself delayed by various amounts τ_o,

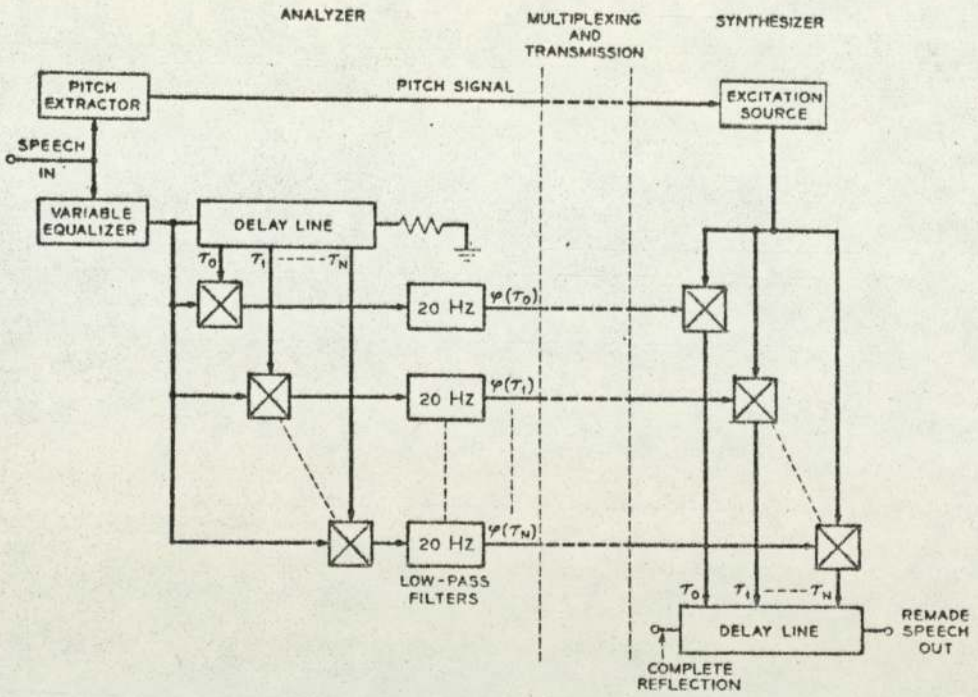


Fig.1.11.-Block diagram of autocorrelation vocoder.

τ_1, ..., τ_n with a delay increment smaller than half the reciprocal bandwidth. The products are low-pass filtered to about 20 Hz to form the respective channel signals. The time-varying channel signals represent a short-time autocorrelation function and can be used to synthesize speech in the time domain. The time-domain synthesizer of the autocorrelation vocoder consists of a time-varying symmetric transversal filter whose implulse response is a replica of the short-time autocorrelation function for delaus between $-\tau_n$ and $+\tau_n$. The excitation signal applied to the synthesizer has a flat spectrum envelope and the proper fine-structure (quasi-periodic pulses for voiced speech sounds and white noise for unvoiced sounds).

Autocorrelation vocoders produce output signals whose spectrum is the square of the spectrum of the input signal. While spectrum squaring does not destroy the intelligi-

bility of speech, it does give rise to an unnatural quality. Also all level differences in the original spectrum are doubled. To overcome the adverse effects of spectrum squaring, autocorrelation vocoders require special equalizers which perform a spectrum square-rooting operation.

The spectrum squaring inherent in autocorrelation analysis can also be avoided by cross-correlation analysis. In this, samples of the cross-correlation function between the speech signal and a 'spectrally-flattened' coherent signal are transmitted. In cross-correlation vocoders as well as in aotocorrelation ones, the number of samples, and thus the total transmission bandwidth, are higher than in channel vocoders of comparable output speech quality. Also, for digital transmission, the number of bits per correlation sample needs to be about twice as high as for spectral samples (7-8 bits/sample instead of 3-4 bits/sample for good speech fidelity)(3).

1.4.2.2.-Orthogonal expansion Vocoders and the Cepstrum(3)(14).

In principle, speech signals or their spectra can be expanded according to any of a large number of orthogonal and complete system of functions. The choice depends on the desired speech quality, available transmission facilities, and on the state of the instrumentation art. For waveform expansion, the Laguerre polynomials are particularly suited because of the weighting function e^{-t} in conformity with the time envelope of a speech formant. For the power spectrum, the eigenfunctions of the autocovariance matrix are an optimum set of expansion functions for minimum r.m.s. error under quantization. Such expansions are called Karhunen-Loeve (KL)

expansions.

Instead of expanding the power spectrum, its logarithm can be expanded into a cosine series. The sequence of expansion coefficients is called 'cepstrum' and a vocoder based on the cepstral coefficients is called a cepstral or homomorphic vocoder. The advantage of cepstral vocoders is that the cepstrum is often available already for measuring voice pitch and no additional analysis is needed.

1.4.2.3.-Linear Prediction Vocoder(19)(20)(21)

The linear Predictive coding (LPC) approach is an analysis-synthesis method of vocoding in which the excitation (pitch) and the vocal tract modelling are treated separately. In the LPC approach the vocal tract is modelled by a time-invariant, all pole recursive digital filter over a short time segment (typically 10 to 30 ms). The time-variant character of the speech is handled by a succession of such filters with different parameters. The excitation is modelled either as a series of pitch pulses (voiced) or as white noise (unvoiced).

Linear predictive systems differ from the adaptive-predictive coding scheme discused earlier in that, in the adaptive schemes, it is the error signal that is transmitted. while in linear predictive systems only selected characteristics of the error signal are transmitted. These parameters include a gain factor, pitch information, and voiced-unvoiced decision information.

There are two types of linear prediction vocoders, pitch-excited(22)(23) and residual-excited(24)(25)(26). The major difference between these two types lies in how the exci

tation signal for the synthesizing filter is characterized. In a pitch-excited LPC vocoder the vocal tract, glottal flow, and radiation are represented by the prediction coefficients. Those coefficients are transmitted together with the information regarding excitation of speech, that is, the fundamental frequency or pitch , the voiced/unvoiced decision, and a gain extracted from either the residual signal or the speech input.

In the residual-excited LPC vocoder the vocal tract is characterized in the same way as in the pitch-excited one. However, instead of the excitation feature properties (pitch, voiced/unvoiced decision, and gain) being extracted and transmitted, the LPC inverse-filtered residual is low-pass filtered, encoded for transmission, and used as the excitation source at the receiver. Typical examples of this vocoder type are adaptive differential pulse code modulation (ADPCM) system with multitaps (24) and the adaptive predictive coder (APC) (23).

1.4.2.4.-Related Vocoder Devices

One of the more difficult tasks in speech analysis is the pitch detection, ie, the extraction of the fundamental frequency from a running speech signal. The basic fault lies in forcing the speech signal into the simplistic model of voiced and unvoiced sounds. There are a number of vocoder devices where the objective is to settle for less bandwidth conservation to avoid the problem and complexity of pitch detection. The earlier examples of these are voice-excited vocoders (VEV) (27)(14) and phase vocoders (ØV)(28)(29). Hybrid arrangements of sub-band coding (SBC), adaptive predictive coding (APC)

and linear predictive coding (LPC) are coming into use, where a portion of the coding is accomplished by waveform techniques and a portion, mainly the upper frequency bands, by voice-excited vocoder techniques. A more recently developed approach(30) for improving the perfomance of waveform coders is based on coding a frequency scaled speech signal. It is the time-domain harmonic scaling (TDHS) algorithm. Sub-band coding combined with TDHS (SBC/HS) at 9.6 kbits/s has been found to provide a quality equivalent to that of SBC alone at 16 kbits/s, ie., a bit rate advantage of about 7 kbits/s. For the speech specific adaptive transform coder (ATC) used, the combined system (ATC /HS) is said to have achieved a bit rate advantage of 4 kbits /s at 7.2 kbits/s.

1.5.-Some comments about Complexity and Quality of coders

The following comments related to complexity and quality of speech coders have been drawn from reference (3). These are based on the authors' experience since by the time of the publication there were not comprehensive, quantitative data on the subject. It is the belief of the author of the present work that there is still not any to date.

The coders range widely in complexity. Table 1.1 shows a relative comparison about complexity which has been made by estimating the complexity of several coders relative to a simple adaptive delta modulator.

The values of relative complexity quoted in the Table 1.1 are very approximate, and depend upon circuit architecture.

Table 1.1.-Relative complexity of speech coders

Relative Complexity	Coder	
1	ADM:	adaptive delta modulator
1	ADPCM:	adaptive differential PCM
5	SUB-BAND:	sub-band coder (with ccd filters)
5	P-P ADPCM:	pitch-predictive ADPCM
50	APC:	adaptive predictive coder
50	ATC:	adaptive transform coder
50	ØV:	phase vocoder
50	VEV:	voice-excited vocoder
100	LPC:	linear predictive coefficient vocoder
100	CV:	channel vocoder
200	ORTHOG:	LPC vocoder with orthogonalized coeff.
500	FORMANT:	formant vocoder

As far as quality is concerned, the authors'[3] experience suggests that toll quality coding of speech can be obtained with the following coders running at, or above, the transmission bit rates indicated in Table 1.2.

Table 1.2.-Toll-Quality Transmission

Coder	kbits/s
Log PCM:	56
ADM:	40
ADPCM:	32
SUB-BAND:	24
Pitch predictive ADPCM:	24
APC, ATC, ØV, VEV:	16

Similarly, communications quality can be achieved by the combination of coders and minimal bit rates shown in Table 1.3.-

Table 1.3.-Communications-Quality Transmission

Coder	kbits/s
Log PCM:	36
ADM:	24
ADPCM:	16
SUB-BAND:	9.6
APC, ATC, ØV, VEV:	7.2

CHAPTER II

THE MICROPROCESSOR DEVELOPMENT SYSTEM (PDP-11/03 - 8X300)

2.1.-Introduction

This chapter describes the utilization of a minicomputer, a PDP-11/03, in a microprocessor application develoment system. The system is intended to aid the production and testing of the 8X300 microprocessor software. This kind of arrangement is known as a Cross-Development System (CDS) (31) to differentiate from dedicated Microprocessor Development Systems (MDS).

It is well known that microprocessors and other forms of programmable microelectronics have caused a breakthrough in cost/performence ratios. This reason and the flexibility offered by these software controlled devices have caused a widespread use in instrumentation, industrial control, computer peripherals and also in data processing. Development and testing of microprocessor software is without any doubt, becoming increasingly important, and has been recognized (32) as being the largest single problem in microprocessor development systems (MDS).

2.2.-Minicomputer System Components

The availability of a minicomputer makes it possible to build up a powerful development workshop that can be used in different microcomputers.

A basic system may comprise a DEC PDP-11 minicomputer with 32 kbytes of memory, dual floppy disks with 512 kbytes storage, a console terminal device and input-output ports. This basic system can be enhanced by the addition of a range of peripherals such as v.d.u.'s and faster printers. Fig. 2.1.

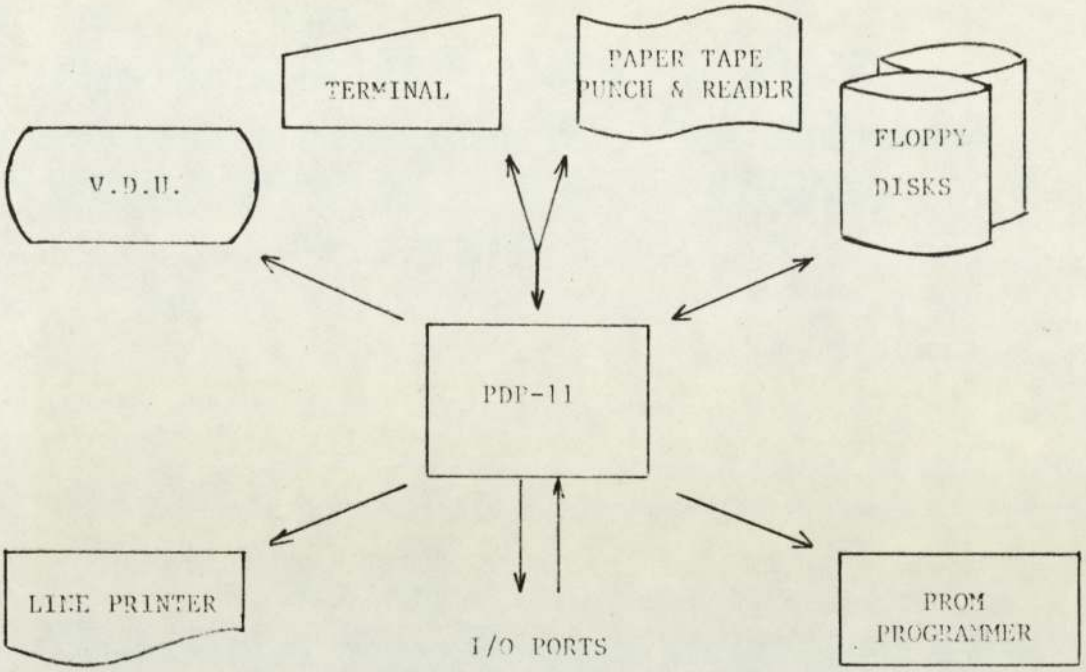


Fig.2.1.-The basic PDP-11 minicomputer system.

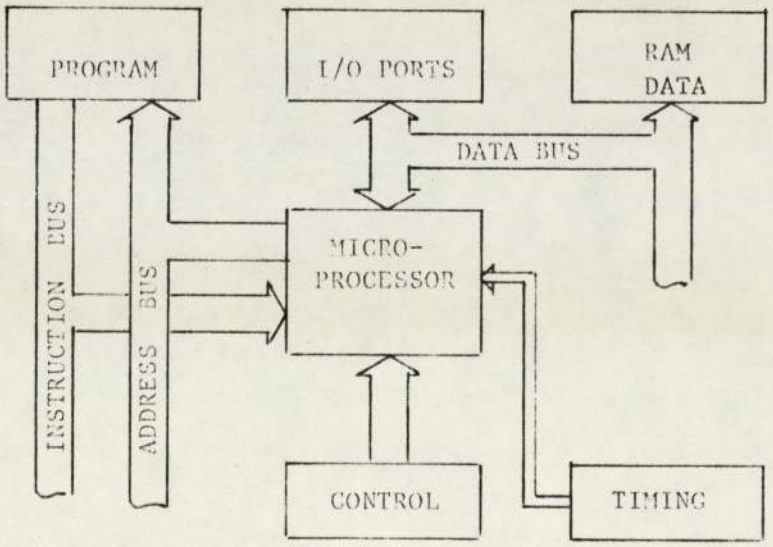


Fig.2.2.-Target microcomputer board.

2.3.-Microcomputer System Components

The typical features provided by microcomputer evaluation boards are shown in Fig. 2.2.

A computer is made up of three basic blocks. The heart of the computer is the C.P.U., capable of performing arithmetic, logic functions, data manipulation, etc. Associated with the C.P.U. there must be some data storage facility to store data that is being worked on and the results, and the instructions which control the operation of the C.P.U. The third block in the computer system is the input/output (I/O) function.

The basic computer configuration is also used in system design using a microprocessor device. When a microprocessor is used to form the C.P.U. function in a system, the system is referred to as a microcomputer.

To perform any useful task, the microcomputer must interact with the outside world. The input/output (I/O) devices provide the necessary data comunications link between the microprocessor and its environment.

The microprocessor communicates with the input-output devices and other parts of the microcomputer by sending information along groups of signal lines called buses. It is common for more than one type of information to share the same bus. The need for introducing a shared bus system is due to the limited number of external pin connections that are feasible on a standard integrated circuit package.

There are two basic types of memory devices used in microcomputer systems. The data memory has data entered

into it and read from it and hence must be capable of both Read and Write functions. The common type of device used for this form of storage is Random Access Memory (RAM). Most semiconductor RAM devices are volatile, i.e. when the power suply is disconnected the data is lost. Program instructions, however, need a storage device which will not change its content when the power is removed. Since the processor does not need to write data into the memory area which is storing the program, this area of memory can be made up of Read Only Memory (ROM) devices.

2.4.-Basic Workshop Software

Two modes of operation are possible for the minicomputer/microcomputer complex(32). The 'master/slave' relationship with the microcomputer as 'master'and with the microcomputer as 'slave'.

In the first case, the microcomputer may view the minicomputer like any other I/O device, and by this means may have access to the large range of sophisticated terminals and peripherals that may be connected to the minicomputer. In the case the microcomputer is the 'slave'; the minicomputer provides the driving force to enable the user to write and debug microprocessor software. The user is thus able to make use of the powerful software facilities available on minicomputers.

2.4.1.-Software Development Aids(33)

The process of transcribing the source code into the binary object code by hand is tedious and prone to errors. Occasionally, small programs may be produced directly in ma-

chine code, but this is unusual nowadays. For this reason the microprocessor instruction set will almost certainly be supported by an 'Asembler' program which converts the symbolic code written by the programmer into machine language instructions which are executable by the processor. It further converts the labelled machine addresses designated by the programmer into real machine memory locations, performs some checks to determine whether the source code statements are valid and provides some useful information about the program.

The mnemonics of the instructions, combined with other control words recognised by the assembler program, constitute a language known as the 'Assembly Language'. It forms a higher level language than programming in object code, but is still a fairly basic and efficient method. Higher level languages exist, e.g. FORTRAN, BASIC, COBOL, etc., and these take care of more of the basic housekeeping functions of the program. A 'Compiler' performs a similar function for a high level language program as an assembler does for an assembly language program.

The assembler and compiler programs are aids to the programmer, but are not part of the end system. Providing they have been written accordingly, they can be run on whatever computer the programmer has convenient access to. If they operate in this method then they are referred to as a 'Cross-assembler' and a 'Cross-compiler' and are known collectively as cross-support. A cross-assembler converts the symbolic code written by the programmer into machine language instructions that are executable not by the processor, but by ano-

-ther computer, such as a minicomputer or large scale computer. Usually, cross-software and test systems are the first items available for a new microprocessor.

Another software aid to the programmer is a 'debugger'. This is a system diagnostic tool that permits the user to analyze the program while it is running. The programmer may insert breakpoints into the program and obtain information, such as register and memory dumps, at specific points of program execution. If the program detects an error, some debugging programs permit the user to make the appropiate modification and let the program continue to run.

A 'simulator' is a specialized program for performing more sophisticated analyses of user programs. A simulator allows the target system program to be run without the target system hardware since it simulates or models the timing and characteristics of hardware, such as peripherals, which may not be available for testing at the time the software is tested.

To assist the programmer in actually writing the source statements in high level or assembly level and manipulating them, there exist another software aid known as an 'editor' program. An editor allows the user to make textual changes in his program, such as adding or deleting a line or a character without reloading or rewriting it. The process is made automatically by the use of various edit commands via a terminal. Some means of saving output from the editor is required. Most commercial development systems now employ floppy disks for storage.

A program that initializes the processor to enable the user program to begin execution is called a 'loader'. Oter types of loaders permits separate groups of machine language code to be linked together and executed by the procesor. These more sophisticated types of loaders are referred to as 'linkage editors'.

Having written the program, compiled or assembled it, if necessary simulated it, and finally debugged it, the resultant object code can be loaded into the program memory of the microprocessor system. During development, the program memory may be of the random access type (RAM), in which case the program needs to be loaded into the RAM's for each test, or Programmable Read Only Memory for which there is the need of a PROM programmer. A PROM programmer is an independent piece of hardware that is available with many development systems for the purpose of programming ROM's. These devices includes such functions as program listing, program by manual keyboard, PROM duplication and program verification functions.

2.5.-Testing the Target System Hardware

There are three methods of hardware testing(31). External hardware emulation, Incircuit testing and direct target system testing. In the first method, the program is executed by the same type of microprocessor, but not the target system. It is suitable especially for testing target systems in which the hardware is relatively uncomplicated. Mock-up of the necessary hardware on the test system will be required for complete testing. In incircuit testing method, the

development system replaces the target processor temporarily. Direct memory access techniques can be employed so that the test system processor can perform all functions in parallel to the target processor. Finally, in direct target system testing, the object software is loaded into the target system either by PROM, ROM emulation hardware or additional RAM. Once in memory, the program is run on the target system. Down loading interfaces are required for this test.

2.6.-The PDP-11/03 - 8X300 Development System

In order to develop the 8X300 microprocessor software, the evaluation board was linked to the PDP-11/03 minicomputer through an interface bus. The attachment to the minicomputer can be made via a standard serial communication channel and also via a programmable parallel interface such as the DRV-11 in the PDP-11. This parallel interface was actually used in the link.

2.6.1.-The DRV-11 Parallel Interface

The DRV-11 paralell line unit(34) is a general purpose interface unit used for connecting parallel line devices to the LSI-11 bus. It features 16 diode-clamped data input lines, 16 latched output lines, 16-bit word or 8-bit byte programmed transfers, logic compatible with TTL and DTL devices and user assigned device address decoding.

Four control lines are available to the peripheral devices, New Data Ready, Data Transmitted, RQSTA (Request A) and RQSTB (Request B)

Any programmed operation that loads either a byte or a word in the output Buffer (DROUTBUF) of the DRV-11

interface causes a New Data Ready H signal to be generated informing user's device of the operation. When data is taken by the processor, a Data Transmitted H pulse is sent to the user device. This data is read on the IN0 - 15H signal lines and must be held on the lines by the user's device until the data input transaction has been completed (at the trailing edge of Data Transmitted Pulse). This is because the input buffer DRINBUF is not capable of storing data.

RQSTA H and RQSTB H are request flags that can be asserted by the user's device when servicing is required and cleared by New Data Ready or Data Transmitted when completed.

2.6.2.-Using the PDP-11 as an External Memory

In a first stage of the development, the PDO-11 memory was used as replacement for ROM. The interconnection between the PDP-11 and the 8X300 microprocessor board was made through a 3-state interface bus as illustrated in Fig.2.3. The microprocessor instructions were executed by stepping the processor with the New Data Ready pulse available at the outport, J2, of the DRV-11 paralell interface.

Address reading and instruction fetching is performed according to the flow diagram of Fig.2.4.

Several programs were written to check the perfomance of the inteface. One of these was the diagnostic program as it is written in the ROM's provided for the 8X300 evaluation board. This program verifies that the assembled kit works correctly.

To monitor the progress of the diagnostic pro-

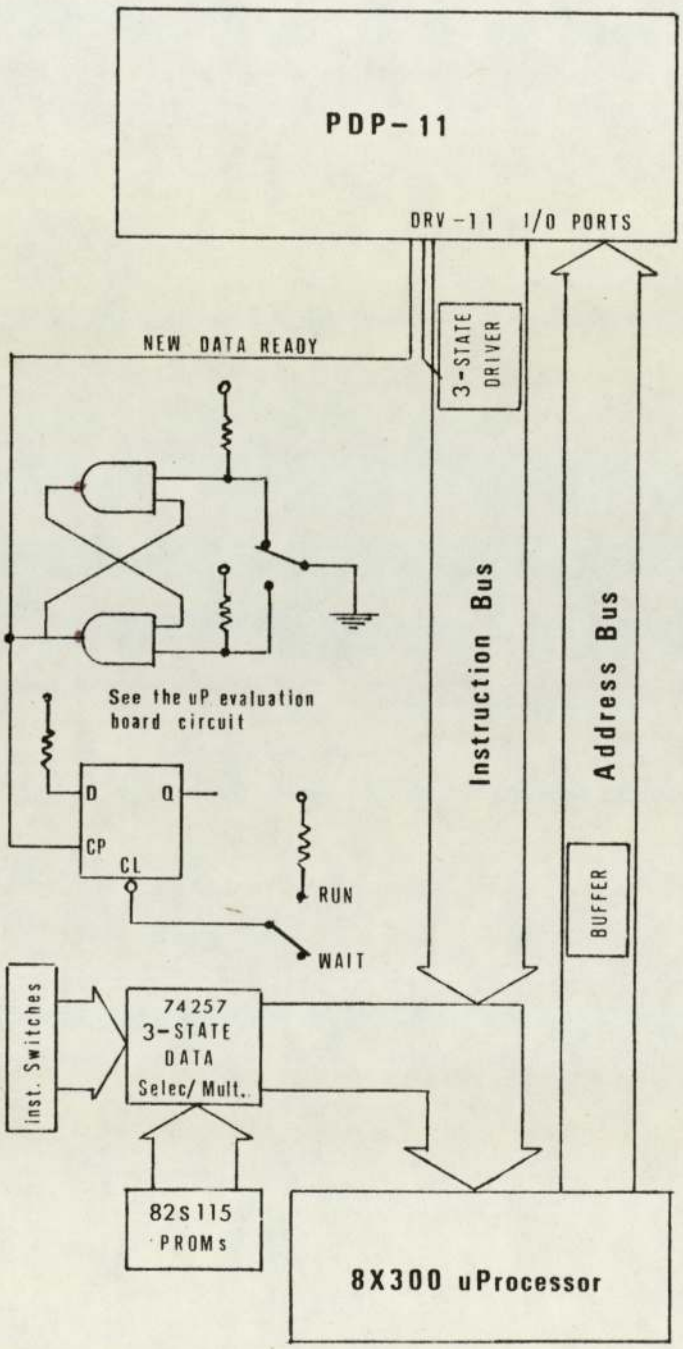


Fig. 2.3.-Interface arrangement between the 8X300 microprocessor and the PDP-11 minicomputer.

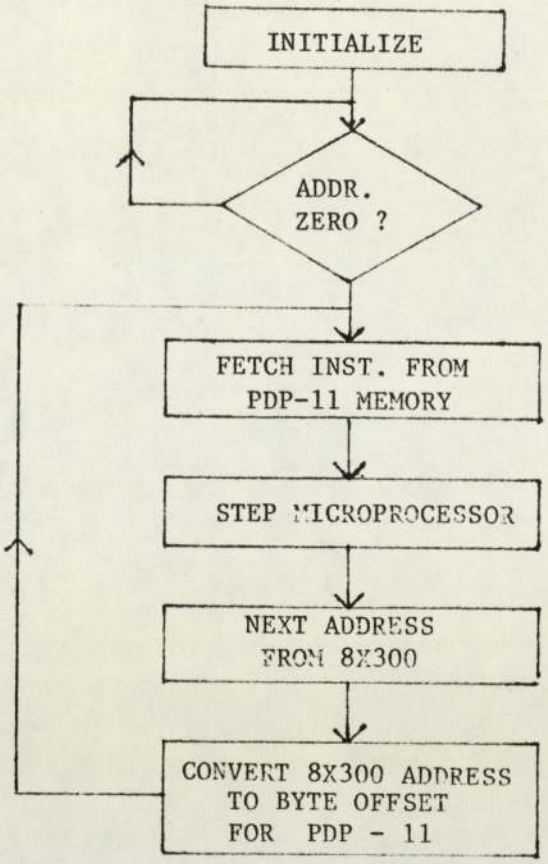


Fig.2.4.-Flow diagram for the interface 8X300 - PDP-11

gram, led arrays were connected to the address and instruction buses and also to the input/output ports of the evaluation board.

The 8X300 code was written within a handshaking routine in direct octal. Fig.2.5 shows the general scheme for these programs and for handshaking.

Object code for tha 8X300 could be generated by a cross-assembler but it was not available at this stage of the work.

2.6.3.-Down Loading Interfacing

To integrate the software and the hardware and

```
 TT:=DX1:PDP8X3.MAC
.TITLE 8X300 USES PDP-11 MEMORY
        ;
        ;*************************************
        ;* THIS PROGRAM IS INTENDED TO INTER *
        ;* CHANGE DATA BETWEEN THE 8X300 MI  *
        ;* CROPROCESSOR AND THE PDP-11 MEMO  *
        ;* RY                                *
        ;*************************************
.MCALL  ..V2.,.,.REGDEF
..V2..
.REGDEF
DROUT=167772
DRIN=167774
START: CLR      R1
       MOV      #160000,DROUT
A:     TST      DRIN
       BNE      A
B:     MOV      X(R1),DROUT
       MOV      DRIN,R1
       ASL      R1
       BR       B
X:     .WORD                        ;*************************
       .WORD                        ;*************************
       .WORD                        ; THIS SPACE IS PROVIDED
       .WORD                        ; FOR THE DATA TO BE PRO
       .WORD                        ; CESSED BY THE  8X300 -
       .WORD                        ;       MICROPROCESSOR
       .WORD                        ;*************************
       .WORD                        ;*************************
       .END    START
*
```

Fig.2.5.-8X300 uses PDP-11 memory. Hand-shaking and data for the 8X300

to perform a direct testing of the target system, data being developed in the PDP-11 has to be transferred to the evaluation board. To hold the instruction program a set of RAM's were added to the board.

Fig.2.6 shows a system using a group of four RAM's, providing for 250 x 16 bit instructions, and Fig.2.7 shows an arrangement using two groups of four RAM's, 500 x 16 bit instructions. The interfaces for interchanging data between the minicomputer and the 8X300 evaluation board are also shown in these figures. As seen, two program controlled

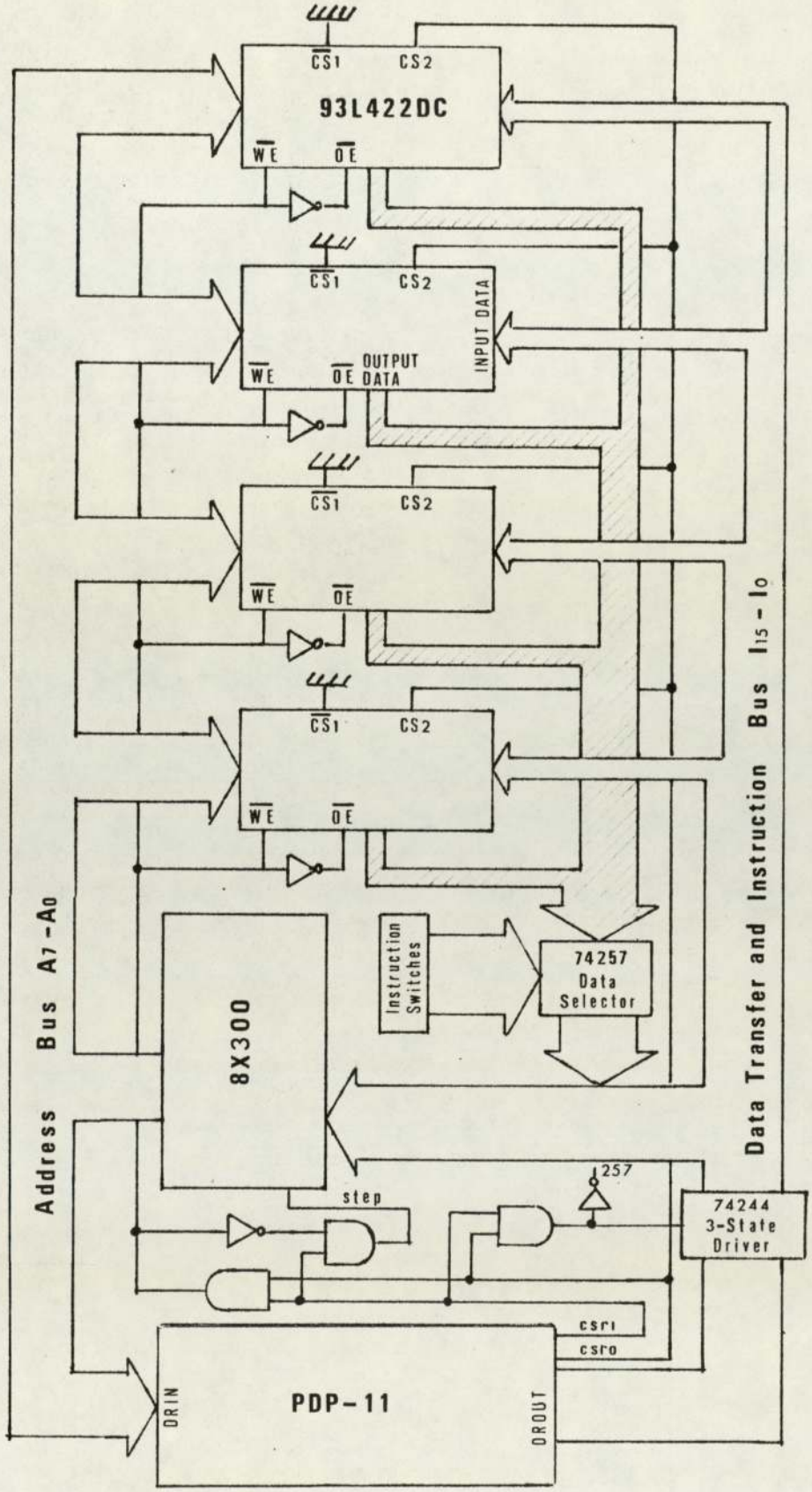


Fig. 2.6.-8X300 uses a system of 4 RAM's providing for 250x16 bit instructions.The RAM's are loaded from the PDP-11 minicomputer.

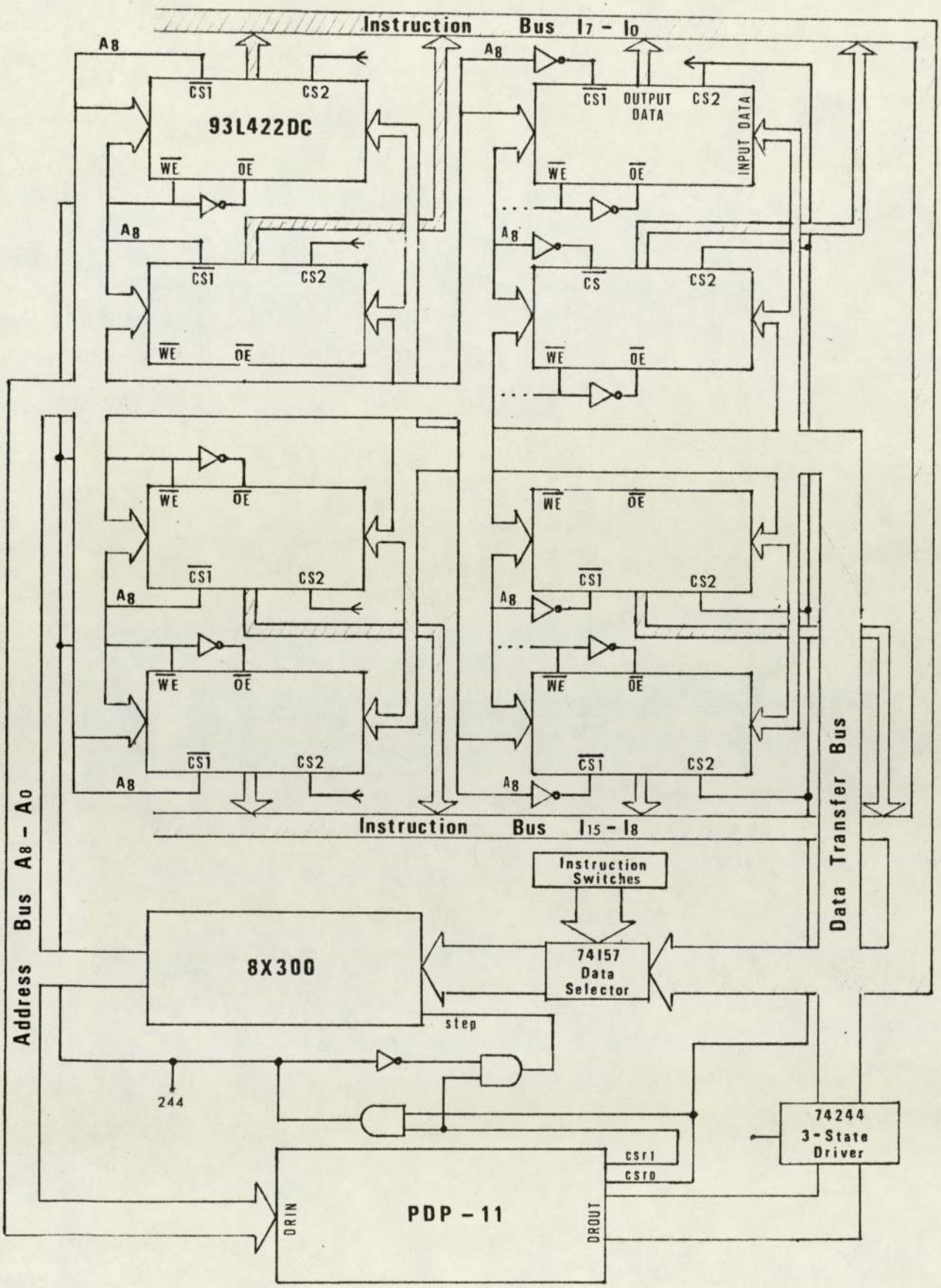


Fig. 2.7.-8X300 uses a syetm of 8 RAM's providing for 500x16 bit instructions. The RAM's are loaded from the PDP-11 minicomputer.

bits, csr0 and csrl, are used to chip select process, to control the write and read operations, to step the microprocessor to make it place a new address on the address bus, since it is used as an address counter in the in the downloading process, and to disable the 3-state Data Transfer and Instruction Bus. This last function is important once the process of loading has been completed, in order to separate the evaluation board from the minicomputer and have it running as a stand-alone device.

In order to achieve speeds of processing compatible with the maximum speed of the 8X300 microprocessor of 250 nsec. per instruction, the RAM's to be added to the microcomputer board have to have an access time compatible with the required speed of processing.

Fairchild TTL isoplanar 93L422 Random Access Memories were used. These are fully decoded 1024-bit organized 256 words by four bits (256x4-bit). Word selection is achieved by means of an 8-bit address, A0 through A7. Two Chip Select are provided. The device typical address access time is 45 ns. (35)

Table 2.1. 93L422 TRUTH TABLE

INPUTS					OUTPUTS		
$\overline{OE}$ PIN 18	$\overline{CS_1}$ PIN 19	CS_2 PIN 17	$\overline{WE}$ PIN 20	D_{1-4} PINS 9,11,13,15	93L412 O.C.	93L422 3-STATE	MODE
X	H	X	X	X	H	HIGH Z	Not Selected
X	X	L	X	X	H	HIGH Z	Not Selected
L	L	H	H	X	$O_1 - O_4$	$O_1 - O_4$	Read Stored Data
X	L	H	L	L	H	HIGH Z	Write "0"
X	L	H	L	H	H	HIGH Z	Write "1"
H	L	H	H	X	H	HIGH Z	Output Disabled
H	L	H	L	L	H	HIGH Z	Write "0" (Output Disabled)
H	L	H	L	H	H	HIGH Z	Write "1" (Output Disabled)

H = HIGH Voltage; L = LOW Voltage; X = Don't Care (HIGH or LOW); HIGH Z = High Impedance; OC = Open Collector

2.6.4.-Down-loading System Interfacing Software

Two subroutines are called by a Main Fortran program (36). Fig.2.8 shows the flow diagram for this Main program. Subroutine 'CODEIN'(36), written in Fortran, places the data to be transferred to the RAM's in an array in the PDP-11 memory. The bidimensional array also contains the number of 16-bit words to be transferred which is used later as a loop counter in the loading routine. The flow diagram for CODEIN is shown in Fig.2.9.

The soubroutine LOAD8X has been written in Assembly Language. This subroutine provides the control signals for the handshaking and transference of data in the PDP-11/8X300 interface. LOAD8X loads the RAM's of the 8X300 evaluation board and set it running from address zero. Fig.2.10 shows the flow diagram of the LOAD8X subroutine.

The data to be transferred to the RAM's, i.e., the object code for the 8X300,is generated by using a modified version (36) of the Signetics'MCCAP' cross-assembler, and placed in a file from where it can be fetched by the subroutine CODEIN.

Once the process has been completed, a message such as 'Transfer to 8X300 Complete' is returned. From this point the evaluation board can be physically separated from the minicomputer and work as a stand-alone system.

The whole process of loading the RAM's and having the system running separated from the minicomputer takes as much as five minutes, provided the object code has been generated previously and placed in a file.

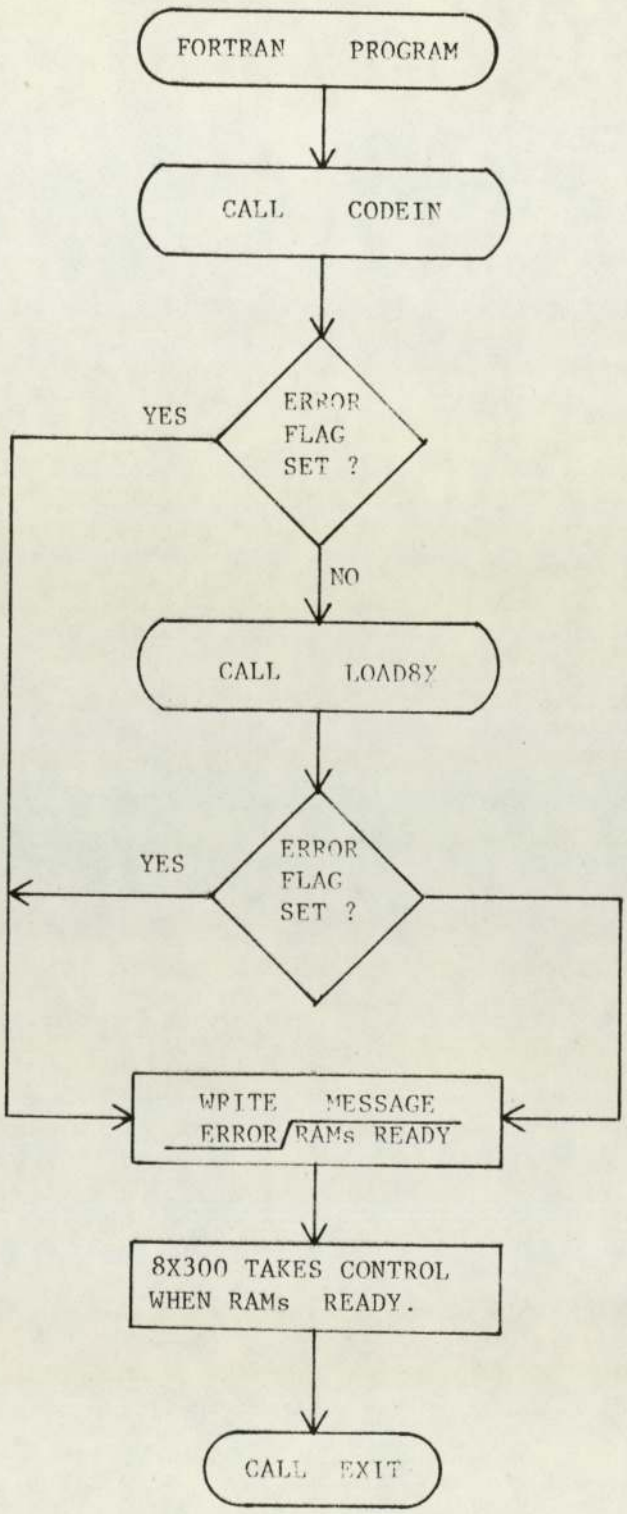


Fig. 2.8.-Flow diagram for the process of loading the 8X300 RAMs from the PDP-11 minicomputer. Two subroutines, CODEIN and LOAD 8X are accessed by a Fortran Program.

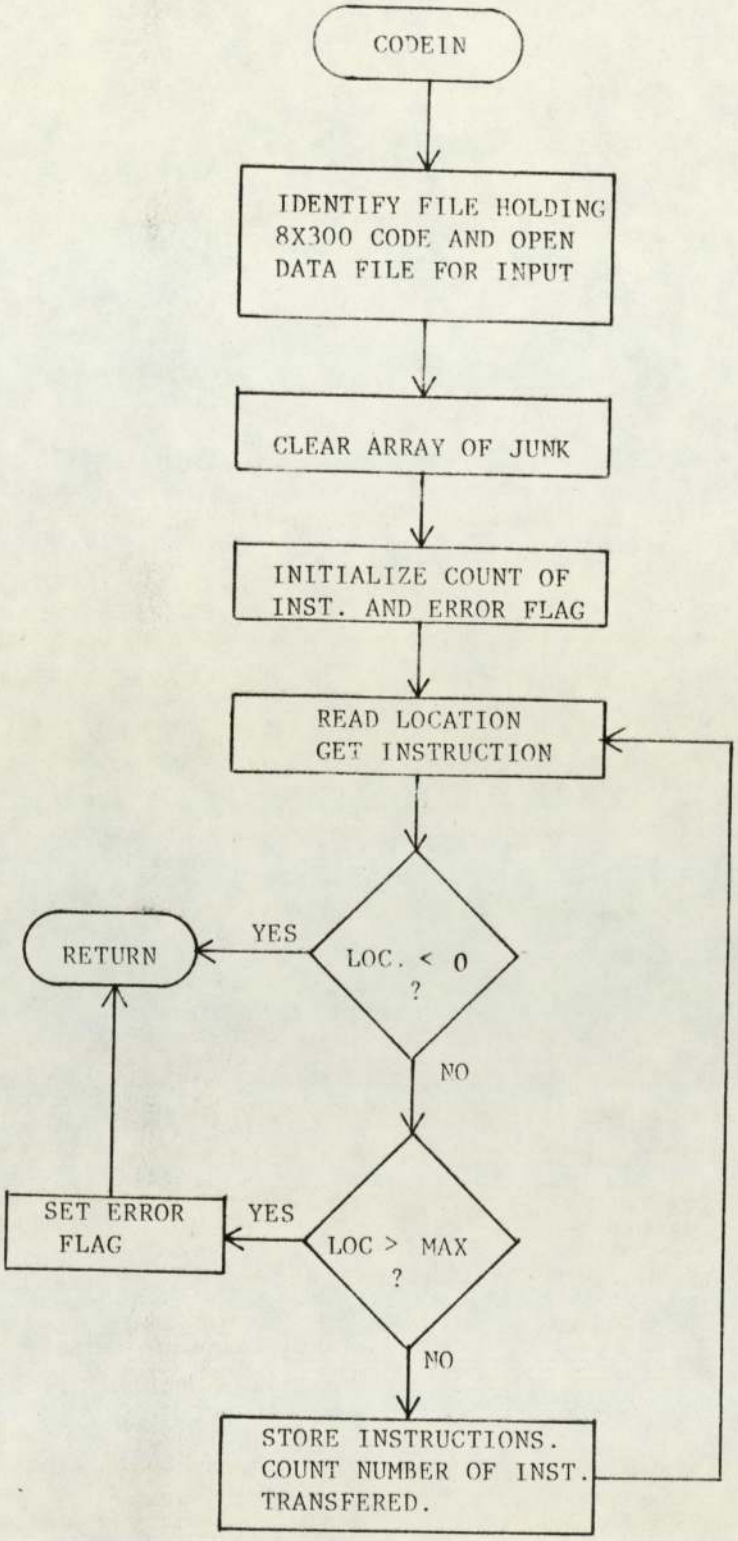


Fig.2.9.-Flow diagram for the subroutine 'CODEIN'.

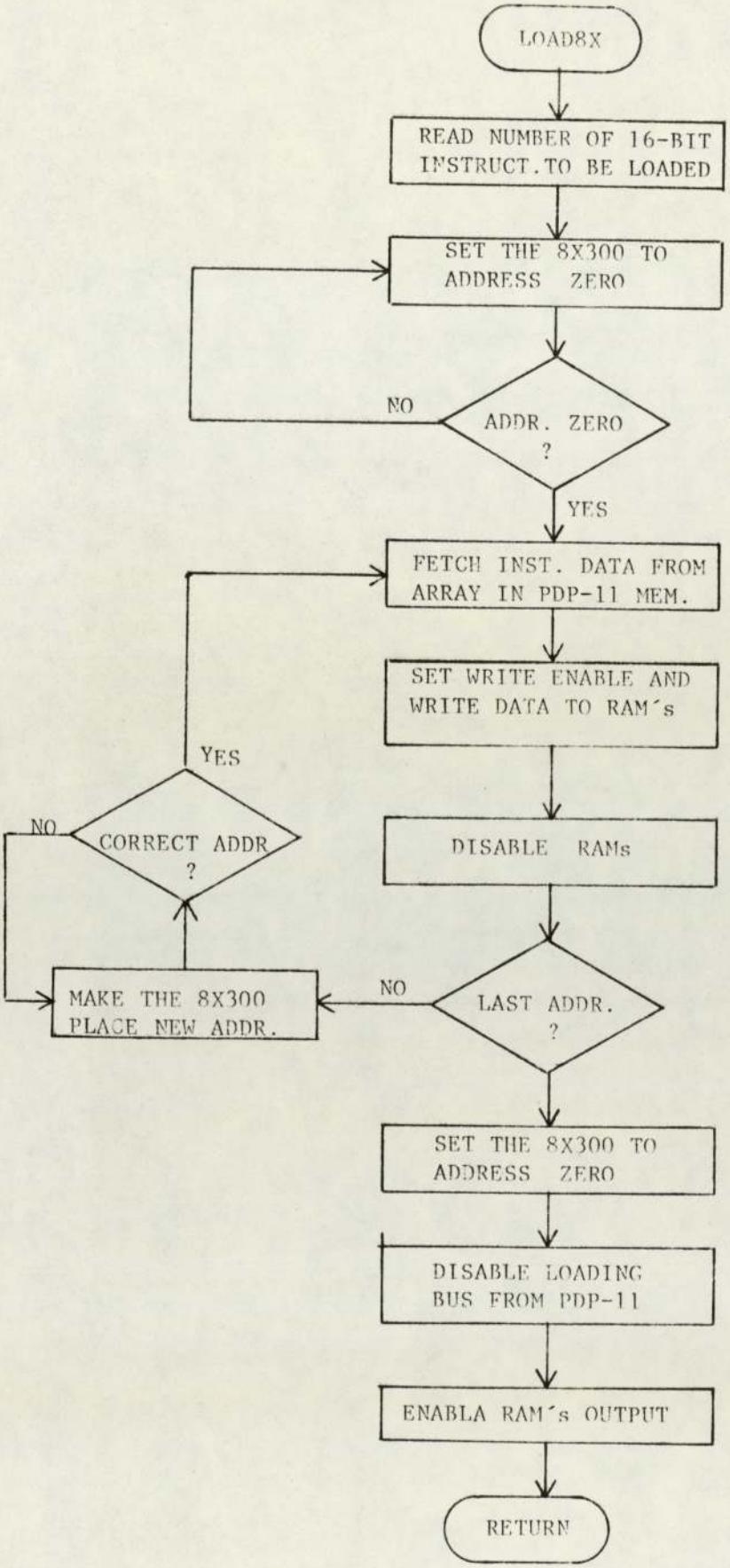


Fig.2.10.-Flow diagram for the sub-routine LOAD8X.

CHAPTER III

THE 8X300 MICROPROCESSOR EVALUATION BOARD

3.1.-Introduction

This chapter presents a brief review of some of the important features of the Signetics 8X300 microprocessor evaluation board,such as its software capabilities, hardware facilities, architecture, etc., aiming to introduce the reader to the main tool used in the developing of the real time speech encoder.

The development of a real-time speech processing system demands a quick response to requests for information. This task can be fulfilled by using very fast and also very expensive general purpose computers. Fortunately, newly developed and commercially available microprocessors boards and microprocessor slices allow us to do the same job at very low cost. These special purpose computers have the software capabilities and the speed required for real-time speech processing(37).

3.2.-The 8X300 Evaluation Board (38)(39)(40)

The 8X300 microprocessor is a fixed instruction bipolar microprocessor, packaged as a 50-pin DIP.

Fig.3.1 shows the schematic diagram of the evaluation board which includes, 4 I/O ports for external device interface, 256 bytes of temporary (working) data storage and 512 words of program storage where development routines may be entered.

Two operating modes are available with the 8X300, the WAIT and the RUN modes.

In the WAIT mode, the program may be on board or externally single stepped. On board, instructions may be

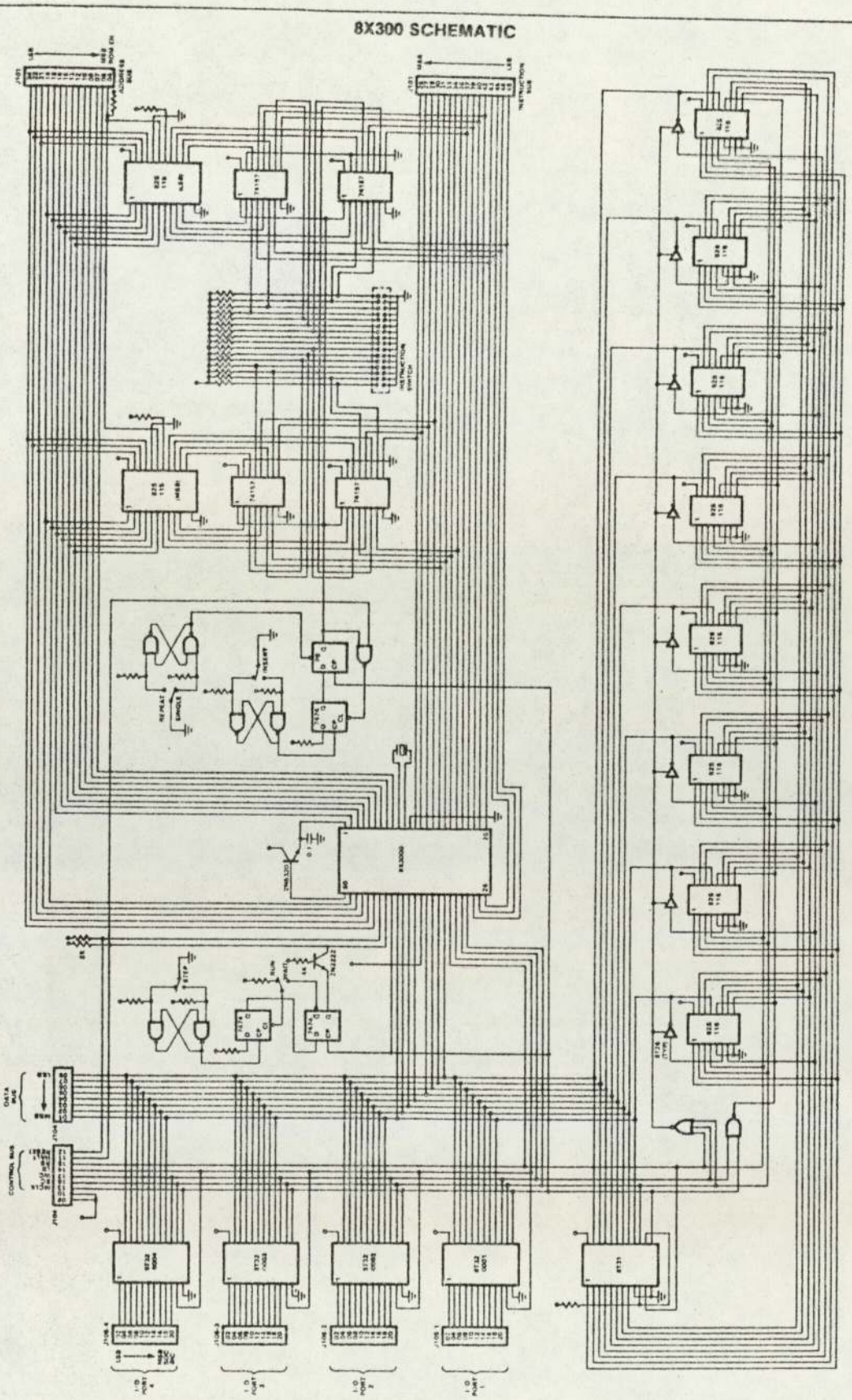


Fig. 3.1.-Schematic diagram of the 8X300 evaluation board. (from Signetics 8X300 user's manual).

executed one at a time by depressing the STEP key. Externally, it can be stepped by a master controller such as the PDP-11 minicomputer by acting on the same circuit as the STEP key does. In this asynchronous arrangement, the speed of the microprocessor to process each instruction may be kept at its maximum rate, the overall speed of processing, however, is dictated by the master controller.

Instructions for the microprocessor can be read from either the PROM's, added memory or the instruction switches. Instructions may also be fetched from the memory of another processor when interfaced with the 8X300. 3-state data selectors have to be used in some of these cases to prevent interference between the differents sources.

The addressing capabilities of the 8X300 microprocessor is 8192 program instruction locations, as there are 13 bits available for addressing. There are also 9 bits for input/output port addressing, providing for up to 512 I/O ports.

To control the input or output of data to and from the I/O ports, two bits named Bit Input Control (BIC) and Bit Output Control (BOC) are used. The I/O ports contain 8 data latches accesible from either the microprocessor or the user port and, to avoid conflicts at the data latches, input from the microprocessor port is inhibited when BIC is at low level, in other words, user port has priority over the microprocessor port for data input.

3.2.1.-8X300 Architecture

The 8X300 is a complete processor on a single chip. The clock generator circuit oscillates at a frequency

determined by an external crystal or timing capacitor. The entire processor operates from a single +5 volts suply.

Fig. 3.2 illustrates the 8X300 architecture. As seen, this includes eight 8-bit working registers, an arithmetic logic unit (ALU), an overflow register, and the 8-bit Interface Vector Bus (IVB).

The Auxiliary register can be used as a normal working register but it is used as the implied operand in ALU operations that require two data inputs such as in the ADD, AND, and XOR (Exclusive-Or) instructions.

The control registers include the Program Counter (PC), The Address Register (AR) and the Instruction Register (IR).

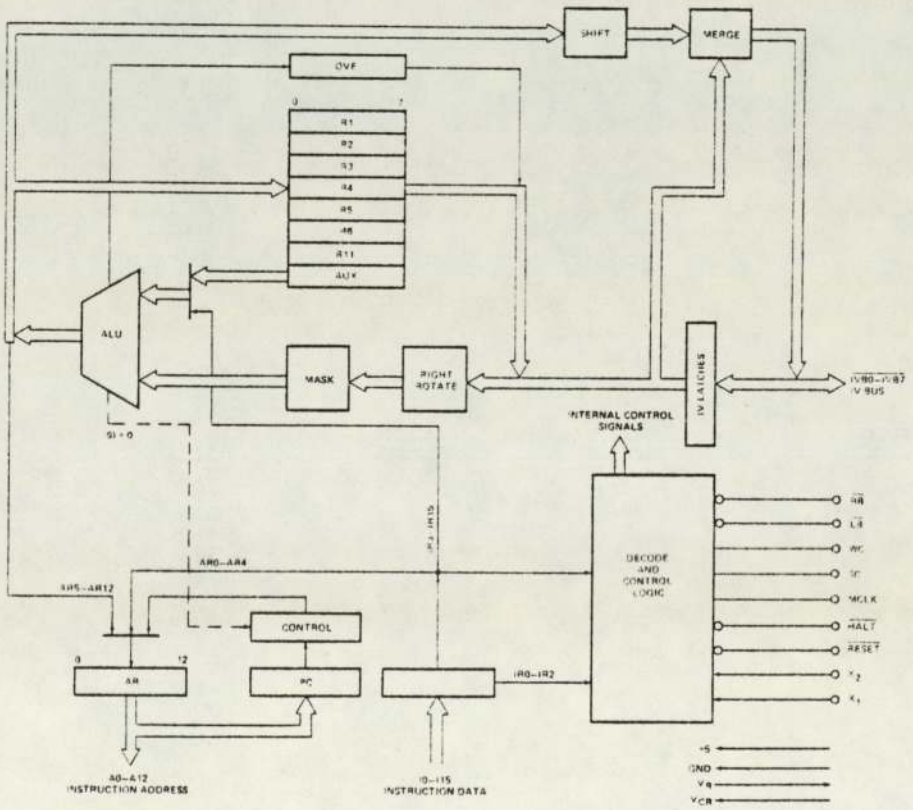


Fig.3.2.-8X300 architecture (from Signetics 8X300 user's manual).

The Program Counter is thirteen bits wide and at all times, this register addresses the next program memory location from which an instruction will be fetched.

The Address Register is a 13-bit register containing the address of the current instruction being accessed from program storage. This is not a programmable register but a location within which effective program memory addresses are computed before being output to the program memory. Its contents, as in the Program Counter, can be changed as the result of special instructions.

The Instruction Register is a 16-bit and it holds the instruction word currently being executed.

All data input to or output from the 8X300 goes via the Interface Vector Bus (IVB). The Interface Vector Bus serves both as an address and data bus, being its function determined by the control signals Select Command(SC) and Write Command(WC).

Two banks for data movment are available. These are the Left and Right Banks,(LB), (RB). Because the Left and Right Banks are independent, one IV byte on each bank can be enabled simultaneously.

Figure 3.3 shows a typical configuration using the 8X300 microprocessor. The concepts of Left Bank (LB), Right Bank (RB), Interface Vector Bus (IVB) control, Select Command(SC), Write Command(WC), and user I/O control, Bit Input Control(BIC), Bit Output Control(BOC), are clearly seen in this figure.

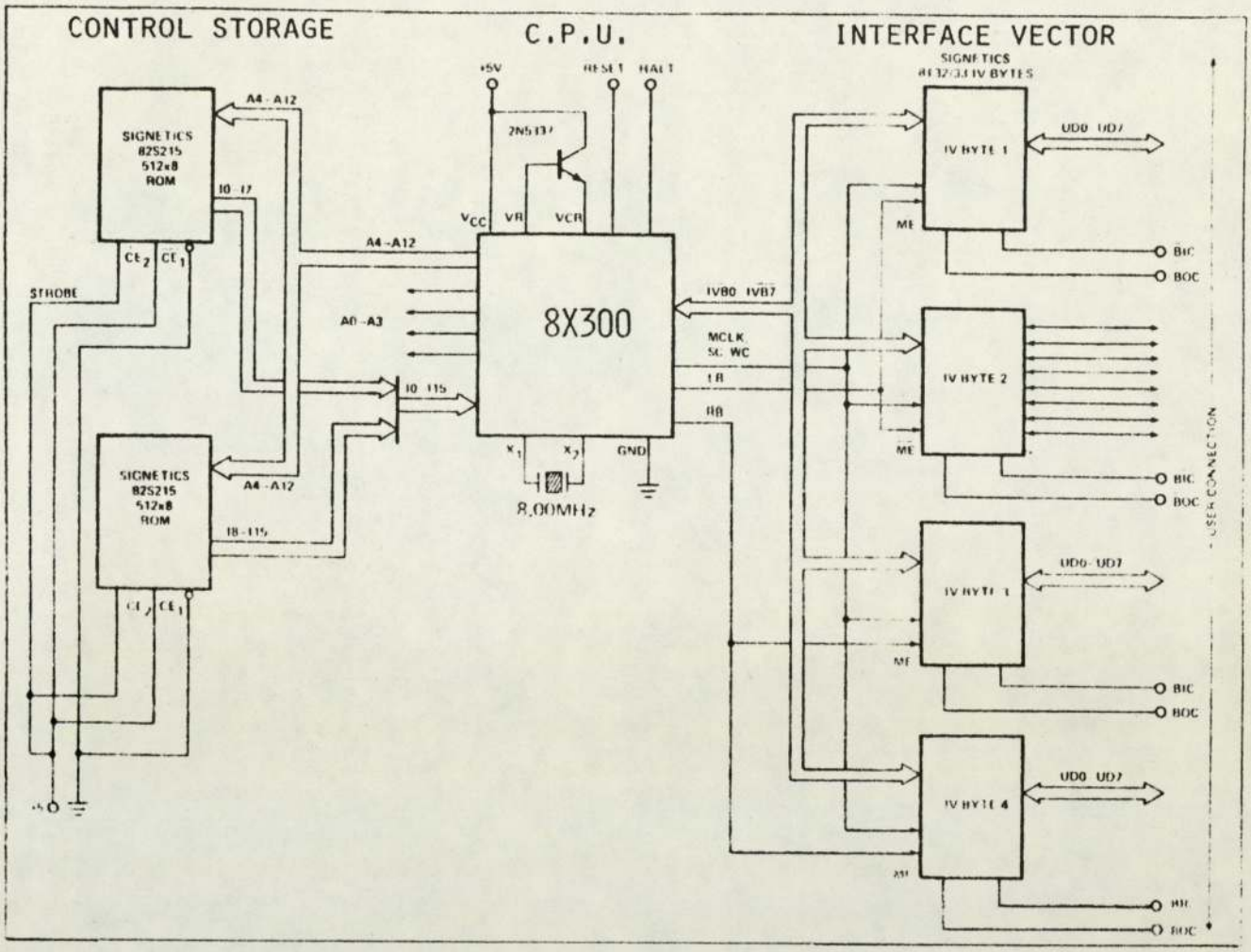


Fig.3.3.-Example of control system
(from 8X300 user's manual)

3.2.2.-Instruction Cycle (Fig 3.4)

The instruction cycle may be viewed as having two halves, an input and an output phase. During the first half of the instruction cycle, data are brought into the processor and stored in an interface vector latch. Storage is completed during the first quarter cycle, and in the next quarter cycle the data is processed through the ALU. In the second half cycle, the data is presented to the bus and finally clocked into the designated I/O port.The instruction address for the next operation is presented at the output of the processor during the third quarter of the instruction cycle. The instruc-

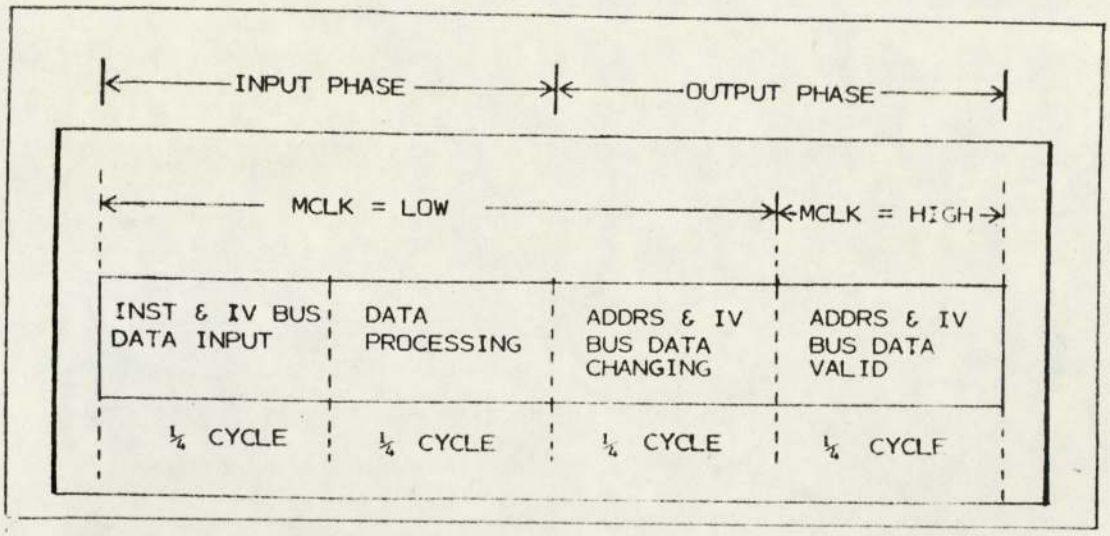


Fig.3.4.- Instruction cycle time.

tion returns to the processor during the first quarter of the cycle in which it is to be executed.

3.2.3.-System Clock

The 8X300 has an integrated oscillator which generates all necessary clock signals. The oscillator is connected directly to a series resonant quartz crystal via pins X1 and X2. The crystal resonant frequency is 8.00 MHz for a 250 nsec. system.

In lower speed aplications, where cycle time need not to be precisely controlled, a capacitor may be connected between X1 and X2 to drive the oscillator. If cycle time is to be varied, X1 and X2 should be driven from complementary outputs of a pulse generator.

3.2.4.-HALT and RESET

HALT and RESET signals are available to stop internal operation of the microprocessor and to set to zero the contents of the Program Counter and Address Register, respectively.

3.3.-Instruction Set Summary

The 8X300 instruction set is composed of eight classes of instruction, each with variations depending upon the operand specifications. These instructions provide for: Arithmetic and Logic operations, ADD, AND, and XOR (Exclusive OR), Data movement, MOVE and XMIT (Transmit), Context alteration, JMP (unconditional Jump), NZT (Test and branch on Non-Zero) and XEC (Execute the instruction at the address specified without program counter alteration).

Table 3.1 shows the instruction mnemonic codes and their actions on the operands.

code	mnemonic	
0	MOVE	data to another place
1	ADD	two sets of data
2	AND	AND operation on two sets of data
3	XOR	perform an EX-OR op. on two sets of data
4	XEC	execute an instruction at a given address
5	NZT	test data field for Non-Zero contents
6	XMIT	transmit a constant to a data field
7	JMP	jump to another instruction sequence

Table 3.1.-8X300 Instruction summary

Every 8X300 instruction has a single 16-bit object code, the 3 high-order object code bits define the instruction class, while the next 13 bits provide additional operand or qualifying data. The operand may consist of the following fields; Source (S) field, Destination (D) field, Rotate/Length (R/L) field, Immediate (I) Operand field, and(Program Storage) Address (A) field.

The MOVE, ADD, AND and XOR instructions have identical object code format. This format is illustrated in Fig. 3.5.

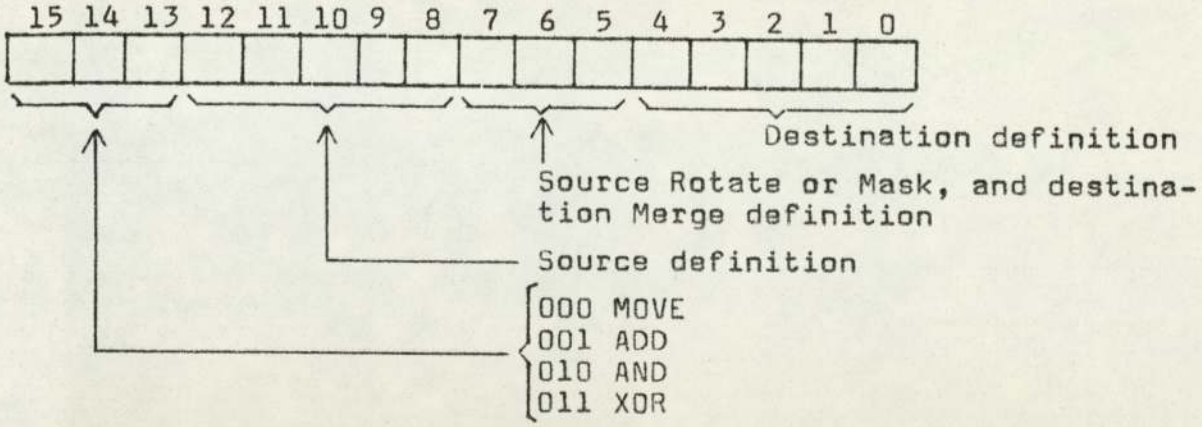
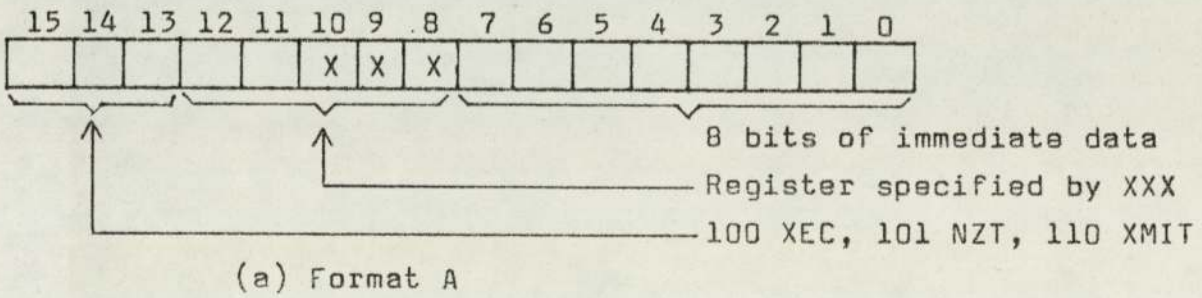
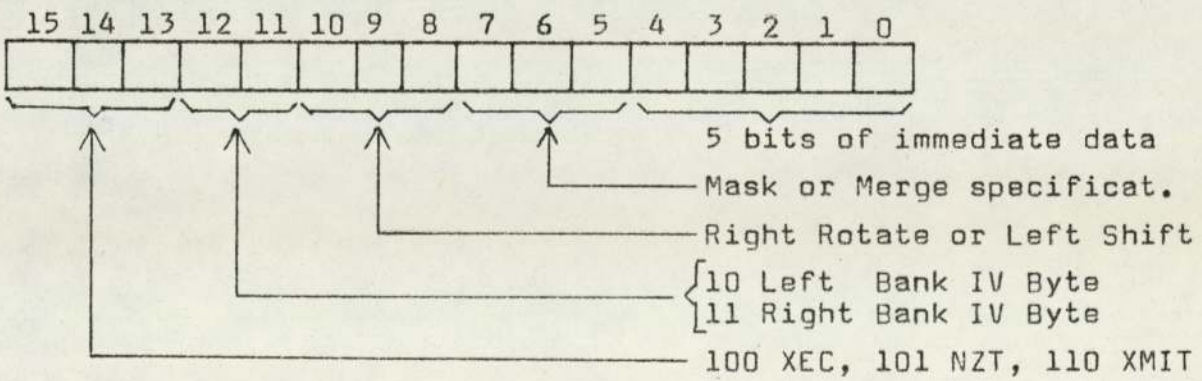


Fig.3.5.-Object Code Format for the MOVE ADD, AND and XOR instructions

Two instruction object code formats are possible for the XEC, NZT and XMIT instructions. These two formats A, and B are illustrated in Fig.3.6 (a) and (b).



(a) Format A



(b) Format B

Fig.3.6.-Object Code Formats for the XEC NZT and XMIT instructions.

The Format A object code uses bits 8 through 12 to specify a general purpose register or the auxiliary register.

The Format B instruction object code uses bits 5 through 12 to specify the currently selected left bank or right bank IV byte, where byte contents will be subject to a mask and a rotate.

The JMP (jump) instruction when executed, the 13 operand bits are loaded directly into the Program Counter, performing in this way a simple unconditional jump to any location in program memory. Fig.3.7.

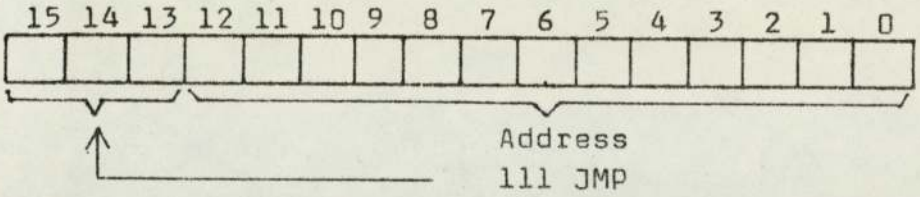


Fig.3.7.-Object Code Format for the JMP instruction

For instructions where both the Source and Destination are registers, only the Rotate function is available. The Mask function allows selection of the least significant L (Length) bits of the rotated IV bus source data for subsequent processing.

3.4.-8X300 Cross Assembly Program (MCCAP)

The 8X300 microcontroller (40) counts with a programming language, the 8X300 MCCAP, which allows the user to write programs for the 8X300 in symbolic terms. This Cross-Assembly program translates the user's symbolic instructions into machine-oriented binary instructions.

The programmer is allowed to define symbolic variable names for data elements by means of the Assembler Declaration Statements. Individual bits and sequences of bits in working storage on the Interface Vector may be named and operated upon directly by 8X300 instructions.

The MCCAP, apart from standard features such as mnemonic opcodes and address labels, includes macros, automatic subroutine handling and conditional assembly. After assembling the source input, MCCAP produces an assembly listing and an object module. Information, such as memory addresses and their machine code contents, the original source code, error indications and the programmer comments, may be obtained from the assembly listing.

As MCCAP is written in ANSIFORTRAN IV, it can be run on large computers and most 16-bit minicomputers.

Written literature (41) about the microcomputer cross assembly program (MCCAP) describes details about syntax and format rules, symbolic references, assembler declarations, assembler directives, executables statements, macros, the assembly process and examples.

CHAPTER IV

DELTA MODULATION

4.1.-Introduction

Delta modulation has received widespread attention in this decade and it can be said that it is now well introduced into the telecommunication field and being introduced in many other areas. Adaptive delta modulation (ADM) is replacing PCM in some parts of telephone communications where PCM has been used exclusively(42). The use of delta modulation as a source encoding scheme has been shown to be a viable and efficient technique for use in a packet voice system(43). Applications such as programmable digital filters, remote motor control, speech scrambling, have been recently reported(44).

ADM has proved to be very simple for hardware realization with lower level of complexity than other forms of speech encoding systems, for comparable quality of speech.

With the advent of high speed microcomputer systems, the way is open for the implementation of these encoders with the flexibility and other advantages that microprocessor usage implies. By using the microcomputer technology it is very easy to try out new systems, new algorithms or to produce alterations on them in a speedy way.

4.2.-Linear Delta Modulation

The basic delta modulator is shown in Fig.4.1(45). At the input of the encoder the incoming analogue signal M(k) is compared to the quantized approximation of $\hat{M}(k)$ in the comparator. If M(k) is greater than $\hat{M}(k)$, the comparator output e(k) becomes a logic 1, for the duration of a clock period, while if M(k) is less than $\hat{M}(k)$, the comparator output e(k)

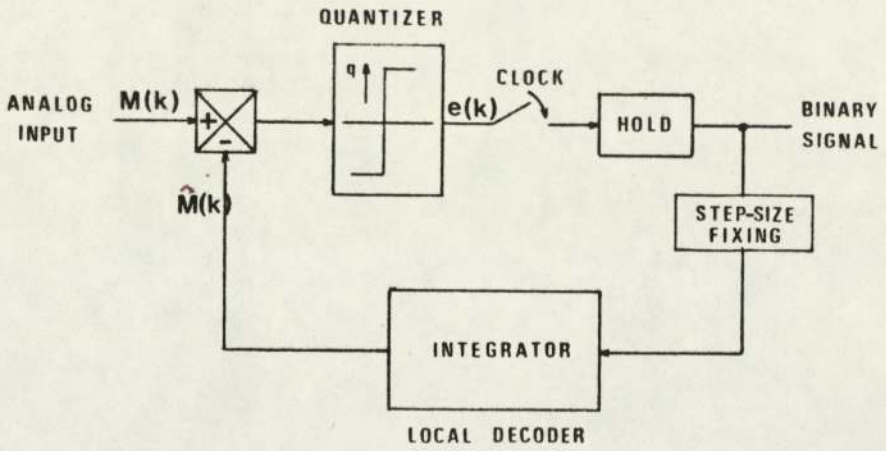


Fig.4.1.-The basic delta modulator

becomes a logic 0, ie, the system then tracks the analog input signal M(k) with the locally demodulated feedback signal $\hat{M}(k)$ in the attempt to minimize the error signal, and encodes this error into binary form e(k). The voltage e(t) is sampled at the delta modulator bit rate f_s and then transmitted.

The step size Ss, by which the quantized feedback signal $\hat{M}(k)$ is increased or decreased every sampling period is constant in conventional (linear) delta modulation. If M(k) is geater than $\hat{M}(k)$ then $\hat{M}(k)$ is increased in the following sampling period by the fixed step size Ss = So, while if M(k) is less than $\hat{M}(k)$, $\hat{M}(k)$ is decreased by So.

Fig.4.2 illustrates the case when the input analog signal M(k) is a amplitude variable sinewave of frequency f_m being sampled at the rate of $f_s = 16f_m$ in (a), (b), and (c), and at the sampling rate of $f_s = 32f_m$ in (d). The step size is kept fixed in all these cases.

Fig.4.2 brings out the distinction between two ty-

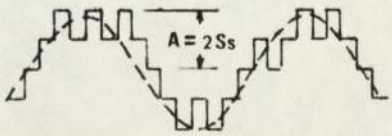
pes of encoding errors in delta modulation, viz, granular or quantizing noise and slope overload distortion.

Granular noise is produced by the difference between the original and reconstructed analog signals in the quantizing process. This type of error can be reduced by increasing the sampling frequency and decreasing the magnitude of the step size. As seen in Fig.4.2 (a) the comparatively large step size (So = A/2) creates a significant amount of quantization noise as the feedback signal $\hat{M}(k)$ makes substantial variations about the small input signal. The result is that the lower the input power the lower the S/N ratio.

Fig.4.2 (b) shows a different case in which the slope of the sine wave is now too steep to be followed by the linear delta modulator. Any fast increase or decrease in the input signal can no longer be tracked by the feedback signal and the signal recovered at the output of the decoder may be very different from the original input signal. The difference between $M(k)$ and $\hat{M}(k)$, ie, the noise, is due in this case to slope overload condition. Once slope overload has occurred noise power then increases more rapidly than the signal power. As a result, the S/N ratio decreases with increasing signal power after slope overload has occurred.

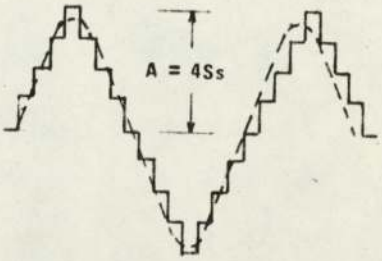
In Figures 4.2 (c) and (d), it can be seen that the feedback signal $\hat{M}(k)$ follows the input analog signal. There is at this point an optimized compromise between the step size, the amplitude of the sinewave, the frequency of the analog input and the sampling rate.

A graphical description of all that has been said



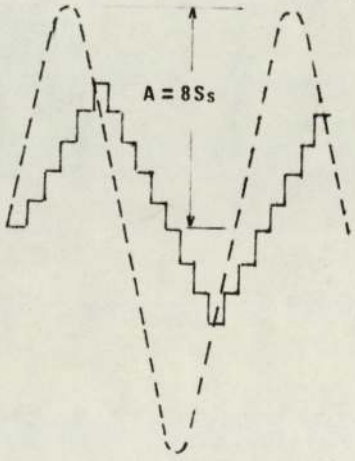
(a)

The amplitude A of the sinewave is A = 2 Step size Ss. The sampling frecuency, f , is f_S = 16 sinewave frecuency, f_m.



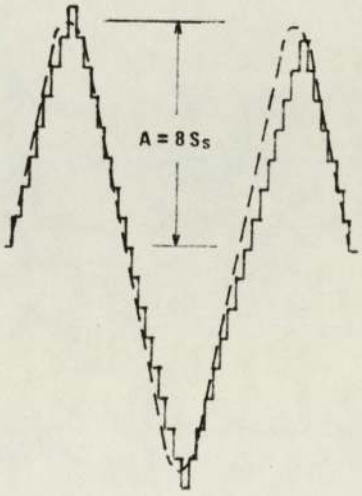
(c)

Signal amplitude to step-size ratio has been optimized. The sampling frecuency is still 16 times the sinewave frecuency.



(b)

The slope of the sinewave is too step to be followed by the LDM at the sampling rate of f_S = 16 f_m. The amplitude is A = 8 S_s.



(d)

As the sampling rate is now f_S = 32 f_m, the LDM is able to follow the input sinewave. The sinewave amplitude to step-size ratio is 8 as in (b).

Fig.4.2.-A sinewave of frequency f_m is being sampled at bit rates of 16 and 32 f_m. The step-size is kept fixed

about signal-to-noise ratio and amplitude of the input signal is represented in figure 4.3. This is related to the dynamic range of the lineal delta modulator.

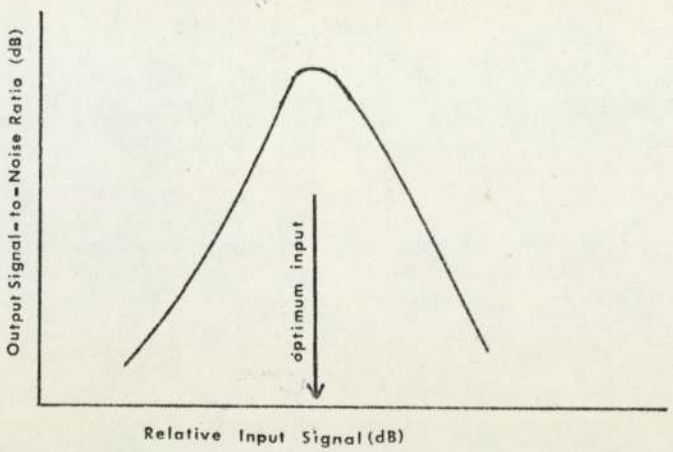


Fig.4.3.-Dynamic Range of the Linear Delta Modulator

Figure 4.3 shows that the system is optimum for a very narrow range of input signal power, ie, for an acceptable signal-to-noise ratio. The slope for low-input power is due to the granular noise produced by the finite step size. The downward slope for high-input signal power is accounted for by the inability of the accumulator to follow the input.

The dynamic range of linear delta modulator has been found to be about 12 dB(46).

If we consider a sinewave input

$$m(t) = A \sin Wm\, t \qquad (4.1)$$

its maximum slope is given by AWm.

The maximum slope of the linear delta modulator with step size So and sampling frequency fs, is Sof_s. Then slope overload can be avoided if

$$S_o f_s \geq 2\pi f_m A \qquad (4.2)$$

This equation sumarizes what is seen in Fig.4.3. Linear delta modulation is more tolerant of large, low frequency signals and less tolerant to large, high frequency signals.

4.3.-Adaptive Delta Modulation

Since the invention of the delta modulation, research workers have been presenting new approaches and techniques such as adaptive delta modulators (ADM) aiming to improve the coding efficiency of the basic system.

Linear delta modulators suffer the disadvantage of a limited dynamic range for an acceptable signal to noise ratio. Quantization noise and slope overload distortion severely limit the usefulness of this scheme. Even when the step size is optimized, the perfomance of these modulators will only be satisfactory at sampling frequencies that may be undesirably high.

Fig.4.4 illustrates the mechanism of an adaptive delta modulator and demostrates how suitable increases and decreases of step size facilitate better encoding during steep and flat regions of the input waveform. For comparison purposes the tracking of a linear delta modulator is included in the figure.

In the slope overload region the step size increases are controlled by a suitable algorithm to ensure good tracking. In the flat region of the input waveform the step size recovers its smallest size to minimize quantization noise.

Many adaptive algorithms have been presented in the literature within the instantaneous companding, syllabic

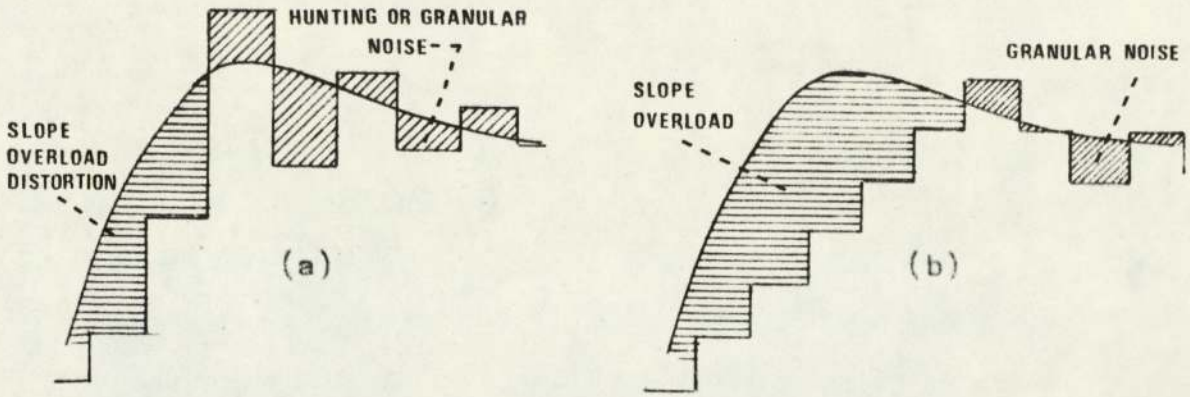


Fig.4.4.-(a)Tracking of an adaptive delta modulator. (b)Tracking of a linear delta modulator.

companding and hybrid companding adaptive schemes, and every strategy either intended for speech or for rapidly varying signals such as television signals results in a considerable improvment in efficiency. Fig.4.5 shows the signal to noise ratio variation with input power for a linear and an adaptive delta modulation system.

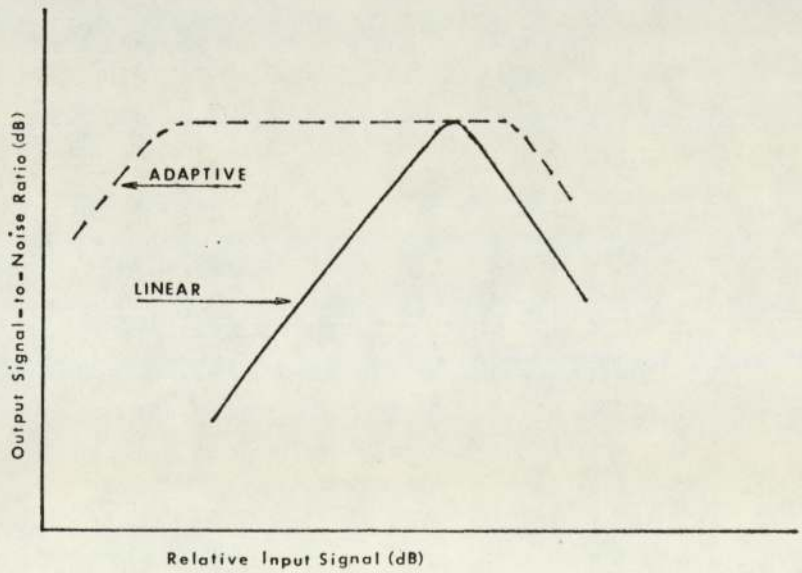


Fig.4.5.-Comparison between dynamic ranges of linear and adaptive delta mod. (44)

The ratio is relatively constant in an adaptive modulation system, but in a linear delta modulation system it depends on signal level.

4.4.-The SONG Voice ADM Algorithm(43)(46)(47)

The reason for chosing the Song voice adaptive delta modulator is that it is very easy to implement and produces good quality speech at fairly low bit rates. 99% word intelligibility at 16 kbit/s bit rate with a dynamic range of 40 dB has been reported(43).

The basic arrangement of the Song adaptive delta modulator is shown in Fig.4.6.

The Song algorithm produces the step size S(k) which minimizes the mean square error between the incoming signal M(k) and the signal estimate $\hat{M}(k)$ at each sampling instant.

The equation describing the Song algorithm is,

$$X(k+1) = X(k) + S(k+1) \qquad (4.3)$$

where X(k) is the digital estimate of the incoming analog signal at the sample time k/f_s, where f_s is the sampling rate, and S(k+1), the current or new step size at time $(k+1)/f_s$.

The equation for the step size is

$$S(k+1) = S(k)e(k) + S_o e(k-1) \qquad (4.4)$$

where e(k) is the sign of the current error which occurs at k/f_s, e(k-1) is the sign of the previous error, S(k) is the previous step size and S_o is the voltage associated with the minimum step size. If M(k) is the signal value at time k/f_s, then

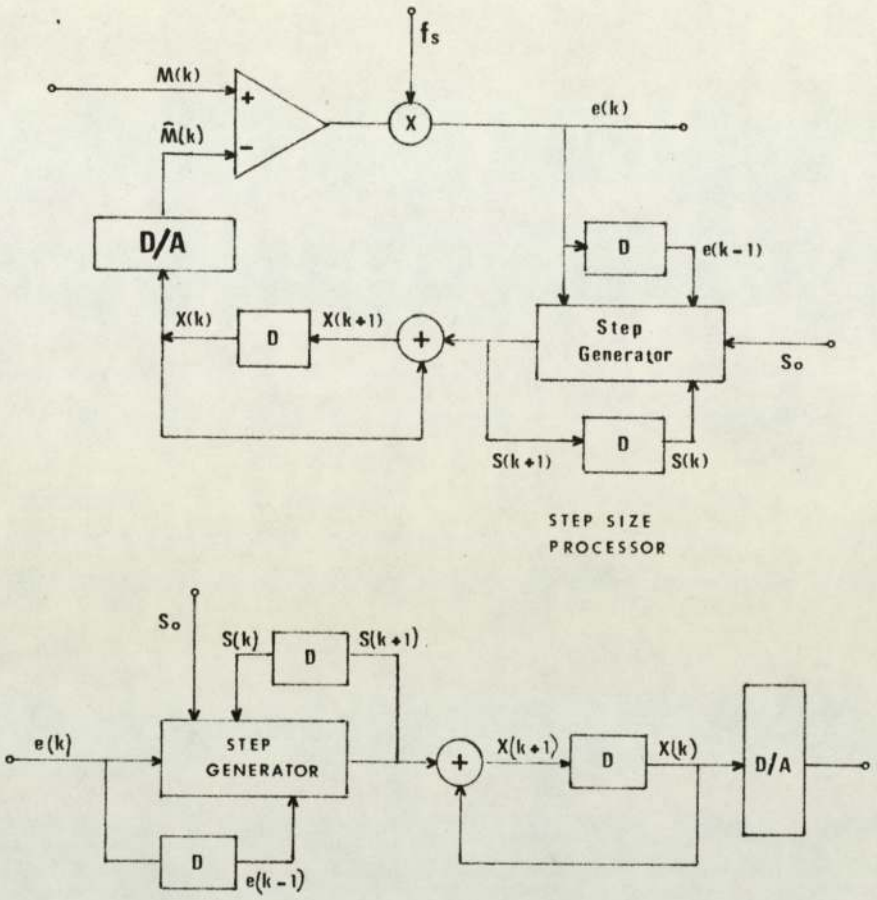


Fig.4.6.-Block diagram for the Song ADM.
(a)encoder, (b)decoder.

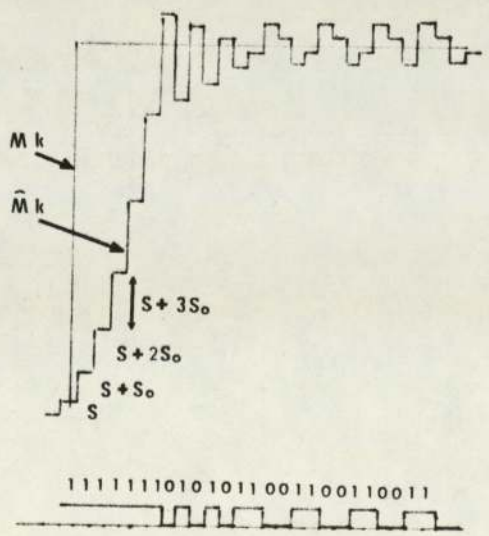


Fig.4.7.-Response of the Song ADM to an
step input. Bit pattern.

$$e(k) = \text{sgn}\left[M(k) - \hat{M}(k)\right] \qquad (4.5)$$

As seen in equation (4.4) the current step size S(k+1) differs, in magnitude, from the previous step size by $\pm S_o$.

Fig.4.7 shows the changing step size and bit pattern generated for an step input. As seen in the figure, when the input signal reaches the steady state, the response of the Song ADM is an estimate signal which exhibits a periodic pattern repeating itself every four samples. The corresponding error pattern generated is ...11001100..., thus, an oscillation with a fundamental frequency of $f_s/4$ is present in the reconstructed output of the Song ADM. The amplitude of the oscillation depends on the step size at the time of the oscillation, but it is usually $2S_o$ when voice signals are encoded. For a band limited speech signal of 300 to 3600 Hz, if the Song operates at f_s = 40 kbit/s, $f_s/4$ = 5khz and therefore within the rejected band. For lower sampling rates, say 20 kbits/s, the effect of the oscillation within the band can be eliminated by using a digital low pass filter at the output of the ADM decoder.

In the actual implementation of the Song system, the maximum signal level was ± 5 V, and the minimum step size with 12 bits of arithmetic (4096 different step-sizes) was 2.4 mV. (S_o = 2.4 mV).

CHAPTER V

THE SOFTWARE FOR THE LINEAR AND SONG ADAPTIVE DELTA MODULATORS

5.1.-Introduction

Software for both linear and Song adaptive delta modulators was developed. The Microcomputer Crossassembly program MCCAP was used to generate the object code for the 8X300. The host computer was a PDP-11/03 with v.d.u. and fast printer facilities.Floppy disk was used for mass storage.

5.2.-Linear Delta Modulator Software

Fig.5.1 shows the block diagram for the linear delta modulator process. The algorithm includes encoding-decoding process with 12-bit arithmetic, clipped output for the estimate signal, and sampling rate selection.

A set of 51 16-bit instructions was needed to implement the linear delta modulator algorithm, although actually only 35 are used in each encoding path. The actual program is enclosed in appendix A.

Dummy instructions were necessary in the algorithm in order to balance each different path and thus keep the sampling period at a steady rate, for whatever action is taken by the processing algorithm. Since the 8X300 is crystal controlled, all the encoding process timing was derived from it via timed software. The time between successive sample outputs to the D/A converter was set by adding up the time taken by the necessary code and then adding wait loops to make up the desired sampling interval.

5.3.-Song Adaptive Delta Modulator Software

The software flow diagram for the Song ADM is shown in Fig.5.2. This is also a 12-bit resolution process with clipped output for the estimate signal, clipped output

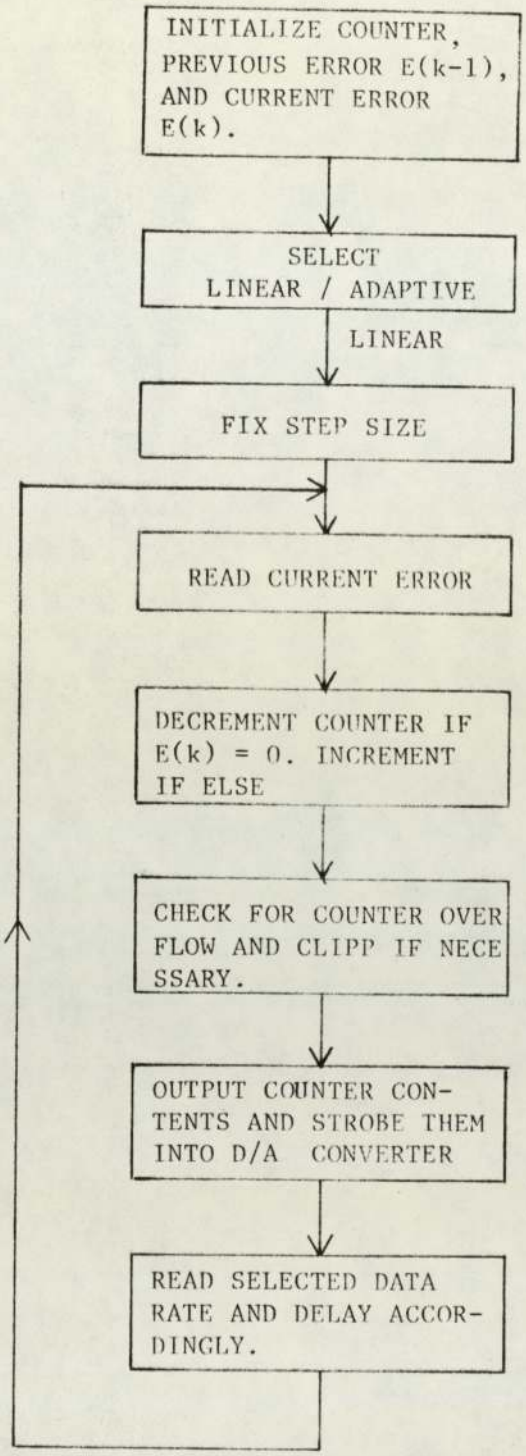


Fig.5.1.-Flow diagram for the Linear Delta Modulation process.

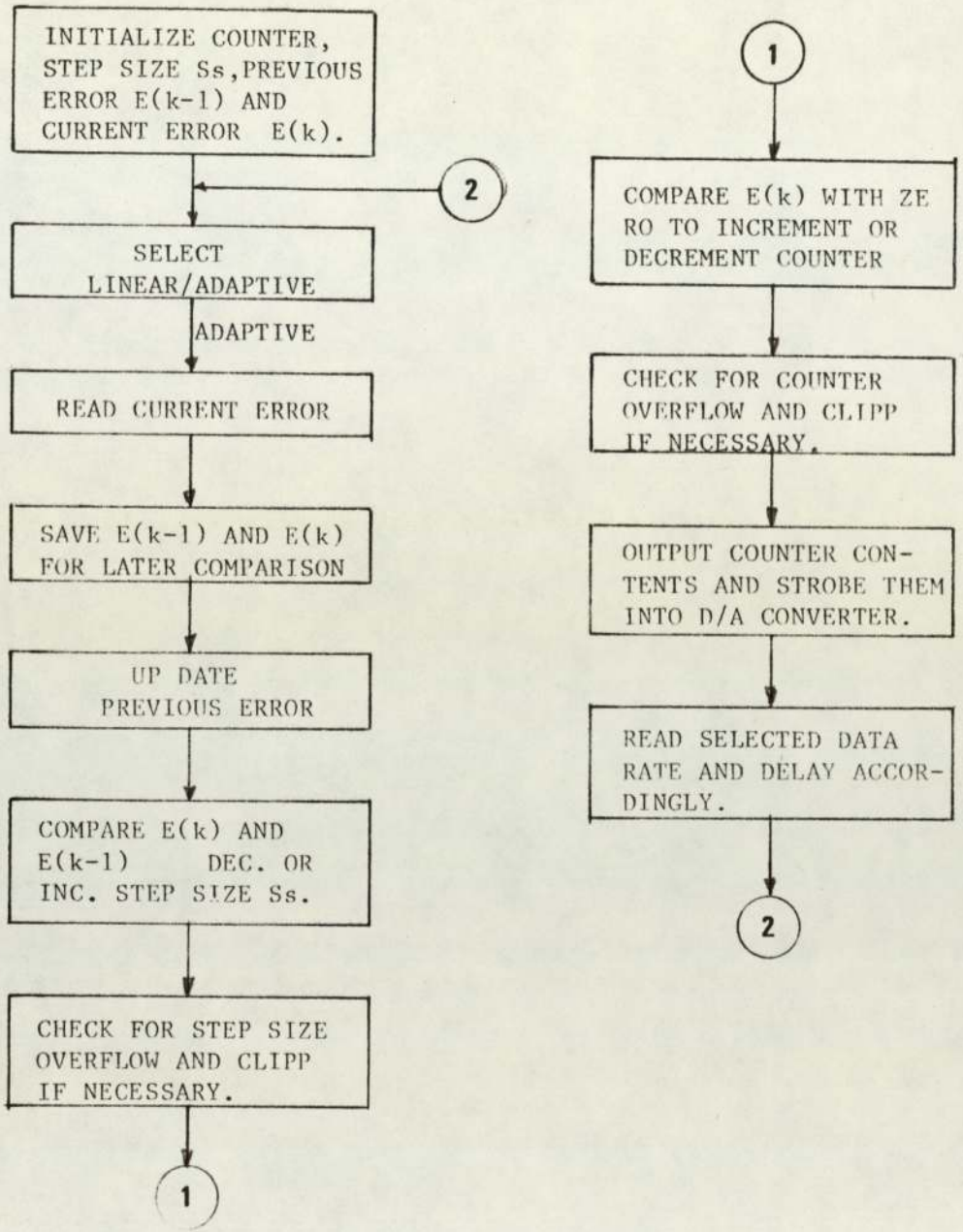


Fig.5.2.-Flow diagram for the Song Adaptive Delta Modulation Process.

for the step size, and selection from 6 different sampling rates, 70, 56, 50, 40, 30 and 20 kbits/s.

A complete code conversion for the Song ADM algorithm is computed entirely in 11.75 microsec. This implies a maximum data rate of 85 kbits/s. For a sampling rate of 20 kbits/s this allows for up to four audio channels to be multiplexed and encoded.

Additional increase in speed can be achieved by using a tabulated step size determination, for instance, but would probably require the addition of memory depending on the maximum step size itself.

The probability density of the step-size S(k+1) for a male voice bandlimited at 2500 Hz and sampled at 32 kbits/s has been experimentally determined(46). The result has shown that for 10-bit arithmetic the most likely step-size is $2S_o$, and it rarely exeeds $64S_o$, this probability being 0.002. For 12-bit arithmetic lower values than these are expected.

Any variable step size ranging from $1S_o$ to $256S_o$, where S_o is a 1 lsb, will need an 8-bit register to hold this this variable. Since this is the case for the registers of the 8X300, a maximum step size of $256S_o$ was chosen for the adaptive algorithm.

Dummy instructions, waiting loops and timed software are also used in the Song adaptive delta modulator software. The actual program is annexed in appendix B.

CHAPTER VI

DELTA MODULATION ENCODING SYSTEM HARDWARE

6.1.-Introduction

This chapter describes the architecture of the delta modulation encoding system, showing the major components of the subunits as well as their interconnecting paths.

The whole of this encoding system has been constructed in the form of two boards, one containing the analogue input unit 710 comparator, the D/A converter, DAC-HK12BGC, algorithm selection switch and data rate selection switches, and the other being the 8X300 microprocessor evaluation board with set of four RAM's added to the wrapped area.

The twelve-bit encoding system is shown in the schematic diagram of Fig. 6.1. Linear or adaptive processing is implemented by means of software and externally selected by a switch. Sampling rate is also selected by a set of three switches using a binary code that the program recognizes.

6.2.-System Operation

The binary error signal resulting from the comparison made between the low pass filtered audio signal and the estimate output from the coder, is sent to I/O port 2, bit 6. According to this error signal, action is taken by the processor to increment or decrement the step size in the adaptive algorithm, and to increment or decrement the accumulator. The contents of the accumulator are then transmitted to I/O ports 1 and 3 as two consecutive data bytes. Port 3 uses all its 8 bits and port 1 uses only its 4 lsb. The 12 bits in I/O ports 1 and 3 are sent to the D/A converter through a set of latches where the two bytes are reassembled as a single word. Clock control for these latches is provided through I/O port

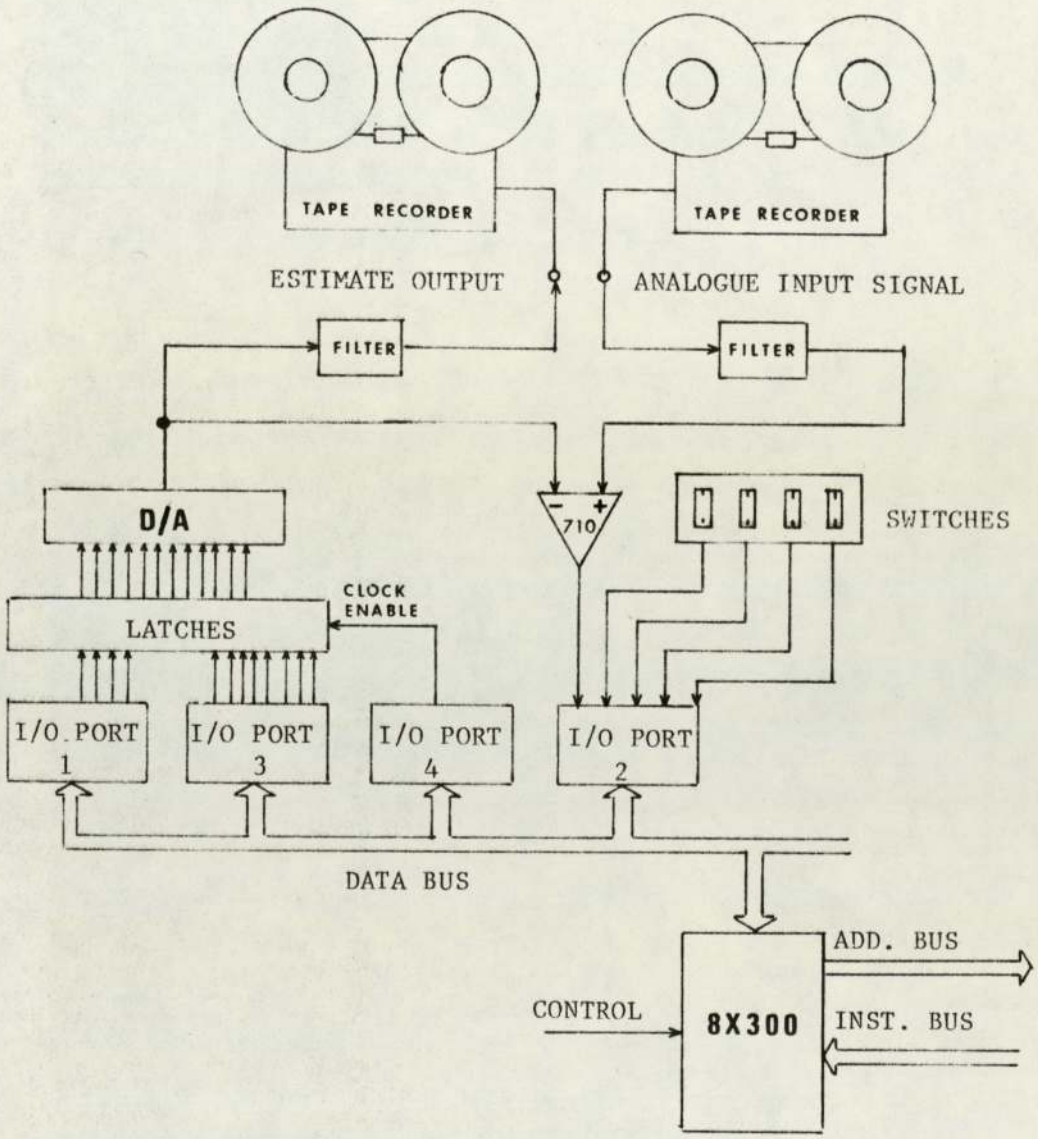


Fig.6.1.-Schematic diagram of the Delta Modulation encoding system, using the 8X300 microprocessor. (12-bit resolution).

4, bit 0, and derived by timed software.

The output signal from the D/A converter is passed through a reconstruction filter to smooth out discrete changes in the slope of the estimate signal and remove out-of band quantization noise. The filtered signal is then stored in an audio tape.

The estimate signal from the D/A converter is also sent to the 710 comparator for a new process.

The original speech is derived from a tape recorder and then undergoes an appropiate amount of bandlimiting following the real time acquisition and coding process.

An alternative way of doing the job is converting the incoming analogue signal to a digital format by using an A/D converter. The whole process is then performed digitally, ie, the microprocessor will generate the error signal instead of the external comparator. Fig.6.2 shows the block diagram for a delta modulation encoding system of 8-bit resolution. For a 12-bit resolution encoding system, it will be necessary to have more I/O ports than the encoding system of Fig.6.1, to input and output data and control signals, and also to include a few more instructions to implement the algorithm.

As seen in Fig.6.2, I/O port 1 is set as a dedicated input only. For this, the control bits BIC (Bit Input Control) has to be low and BOC (Bit Output Control) has to be high. The eight bits output from the A/D converter are taken through this port.

Port 2 is a CPU controlled bidirectional I/O.

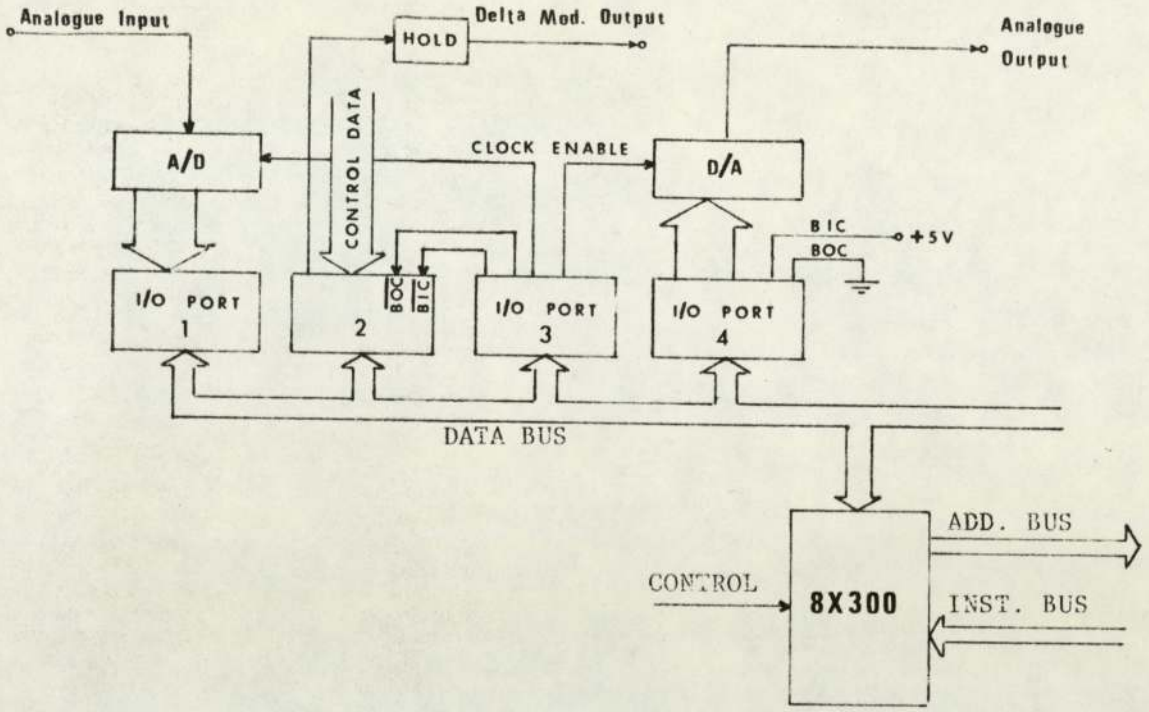


Fig.6.2.-Delta Modulation Encoding System using digitized input.(8-bit resolution).

Control data can be used to select a processing algorithm, bit data rate, minimum and maximum step size, etc. The digital output from the delta modulator is also sent through this port.

Ports 3 and 4 are dedicated output only. Port 4 sends to the D/A converter the processed 8-bit byte to form the analogue estimate output from the delta modulator.

CHAPTER VII

EXPERIMENTAL RESULTS

7.1.-Introduction

The performance of the actual real-time microprocessor-based encoding system is presented in the following section. Several photographs were taken to show the system performance capabilities, and an informal subjective test was performed to have a rough idea of the speech quality obtained through this encoding system.

7.2.-Linear Delta Modulator Perfomance

Fig. 7.1 shows the tracking of the linear delta modulator to an 800 Hz sinusoidal input being sampled at 70 and 40 kbits/s. It is seen that for this 2V p-p input signal the encoder is correctly tracking it. The step size is in this case 125 mV, ie, 50 times greater than the step for the least significant bit for a 12-bit $\pm$5V D/A conversion. This step size has been made on purpose in order to see the staircase-like waveform result of the encoding-decoding process.

Slope overload distortion depends on the slope of the input signal rather than just on the signal amplitude. This effect can be seen in Fig. 7.2(a) and (b), where the frequency of the input signal has been raised to 1800 Hz. Fig.7.2 (a) shows the slope overload effect for a 4V p-p sinewave input with frequency of 1800 Hz, sampling rate of 70 kbits/s, and step size of 125 mV. In (b) the sinewave amplitude has been reduced to 2V p-p and the sampling rate to 30 kbits/s. It should be noted that two factors of Eq. (4.2) have been reduced simultaneously by approximately the same amount.

Fig. 7.2(c) shows the steady state or idling state estimate output of the linear delta modulator for 0V. in-

put, 50 kbits/s sampling rate and step size of 250 mV.

7.3.-Adaptive Delta Modulator Perfomance

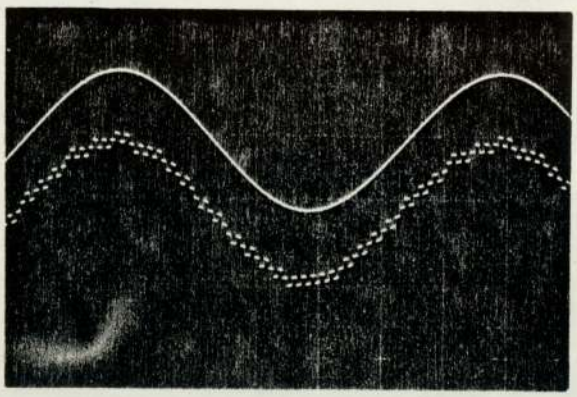
Fig. 7.3 shows the tracking of the Song ADM to a 3V p-p 800 Hz sinewave input at bit rates of 70, 56, 40, 30, and 20 kbits/s. In Fig. 7.3(e), slope overload becomes apparent for the sampling rate of 20 kbits/s, and the tracking is much better when the signal amplitude is reduced to 1.4V p-p as in Fig. 7.3(f).

The response of the system to a 200 Hz square wave input is shown in Fig. 7.4, for sampling rates of 70 and 20 kbits/s, and input signal amplitude of 1V.

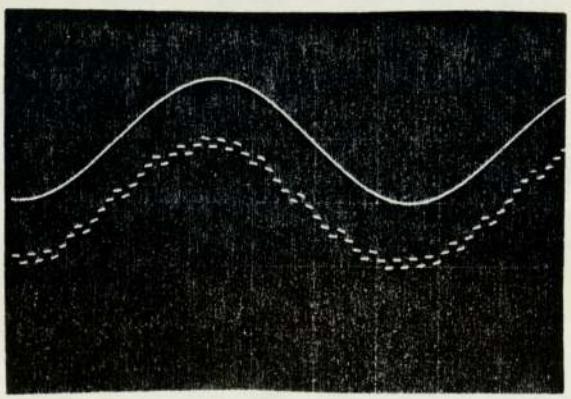
Voice signals were also input to the encoding system and photographs taken from the real time process. In Fig. 7.5, the upper trace shows the filtered audio input, the middle trace the estimate output from the delta modulator and the lower trace the estimate output after being filtered. The sampling frequencies from (a) through (d) are 56, 56, 40 and 20 kbits/s respectively. Clipping effect is seen in (b) where the amplitude of the input signal is greater than $\pm$5V.

7.3.1.-Subjective Evaluation

It has to be stressed here that the following evaluation of the speech encoding system is not intended to constitute a formal one, but to give a rough idea of the speech quality at various speed rates. This is because it was not possible to fulfill all the prerequisites necessary for a formal and total speech quality evaluation. Among other reasons, the test was carried out with a reduced listening group with a limited time available to be trained. A group of un-

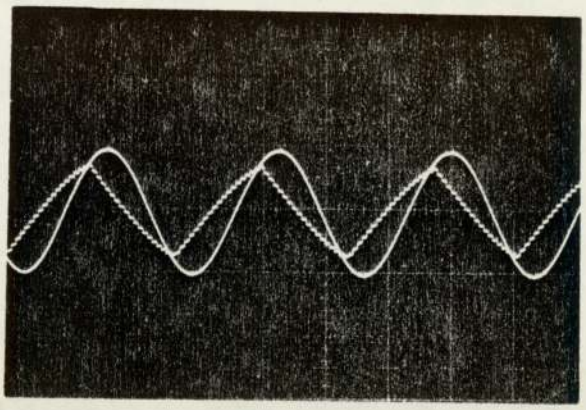


(a) f_m = 800 Hz., f_s = 70 Kbits/s., 1V/div

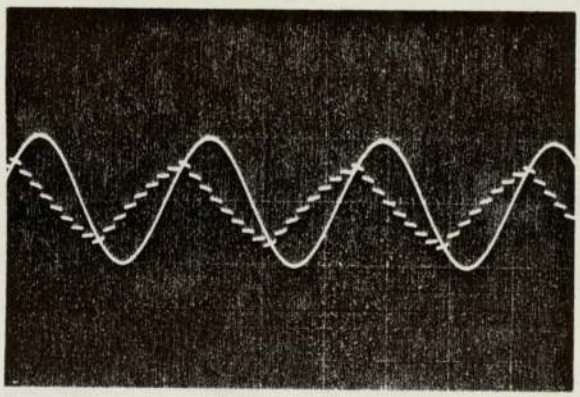


(b) f_m = 800 Hz., f_s = 40 Kbits/s., 1V/div

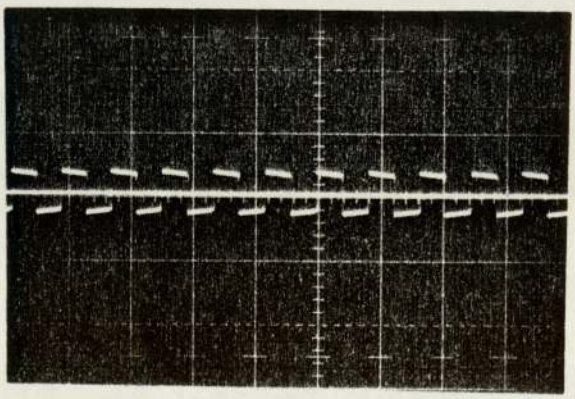
Fig.7.1.-Tracking of the linear delta modulator to a 800 Hz. sinewave input



(a) f_m = 1800 Hz., f_s = 70 Kbits/s., 2V/div.

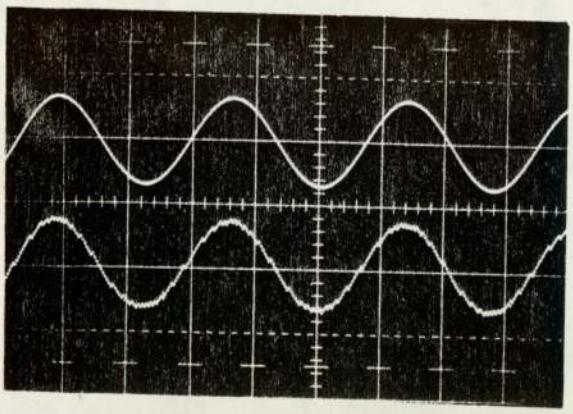


(b) f_m = 1800 Hz., f_s = 30 Kbits/s.. 1V/div.

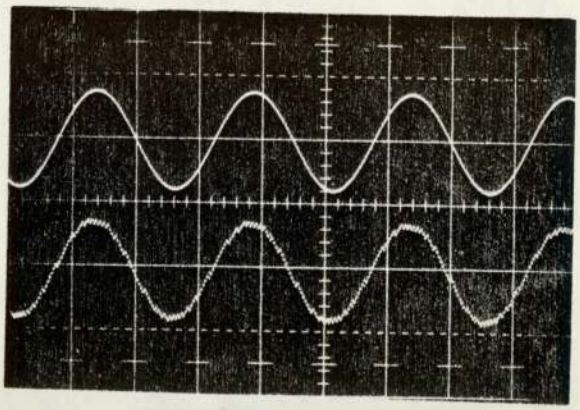


(c) zero input, f_s = 50 Kbits/s., 0.2V/div.

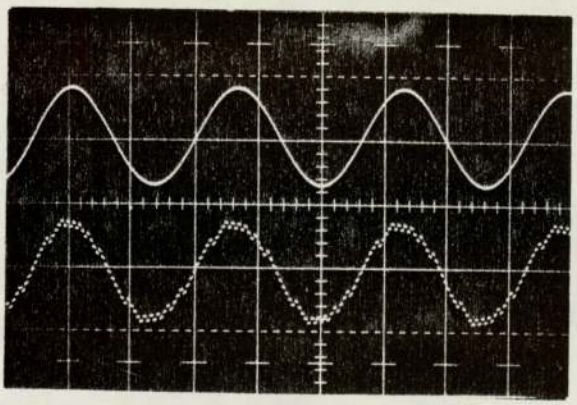
Fig.7.2.-(a) and (b) shows slope overload distortion of the LDM. (c)idling state output.



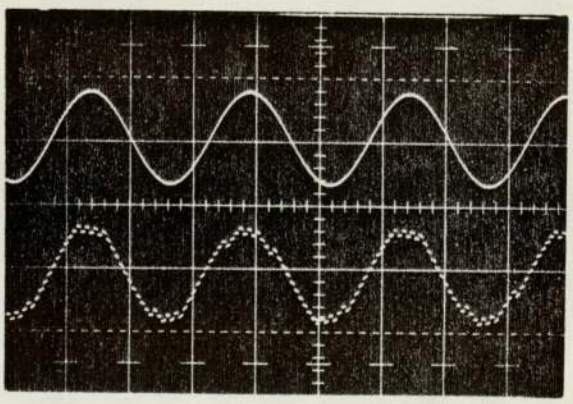
(a) f_s = 70 Kbits/s.



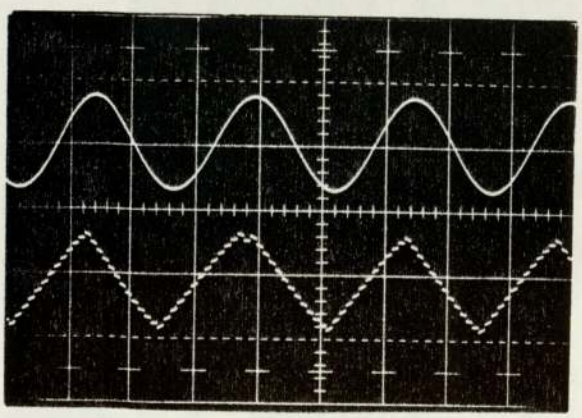
(b) f_s = 56 Kbits/s.



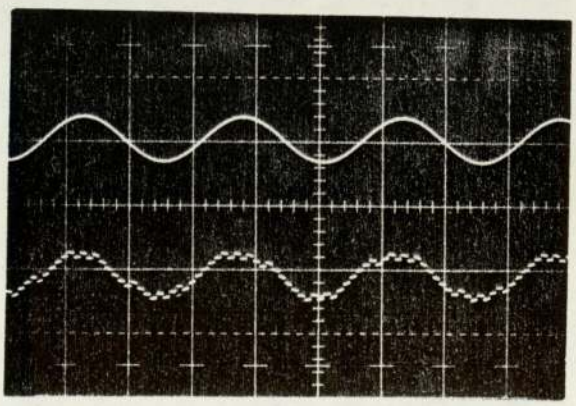
(c) f_s = 40 Kbits/s.



(d) f_s = 30 Kbits/s.

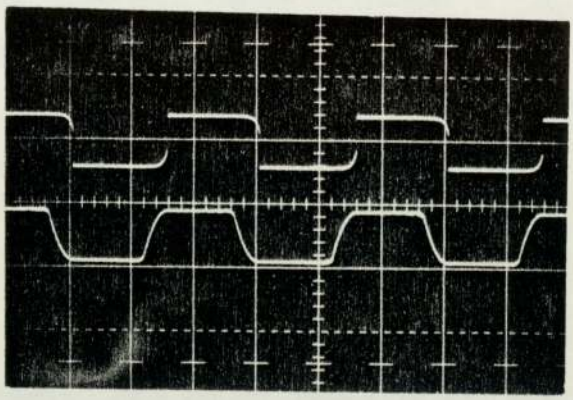


(e) f_s = 20 Kbits/s.

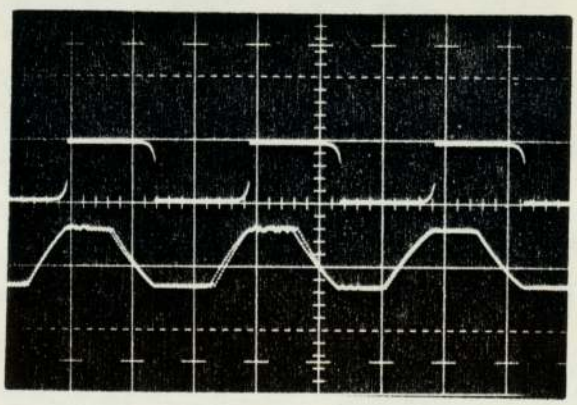


(f) f_s = 20 Kbits/s.

Fig.7.3.-Tracking of the adaptive algorithm to a 800 Hz sinewave input at different sampling rates. Upper trace sinewave input. (2V/div.)

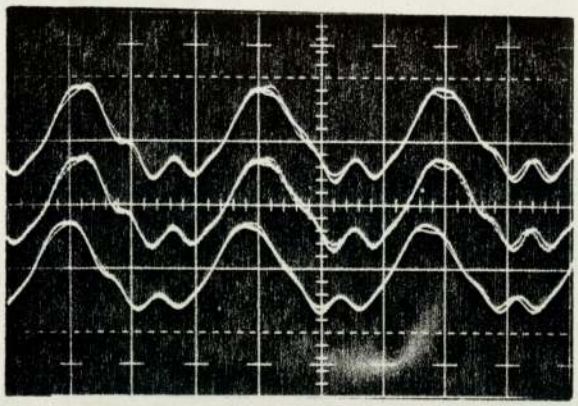


(a) 1V/div., f_s = 70 Kbits/s.

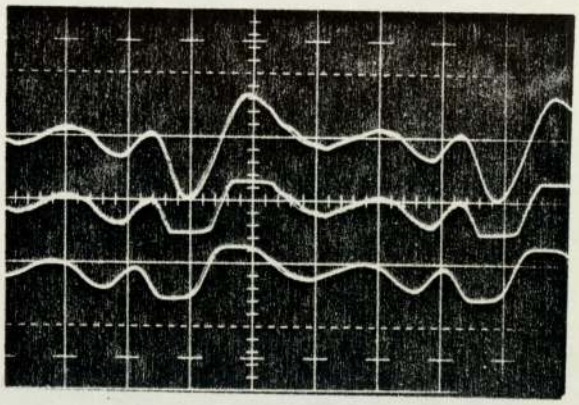


(b) 1V/div., f_s = 20 Kbits/s.

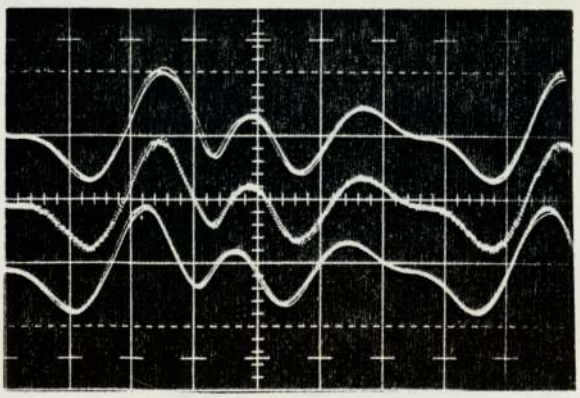
Fig.7.4.-Response of the adaptive algorithm to a 200 Hz. square wave input. Upper trace input signal. Lower trace, encoder output.



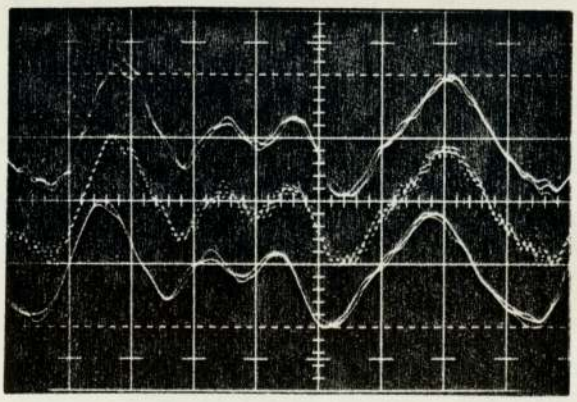
(a) 1 msec/div., 2V/div.
f_s = 56 Kbits/s.



(b) 1 msec/div., 10 V/div.
f_s = 56 Kbits/s.
clipped output.



(c) 1 msec/div., 2V/div.
f_s = 40 Kbits/s.



(d) 1 msec/div., 2 V/div.
f_s = 20 Kbits/s.

Fig.7.5.-Response of the adaptive delta modulator to voice at various sampling rates. Clipping effect is shown in (b). Upper trace is the filtered audio input. Lower trace is the filtered DM output.

trained listeners probably should consist of no less than 50 listeners(2). For the same reason, the test was realized in only one run. It is advisable to repeat the test several times with one-week intervals, to have time-independent results. There was a lack of number of sentences as to avoid repetition of them during test and the number of methods was limited to two.

The IEEE Subcommittee on Subjective Measurements has written an engineering practice(2) that describes several preferent measurement methods for the measurement of speech quality. For the convenience of the reader, Appendix C contains a short description of these methods.

7.3.1.1.-Speech Material and Listeners

The speech material used for the tests consisted of three phonetically balenced sentences, 'It's easy to tell the depth of a well, These days a chicken leg is a rare dish, A large size in stocking is hard to sell'. These sentences were spoken by a male speaker using British accent English. The sentences were encoded using the system of Fig. 6.1, at bit rates of 70, 56, 50, 40, 30, and 20 kbits/s, and recorded on a single track tape at $7\frac{1}{2}$ in/s.

To be used as a source of comparison, a second set of coded speech signals were included in the tape. This speech material consisted of the sentence 'Grab every dish of sugar' coded also through an adaptive delta modulation system at the bit rates of 32 and 16 kbits/s. This speech material was taken from a record prepared for a demostration of speech coding systems(3). The voice recorded used American accent English and a male speaker.

The test tape was presented to twelve listeners, professional and student, through headsets in a languages laboratory room. All of the listeners were native English speakers and none had any previous experience in subjective testing of speech material. Their ages renged from 20 to 44 with mean age slightly over 30. Nine of them used American accent English and seven of them were male.

7.3.1.2.-Subjective Evaluation Procedure

The evaluation was started with a direct forced-pairs comparison of two sentences. The pairs were formed by the sentence 'It's easy to tell the depth of a well' coded through the system being evaluated, and the sentence 'Grab every dish of sugar' from the record. These sentences will be known here as sentence (a) and sentence (b), respectively. Sentence (a) was presented at the bits rates of 70, 56, 50, 40, 30, and 20 kbits/s against two bit rates, 32 and 16 kbits /s of sentence (b). In all there were 12 combinations presented randomly to the listeners.

Prior to this session, the listeners were given the following instructions.

'In this test you will hear pairs of sentences (A,B) repeated twice, according to the pattern shown in the following figure.

A B A B	5 to 7 seconds rest period	A B A B	5 to 7 seconds rest period	A B A B	5 ...

As you can see from the pattern, each set of sentences ABAB is separated by a 5 to 7 seconds rest period. After listening to the repeated pairs, specify which sentence you would pre-

fer to hear (A or B) marking it with an "X" in the appropiate box below. This must be done during each rest period. If both sentences sound equally good, make an arbitrary choice. The first and third sentences in the pattern is the same sentence "A", and the second and fourth, the same sentence "B".'

After this initial step an absolute method, the Mean Category Judgement method was utilized to evaluate the quality of the speech samples. This time, only the material processed by the system under evaluation was used. The test was done in two runs using the following sentences: 'These days a chicken leg is a rare dish' and 'A large size in stocking is hard to sell'.

Prior to this session, the listeners were given the following written instructions.

'For this test you will not hear pairs of sentences as in the former test, but single sentences separated also by rest periods of 5 to 7 seconds.

This time you will specify the quality of each sentence by chosing one of the categories on the scale, Excellent, Good, Fair, Poor and Unsatisfactory and marking it with an "X" in the appropiate box below.

You are going to hear an example of an "Excellent" sentence and an "Unsatisfactory" sentence. The pair will be repeated. Use them as points of reference for your judgements'.

There were in total 12 samples to be evaluated, 6 for each sentence corresponding to 6 different bit rates.

For an excellent sentence, the original sentence was used as recorded from the speaker without undergoing any

coding process and filtering. For an unsatisfactory sentence the low-pass filtered original sentence was used after being re-recorded several times on a low quality cassette tape at high recording level, plus some low-pass filtered white noise added to it. The resultant sentences although still recogniza ble, had a very annoying noise-like structure.

7.3.1.3.-Results and Discussions of subjective evaluations

Table 7.1 shows the results for the direct comparison test between signal (a) (6 bit rates), and signal (b)(2 bit rates). The first and third rows in Table 7.1 show the number of listeners who voted favourable to signal (a), and the second and fourth rows the same but in terms of percentages.

Table 7.1.-Preference for direct comparisons (a,b)

signal (b) \ signal (a)	70 kbits/s	56 kbits/s	50 kbits/s	40 kbits/s	30 kbits/s	20 kbits/s
32 kbits/s	8	8	9	8	9	8
	66.6%	66.6%	75.0%	66.6%	75.0%	66.6%
16 kbits/s	9	10	8	9	10	9
	75.0%	83.3%	66.6%	75.0%	83.3%	75.0%

Fig. 7.6 shows the percentage of listeners who preferred the test sample signal (a) to signal (b), against the sampling bit rate of signal (a) and sampling bit rate of signal (b) as a parameter.

The results from Fig. 7.6 indicate that the speech

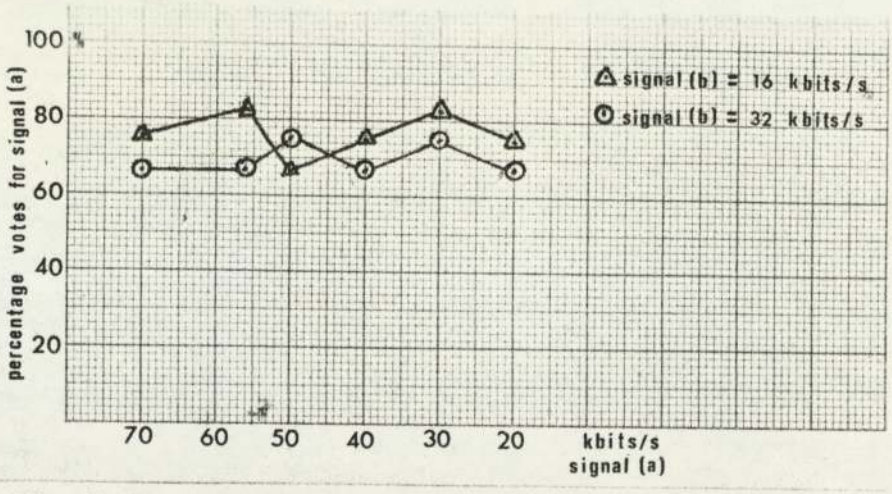


Fig.7.6.-Direct comparison test

samples of signal (a) are consistently better than those of signal (b), from the record. The listeners showed a clear preference to these even for signal (a) at 20 kbits/s, but the results do not appear to be very discriminable at the different bit rates of signal (a). The expected results should have shown a negative slope in both traces, indicating lesser preference for signal (a) as the bit rate was decreasing toward 20 kbits/s.

It is not known whether the record had been recorded by means of a high-fidelity process. In fact, the volume level of the sentences taken from the record was too low and had to be elevated during recording on tape, to make it comparable to that of signal (a). This obviously implied a rise in the inherent noise in records, such as surface and static electricity noise. It is also difficult to judge the effect on the listeners of the difference in tone and accent of the two speakers.

An additional feature that can be seen in Fig.7.6 is that for signal (b) sampled at 16 kbits/s, the percentage of votes favourable to signal (a) is actually slightly higher than those for signal (b) sampled at 32 kbits/s. This repre-

sents an expected result since signal (b) sampled at 16 kbits /s presented a more noticeable noise degradation, and thus received less favourable votes.

Fig. 7.7 summarizes the results of the tests of the two sentences evaluated by the Mean Category Judgment method. This Fig. shows the overall rating of the answers for each bit rate (vertical bars), and the Mean Opinion Score(MOS) for each bit rate (dot within the bars).

The Mean Opinion Scores is calculated in the following way. Each category used in the test is given a number,

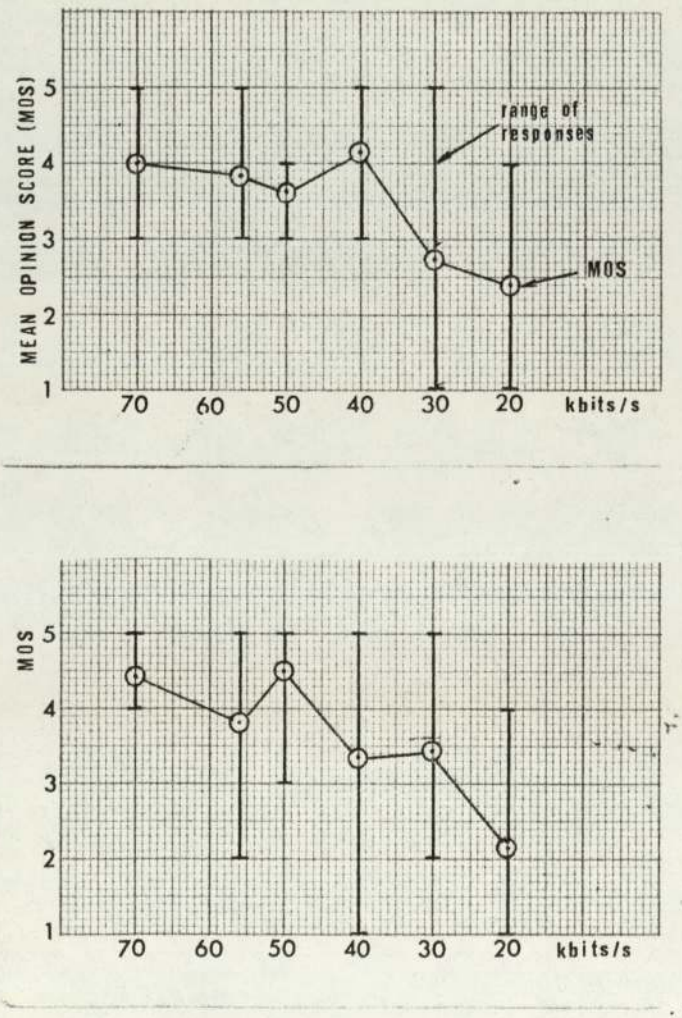


Fig. 7.7.-Mean opinion scores against bit rate. (a) first sentence, (b) second sentence.

e.g., Unsatisfactory = 1, Poor = 2, Fair = 3, Good = 4, and Excellent = 5. The number of listeners choosing each category is multiplied by the number assigned to that category. These totals are then added, and divided by the total number of listeners. The number then obtained is the mean opinion score (MOS) for that condition(2).

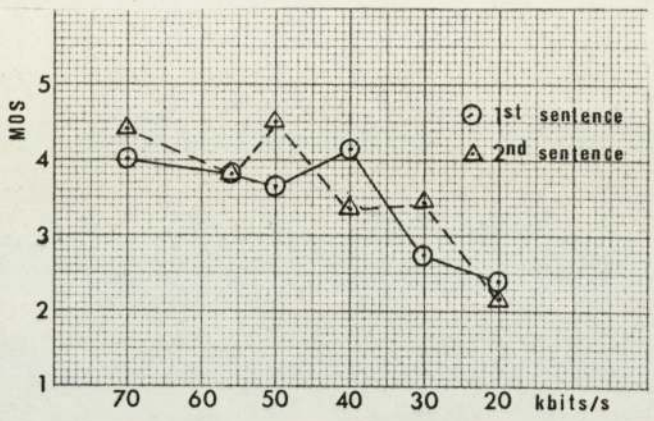


Fig. 7.8.-Mean opinion scores for both sentences

Fig. 7.8 shows the MOS for both sentences and it can be seen that the subjective quality at the different bit rates are quite close to each other. In both cases, the decrease in the quality of the speech samples when going from 70 kbits/s through 20 kbits/s is quite evident.

Fig. 7.9 presents the mean taken over the mean opinion scores of Fig. 7.8. It can be seen that the slope of the trace is steeper from 40 kbits/s to 20 kbits/s, indicating a more rapid decrease in the relative quality of the speech.

It can be said that the results obtained by the mean opinion score method are more significant than the re-

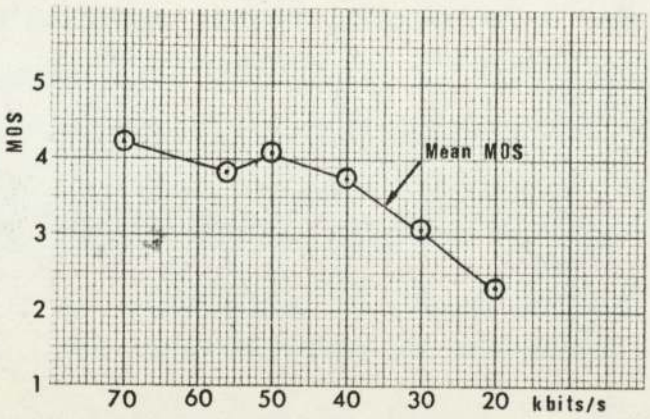


Fig. 7.9.-Mean over the mean opinion scores.

sults obtained by the direct comparison method. This is because for the direct comparison method there were serious limitations in controlling some factors such as tone, volume, noise other than that produced by the encoding process, and as it was pointed out earlier, it is difficult to judge the effect on the listeners of the difference in tone and accent of the two speakers. It is also not known the type of algorithm used in the adaptive delta modulator presented in the record.

CONCLUSIONS

A real-time microprocessor-based speech encoding system has been implemented. The system is based on the Signetics 8X300 microprocessor evaluation board which features high speed of processing.

Delta modulation was used as the encoding scheme, and the adaptive algorithm used for the delta modulator was the Song algorithm, which proved to be very easy to implement by means of software.

The development of this software controlled encoding system has been supported by a PDP-11/03 minicomputer, and the MCCAP crossassembler.

The resulting system is easy to program and can be used not only for real-time signal processing purposes but also for decision and control applications or combinations of these. Different programs can be stored in memory and be selected according to an external code or interrupting routine.

The entire microprocessor-based delta modulation encoding system was found to be comparatively simple to construct and useful for practical applications.

In general, the system is sufficiently flexible to allow experimentation and development of a particular codec algorithm. As this device operates under software control, this feature allows for making as much modification as required during development at almost no delay or expense. Instantaneous parameter alteration is easily achieved by program control and has the potential to incorporate many different code conversions within the same harware facility.

A complete code conversion for the adaptive delta modulator actually implemented is computed entirely in 11.75 usec. This allows for up to four audio channels to be multiplexed and encoded at a sampling rate of 20 kbits/s. Data rates as high as 85 kbits/s were possible for the adaptive algorithm. This means that there is room for more elaborate encoding algorithms with adequate output data rates.

Since the microprocessor is crystal controlled, it was convenient to derive all the encoding process timing from it via timed software. The time between successive sample outputs to the D/A converter was set by adding up the time taken by the necessary code and then adding wait loops to make up the desired ampling interval. As there were multiple data-dependent paths through the code, the execution times of these were also made equal by adding dummy instructions.

Non-rigorous tests or evaluation of the proposed design has been carried out, but some graphical results are presented as an illustration of the capabilities of the system. One of the most important outputs obtained from this encoding system is an audio tape containing speech which has been processed at various speed rates.

Results and discussions from an informal subjective test are presented in sub-section 7.3.1.3. These results have indicated the following.

(a)After a direct forced-pair comparison between speech samples processed by the system under evaluation and speech samples processed by another system also based on adptive delta modulation, the listeners showed a clear preferen-

ce (70% in average) to the former, even at sampling rates of 20 kbits/s against 32 kbits/s. Nevertheless, it has to be said that the evaluation conditions were not quite the same for both systems. Particularly, the recording volume of the sentences taken from the record were quite low, and these were spoken with a different accent and tone from those from the system being evaluated. It has also to be pointed out that of a group of twelve listeners, nine of them used the same accent as that used in the record.

(b) From the subjective evaluation by the Mean Category Judgment method, the speech processed by the system under evaluation, in the mean, has been categorized within the following levels. Speech encoded at 70, 56, 50 and 40 kbits/s, as 'good'. Speech encoded at 30 kbits/s, as 'fair' and speech encoded at 20 kbits/s, slightly better than 'poor'. The complete range of response categories being, excellent, good, fair, poor, and unsatisfactory.

To have a more complete view of the perfomance of the microprocessor-based delta modulator, objective measurements should be carried out. The objective measurements should include the following. Bandwidth (Output level Vs. input frequency), for different input levels and sampling frequencies. Idle Channel Noise below the maximum input signal considering channel errors and considering no channel errors. A C-message weighted filter should be used at the output of the coder. Dynamic Range using a single tone (SNR Vs. input level at various bit rates). SNR Vs. Bit Rate. Linearity (Variation of the output to input amplitude ratio).

APPENDIX A

(Linear Delta Modulator Program)

```
                        ************************************
                        *  THIS PROGRAM WILL SIMULATE A    *
                        *  LINEAR DELTA MODULATOR WITH A   *
                        *  BIGGER STEP SIZE THAN ONE LSD   *
                        *  THIS STEP SIZE IS INTENDED TO   *
                        *  SHOW THE STAIRCASE-LIKE WAVE-   *
                        *  FORM RESULT OF THE ENCODING-DE  *
                        *  CODING PROCESS                  *
                        *  SIX BIT RATES ARE AVAILABLE     *
                        *  BY EXTERNAL SWITCHES REQUEST    *
                        *  70, 56, 50, 40, 30, AND 20      *
                        *  KILOBITS-SECOND                 *
                        ************************************


                        ************************************
                        *  THE PROCESSOR IS THE SX300      *
                        ************************************


                              PROG  LINBSS


                        ************************************
                        * DATA AND ADDRESS DECLARATIONS    *
                        ************************************

          000001        INC   EQU  1
          000377        DEC   EQU  377H
        001 7 0         OUT1 LIV  1,7,8
        002 6 1         IN2 LIV  2,6
        002 5 3         PORT2 LIV 2,5,3
        002 2 1         SWIT2 LIV 2,2
        003 7 0         OUT3 LIV  3,7,8
        004 7 0         OUT4 LIV  4,7,8



                        ************************************
                        *     MAIN PROGRAM                 *
                        ************************************

                              ORG   0
00000   6 01000         LINEAR  XMIT  0,R1
00001   6 02010               XMIT 10H,R2
                        *COUNTER HAS BEEN INITIALIZED TO ZERO OFFSET
                        ********************************************
00002   6 07004               SEL   OUT4
```

```
44  00003  6 27001          XMIT  1,OUT4
45                       *REGISTERS OF THE D TO A IN HOLD DATA STATE
46                       ******************************************
47  00004  6 03304          XMIT 304H,R3
48  00005  6 07002      HERE SEL SWIT2
49  00006  0 22106          MOVE  SWIT2,R6
50  00007  6 00001          XMIT  1,AUX
51  00010  3 06000          XOR   R6,AUX
52  00011  5 00013          NZT   AUX,BASIC
53  00012  7 00170          JMP   ADAPTV
54  00013  6 07002      BASIC SEL   IN2
55  00014  0 26105          MOVE  IN2,R5
56  00015  5 05103          NZT   R5,NONULL
57                       *IS CURRENT ERROR  =  0 ?
58                       ************************
59  00016  6 00017          XMIT  17H,AUX
60  00017  2 02002          AND  R2,R2
61                       *PLACE ALL 0'S IN NON-USED BITS OF R2
62                       ************************************
63  00020  6 00377          XMIT DEC,AUX
64  00021  3 03006          XOR R3,R6
65  00022  6 00001          XMIT INC,AUX
66  00023  1 06000          ADD R6,AUX
67  00024  1 01001          ADD R1,R1
68  00025  0 10011          MOVE OVF,R11
69  00026  6 00377          XMIT DEC,AUX
70  00027  1 10000          ADD OVF,AUX
71  00030  1 02002          ADD R2,R2
72                       *DECREMENT COUNTER BY STEP SIZE
73                       ******************************
74  00031  5 10036          NZT OVF,STILL
75  00032  5 11036          NZT R11,STILL
76                       *IS COUNTER < TO ZERO TO CLIP IT ?
77                       *********************************
78  00033  6 01000          XMIT 0,R1
79  00034  6 02000          XMIT 0,R2
80  00035  7 00042          JMP OUT
81  00036  6 11000      STILL XMIT 0,R11
82  00037  6 11000          XMIT 0,R11
83  00040  6 11000          XMIT 0,R11
84  00041  6 11000          XMIT 0,R11
85  00042  6 07003      OUT    SEL   OUT3
86  00043  0 01027          MOVE  R1,OUT3
87  00044  6 07001          SEL   OUT1
88  00045  0 02027          MOVE  R2,OUT1
89  00046  6 07004          SEL   OUT4
90  00047  6 27000          XMIT  0,OUT4
91  00050  6 27001          XMIT  1,OUT4
92                       *OUTPUT COUNTER CONTENTS AND STROBE THEM INTO D TO A
93                       ***************************************************
94  00051  6 07002           SEL PORT2
95  00052  0 25311      AGAIN MOVE PORT2,R11
96  00053  6 00000          XMIT  0,AUX
97  00054  3 11000          XOR  R11,AUX
```

```
 98  00055  5 00057          NZT  AUX,NEX56
 99  00056  7 00127          JMP  DATA70
100  00057  6 00001  NEX56   XMIT  1,AUX
101  00060  3 11000          XOR  R11,AUX
102  00061  5 00063          NZT  AUX,NEX50
103  00062  7 00134          JMP  DATA56
104  00063  6 00002  NEX50   XMIT  2,AUX
105  00064  3 11000          XOR  R11,AUX
106  00065  5 00067          NZT  AUX,NEX40
107  00066  7 00142          JMP  DATA50
108  00067  6 00003  NEX40  XMIT  3,AUX
109  00070  3 11000          XOR  R11,AUX
110  00071  5 00073          NZT  AUX,NEX30
111  00072  7 00147          JMP  DATA40
112  00073  6 00004  NEX30  XMIT  4,AUX
113  00074  3 11000          XOR  R11,AUX
114  00075  5 00077          NZT  AUX,NEX20
115  00076  7 00155          JMP  DATA30
116  00077  6 00005  NEX20  XMIT  5,AUX
117  00100  3 11000          XOR  R11,AUX
118  00101  5 00052          NZT  AUX,AGAIN
119  00102  7 00162          JMP  DATA20
120  00103  6 00360  NONULL  XMIT 360H,AUX
121  00104  3 02004           XOR   R2,R4
122  00105  2 02000           AND   R2,AUX
123  00106  3 04002           XOR   R4,R2
124                         *PLACE ALL 1'S IN NON-USED BITS OF R2
125                         ************************************
126  00107  0 03000          MOVE R3,AUX
127  00110  1 01001           ADD   R1,R1
128  00111  0 10000           MOVE  OVF,AUX
129  00112  1 02002           ADD   R2,R2
130  00113  6 11000          XMIT 0,R11
131  00114  6 11000          XMIT 0,R11
132  00115  6 11000          XMIT 0,R11
133                         *INCREMENT COUNTER BY 1 LSB
134                         ***************************
135  00116  5 10123           NZT   OVF,YES
136                         *HAS THE COUNTER OVERPASS 7777H  VALUE ?
137                         ***************************************
138  00117  6 11000           XMIT  0,R11
139  00120  6 11000           XMIT  0,R11
140  00121  6 11000          XMIT  0,R11
141                         *2 INSTRUCTIONS DELAY TO MAKE PATHS EQUAL
142                         ****************************************
143  00122  7 00042           JMP   OUT
144                         *TO OUTPUT COUNTER CONTENTS
145                         ***************************
146  00123  6 01377  YES   XMIT  377H,R1
147  00124  6 02377           XMIT  377H,R2
148  00125  6 11000          XMIT 0,R11
149                         *CLIP COUNTER TO 7777H AND DELAY BY 2 INSTR.
150                         ********************************************
151  00126  7 00042           JMP   OUT
```

```
152  00127  6 06367    DATA70 XMIT  367H,R6
153  00130  6 00001           XMIT  1,AUX
154  00131  1 06006    RING1 ADD   R6,R6
155  00132  5 06131           NZT   R6,RING1
156  00133  7 00005           JMP   HERE
157  00134  6 06362    DATA56  XMIT 362H,R6
158  00135  6 00001           XMIT  1,AUX
159  00136  1 06006    RING2 ADD   R6,R6
160  00137  5 06136           NZT   R6,RING2
161  00140  6 11000           XMIT  0,R11
162  00141  7 00005           JMP   HERE
163  00142  6 06357    DATA50 XMIT 357H,R6
164  00143  6 00001           XMIT  1,AUX
165  00144  1 06006    RING3 ADD   R6,R6
166  00145  5 06144           NZT   R6,RING3
167  00146  7 00005           JMP   HERE
168  00147  6 06347    DATA40 XMIT 347H,R6
169  00150  6 00001           XMIT  1,AUX
170  00151  1 06006    RING4 ADD   R6,R6
171  00152  5 06151           NZT   R6,RING4
172  00153  6 11000           XMIT  0,R11
173  00154  7 00005           JMP   HERE
174  00155  6 06327    DATA30 XMIT 327H,R6
175  00156  6 00001           XMIT  1,AUX
176  00157  1 06006    RING5 ADD   R6,R6
177  00160  5 06157           NZT   R6,RING5
178  00161  7 00005           JMP   HERE
179  00162  6 06267    DATA20 XMIT 267H,R6
180  00163  6 00001           XMIT  1,AUX
181  00164  1 06006    RING6 ADD   R6,R6
182  00165  5 06164           NZT   R6,RING6
183  00166  6 11000           XMIT  0,R11
184  00167  7 00005           JMP   HERE
185                     *TO OUTPUT COUNTER CONTENTS
186                     ****************************
187  00170  6 01000    ADAPTV XMIT  0,R1
188                           END   LINBSS


TAL ASSEMBLY ERRORS =    0
 --

IP
```

APPENDIX B

(Song Adaptive DM Program)

```
PROG  LISONG                    MICROCONTROLLER CROSS ASSEMBLER VER 1.1.

 1                      ************************************
 2                      *  THIS PROGRAM WILL SIMULATE A    *
 3                      *  LINEAR DELTA MODULATOR AND AN   *
 4                      *  ADAPTIVE DELTA MODULATOR USING  *
 5                      *  THE SONG ALGORITHM. BOTH DELTA  *
 6                      *  MODULATOR ALGORITHMS ARE 12BIT  *
 7                      *  RESOLUTION - CLIPPED OUTPUTS    *
 8                      *  6 BIT RATES ARE AVAILABLE  BY   *
 9                      *  EXTERNAL SWITCHES REQUEST,      *
10                      *  70, 56, 50, 40, 30, AND 20      *
11                      *  KBITS/SECOND                    *
12                      ************************************


14                      ************************************
15                      *  THE PROCESSOR IS THE  8X300     *
16                      ************************************


18                           PROG  LISONG


20                      ************************************
21                      *DATA AND ADDRESS DECLARATIONS *
22                      ************************************

24           000001     INC   EQU   1
25           000377     DEC   EQU   377H
26         003 7 0      OUT3  LIV   3,7,8
27         001 7 0      OUT1  LIV   1,7,8
28         002 6 1      IN2   LIV   2,6
29         002 5 3      PORT2  LIV  2,5,3
30         002 2 1      SWIT2  LIV  2,2
31         004 7 0      OUT4  LIV   4,7,8


33                      ************************************
34                      *  MAIN PROGRAM                    *
35                      ************************************

37                           ORG   0
38  00000  6 01000           XMIT  0,R1
39  00001  6 01000      ADAPT  XMIT  0,R1
40  00002  6 02010           XMIT  10H,R2
41                      *COUNTER HAS BEEN INITIALIZED TO ZERO OFFSET
42                      ********************************************
43  00003  6 07004           SEL   OUT4
```

```
44  00004  6 27001          XMIT  1,OUT4
45                        *REGISTERS OF THE D TO A ARE DISABLED
46                        ************************************
47  00005  6 03001          XMIT  1,R3
48                        *STEP SIZE:MAX 8 BITS
49                        **********************
50  00006  6 04000          XMIT  0,R4
51                        *PREVIOUS ERROR,E(K-1)
52                        ************************
53  00007  6 05000          XMIT  0,R5
54                        *CURRENT ERROR,E(K)
55                        ********************
56  00010  6 07002    START  SEL SWIT2
57  00011  0 22106          MOVE  SWIT2,R6
58  00012  6 00000          XMIT  0,AUX
59  00013  3 06000          XOR   R6,AUX
60  00014  5 00016          NZT   AUX,SONG
61  00015  7 00213          JMP   LINEAR
62  00016  6 07002    SONG  SEL  IN2
63  00017  0 26105           MOVE IN2,R5
64                        *ERROR HAS BEEN READ
65                        ********************
66  00020  0 05000           MOVE R5,AUX
67  00021  3 04000           XOR  R4,AUX
68  00022  0 05004           MOVE R5,R4
69                        *UP DATE PREVIOUS ERROR
70                        ************************
71  00023  5 00143           NZT  AUX,DIFF
72                        *IS CURRENT ERROR = PREVIOUS ERROR ?
73                        ************************************
74  00024  6 00001           XMIT INC,AUX
75  00025  1 03003           ADD  R3,R3
76                        *YES, THEN INC. STEP SIZE BY 1 LSB
77                        **********************************
78  00026  5 03031           NZT  R3,NOYET
79                        *IS STEP SIZE BEYOND 377H ?
80                        ***************************
81  00027  6 03377           XMIT 377H,R3
82                        *YES,THEN CLIP IT
83                        *****************
84  00030  7 00033           JMP DECRM
85  00031  6 11000    NOYET  XMIT  0,R11
86  00032  6 11000          XMIT  0,R11
87                        *LAST TWO ARE DUMMY INSTRUCTIONS
88                        *******************************
89  00033  5 05121    DECRM  NZT  R5,NEQUAL
90                        *IS CURTRENT ERROR = 0  ?
91                        ************************
92  00034  6 00017           XMIT 17H,AUX
93  00035  2 02002           AND  R2,R2
94                        *YES,THEN PUT ZEROS IN UNUSED BITS IN R2
95                        ****************************************
96  00036  6 00377           XMIT DEC,AUX
97  00037  3 03006           XOR  R3,R6
```

```
 98  00040  6 00001          XMIT INC,AUX
 99  00041  1 06000          ADD  R6,AUX
100  00042  1 01001          ADD  R1,R1
101  00043  0 10011          MOVE OVF,R11
102  00044  6 00377          XMIT DEC,AUX
103  00045  1 10000          ADD  OVF,AUX
104  00046  1 02002          ADD  R2,R2
105                     *AND DEC COUNTER BY THE STEP SIZE
106                     **********************************
107  00047  5 10054          NZT  OVF,OKDM
108  00050  5 11054          NZT  R11,OKDM
109                     *HAS THE COUNTER GONE BELOW ZERO  ?
110                     **********************************
111  00051  6 01000          XMIT 0,R1
112  00052  6 02000          XMIT 0,R2
113                     *YES, THEN CLIP IT TO ZERO
114                     **************************
115  00053  7 00060        JMP  OK
116  00054  6 11000    OKDM  XMIT  0,R11
117  00055  6 11000        XMIT  0,R11
118  00056  6 11000        XMIT  0,R11
119  00057  6 11000        XMIT  0,R11
120                     *LAST FOUR ARE DUMMY INSTRUCTIONS
121                     **********************************
122  00060  6 07003    OK     SEL  OUT3
123  00061  0 01027          MOVE R1,OUT3
124  00062  6 07001          SEL  OUT1
125  00063  0 02027          MOVE R2,OUT1
126  00064  6 07004          SEL  OUT4
127  00065  6 27000          XMIT 0,OUT4
128  00066  6 27001          XMIT 1,OUT4
129                     *OUTPUT COUNTER CONTENTS AND STROBE THEM TO D/A CONV.
130                     ***************************************************
131                     *NOW START CHECKING FOR EXTERNAL REQUEST OF BIT RATE
132                     ***************************************************
133  00067  6 07002        SEL  PORT2
134  00070  0 25306    REPT  MOVE  PORT2,R6
135  00071  6 00000        XMIT  0,AUX
136  00072  3 06000        XOR   R6,AUX
137  00073  5 00075        NZT   AUX,NEXT56
138  00074  7 00152        JMP   KBIT70
139  00075  6 00001    NEXT56  XMIT  1,AUX
140  00076  3 06000        XOR   R6,AUX
141  00077  5 00101        NZT   AUX,NEXT50
142  00100  7 00157        JMP   KBIT56
143  00101  6 00002    NEXT50  XMIT  2,AUX
144  00102  3 06000        XOR   R6,AUX
145  00103  5 00105        NZT   AUX,NEXT40
146  00104  7 00165        JMP   KBIT50
147  00105  6 00003    NEXT40  XMIT  3,AUX
148  00106  3 06000        XOR   R6,AUX
149  00107  5 00111        NZT   AUX,NEXT30
150  00110  7 00172        JMP  KBIT40
151  00111  6 00004    NEXT30  XMIT  4,AUX
```

```
152  00112  3 06000          XOR   R6,AUX
153  00113  5 00115          NZT  AUX,NEXT20
154  00114  7 00200          JMP  KBIT30
155  00115  6 00005    NEXT20  XMIT  5,AUX
156  00116  3 06000          XOR   R6,AUX
157  00117  5 00070          NZT   AUX,REPT
158  00120  7 00205          JMP   KBIT20
159  00121  6 11000    NEQUAL XMIT  0,R11
160  00122  6 11000           XMIT  0,R11
161  00123  6 11000           XMIT  0,R11
162                     *LAST TWO ARE DUMMY INSTRUCTIONS
163                     **********************************
164  00124  6 00360           XMIT  360H,AUX
165  00125  3 02006           XOR  R2,R6
166  00126  2 02000           AND  R2,AUX
167  00127  3 06002           XOR  R6,R2
168                     *SEND ALL ONES TO NONUSED BITS IN R2
169                     ************************************
170  00130  0 03000           MOVE R3,AUX
171  00131  1 01001           ADD  R1,R1
172  00132  0 10000           MOVE OVF,AUX
173  00133  1 02002           ADD  R2,R2
174                     *AS ERROR IS NEQUAL TO ZERO INC COUNTER BY STEP SIZ
175                     ************************************************
176  00134  5 10140           NZT  OVF,SORRY
177                     *HAS COUNTER GONE GREATER THEN 7777H  ?
178                     ***************************************
179  00135  6 11000           XMIT  0,R11
180  00136  6 11000           XMIT  0,R11
181                     *LAST TWO ARE DUMMY INSTRUCTIONS
182                     **********************************
183  00137  7 00060           JMP  OK
184                     *NO, THEN OUTPUT ITS CONTENTS
185                     ******************************
186  00140  6 01377     SORRY  XMIT 377H,R1
187  00141  6 02377           XMIT 377H,R2
188                     *YES, THEN KEEP IT AT 7777H
189                     ***************************
190  00142  7 00060           JMP  OK
191  00143  6 00377     DIFF   XMIT DEC,AUX
192  00144  1 03003           ADD  R3,R3
193                     *E(K) DIFF. FROM E(K-1) THEN DEC. STEP SIZE BY 1LSB
194                     ************************************************
195  00145  5 03150           NZT  R3,OKEY
196                     *IS STEP SIZE = 0  ?
197                     ************************
198  00146  6 03001           XMIT 1,R3
199                     *YES, THEN RESTORE IT TO 1 LSB
200                     *******************************
201  00147  7 00033           JMP  DECRM
202                     *CARRY ON
203                     *********
204  00150  6 11000     OKEY   XMIT  0,R11
205  00151  7 00033           JMP  DECRM
```

```
206  00152  6 06371    KBIT70  XMIT  371H,R6
207  00153  6 00001            XMIT  1,AUX
208  00154  1 06006    LOOP1   ADD   R6,R6
209  00155  5 06154            NZT   R6,LOOP1
210                     *DELAY TO ALLOW A BIT RATE OF 70 KBITS-S
211                     ****************************************
212  00156  7 00010            JMP   START
213                     *GET A NEW ERROR
214                     ****************
215  00157  6 06374    KBIT56  XMIT  374H,R6
216  00160  6 00001            XMIT  1,AUX
217  00161  1 06006    LOOP2   ADD   R6,R6
218  00162  5 06161            NZT   R6,LOOP2
219                     *DELAY TO ALLOW A BIT RATE OF 56 KBITS-S
220                     ****************************************
221  00163  6 11000            XMIT  0,R11
222  00164  7 00010            JMP   START
223  00165  6 06364    KBIT50  XMIT  364H,R6
224  00166  6 00001            XMIT  1,AUX
225  00167  1 06006    LOOP3   ADD   R6,R6
226  00170  5 06167            NZT   R6,LOOP3
227                     *DELAY TO ALLOW A BIT RATE OF 50 KBITS-S
228                     ****************************************
229  00171  7 00010            JMP   START
230  00172  6 06354    KBIT40  XMIT  354H,R6
231  00173  6 00001            XMIT  1,AUX
232  00174  1 06006    LOOP4   ADD   R6,R6
233  00175  5 06174            NZT   R6,LOOP4
234                     *DELAY TO ALLOW ABIT RATE OF 40 KBITS-S
235                     ****************************************
236  00176  6 11000            XMIT  0,R11
237  00177  7 00010            JMP   START
238  00200  6 06334    KBIT30  XMIT  334H,R6
239  00201  6 00001            XMIT  1,AUX
240  00202  1 06006    LOOP5   ADD   R6,R6
241  00203  5 06202            NZT   R6,LOOP5
242                     *DELAY TO ALLOW A BIT RATE OF 30 KBITS-S
243                     ****************************************
244  00204  7 00010            JMP   START
245  00205  6 06275    KBIT20  XMIT  275H,R6
246  00206  6 00001            XMIT  1,AUX
247  00207  1 06006    LOOP6   ADD   R6,R6
248  00210  5 06207            NZT   R6,LOOP6
249                     *DELAY TO ALLOW A BIT RATE OF 20 KBITS-S
250                     ****************************************
251  00211  6 11000            XMIT  0,R11
252  00212  7 00010            JMP   START
253  00213  6 01000    LINEAR  XMIT  0,R1
254                             END   LISONG

TAL ASSEMBLY ERRORS =     0
 --
```

```
000C0 140400
000C1 140400
000C2 141010
000C3 143404
000C4 153401
000C5 141401
000C6 142000
000C7 142400
00010 143402
00011 011046
00012 140000
00013 063000
00014 120016
00015 160213
00016 143402
00017 013045
00020 002400
00021 062000
00022 002404
00023 120143
00024 140001
00025 021403
00026 121431
00027 141777
00030 160033
00031 144400
00032 144400
00033 122521
00034 140017
00035 041002
00036 140377
00037 061406
00040 140001
00041 023000
00042 020401
00043 004011
00044 140377
00045 024000
00046 021002
00047 124054
00050 124454
00051 140400
00052 141000
00053 160060
00054 144400
00055 144400
00056 144400
00057 144400
00060 143403
00061 000427
00062 143401
00063 001027
00064 143404
00065 153400
00066 153401
00067 143402
00070 012546
00071 140000
00072 063000
00073 120075
00074 160152
00075 140001
00076 063000
00077 120101
00100 160157
00101 140002
00102 063000
00103 120105
00104 160165
00105 140003
00106 063000
00107 120111
00110 160172
00111 140004
00112 063000
00113 120115
00114 160200
00115 140005
00116 063000
00117 120070
```

```
00120 160205
00121 144400
00122 144400
00123 144400
00124 140360
00125 061006
00126 041000
00127 063002
00130 001400
00131 020401
00132 004000
00133 021002
00134 124140
00135 144400
00136 144400
00137 160060
00140 140777
00141 141377
00142 160060
00143 140377
00144 021403
00145 121550
00146 141401
00147 160033
00150 144400
00151 160033
00152 143371
00153 140001
00154 023006
00155 123154
00156 160010
00157 143374
00160 140001
00161 023006
00162 123161
00163 144400
00164 160010
00165 143364
00166 140001
00167 023006
00170 123167
00171 160010
00172 143354
00173 140001
00174 023006
00175 123174
00176 144400
00177 160010
00200 143334
00201 140001
00202 023006
00203 123202
00204 160010
00205 143275
00206 140001
00207 023006
00210 123207
00211 144400
00212 160010
00213 140400
```

APPENDIX C

(Preference Measurement Methods)

C.-Methods of Subjective Measurements of Speech Quality

C.1.- Introduction.-For convenience of the reader, brief descriptions of four preference measurement methods will be given. More details can be found in references (2), (48), (58), (59) and (60). The following has been taken directly from (2) and (48).

C.2.-Relative Methods

C.2.1.-Isopreference Method

The term 'isopreference' in this context states that two speech siganls are isopreferent when the votes averaged over all listeners show an equal preference for the speech test and speech reference signals.

The speech test signal is compared in a forced-choice test procedure directly with a high-fidelity speech reference signal that is subjected to variable degrees of degradation.. Thus the isopreference level of the speech test signal is defined as the signal-to-noise ratio of the speech reference signal at which preference votes of a listener group are equally divided. The test and reference signals are then 'isopreferent'. The preference of the test signal is described by the signal-to-noise ratio in decibels of the degraded reference signal.

Two types of reference signals are proposed, both of them utilizing shaped random noise for the generation of their respective degradation signals. These reference signals are fundamentally different, not only with regard to the generation principle, but also with regard to their auditory impression.

A reference signal r(t) is generated by adding a certain amount k of a degradation signal d(t) to a high-fidelity speech signal s(t).

$$r(t) = s(t) + k.d(t) \qquad (C.1)$$

The degradation that are used are, ADDITIVE random noise, for which the respective reference signal will be referred to as 'additive reference'

$$d(t) = n_o(t) \qquad (C.2)$$

or MULTIPLICATIVE random noise, for which the respective reference signal will be referred to as 'multiplicative reference'

$$d(t) = s(t).n_o(t) \qquad (C.3)$$

$n_o(t)$ in Eq. (C.2) and (C.3) should be A-Weighted Pink Noise. (USA Standard S1.4-1961)

The reference signal used should be adjustable over a S/N range of about 60 dB to cover the total quality range. The test results are given by isopreference levels.

C.2.2.-Relative Preference Method

In this method, the quality of the test signal is measured relative to the quality defined by several reference signals that represent different types of speech distortion. Typically, five reference systems are used.

The selected reference signals are placed on an arbitrary rating scale by considering how often each reference signal is preferred to the other reference signals. This can be done by isopreference tests as well as by direct comparison of all the possible pairs among the set of selected reference signals. The test signal is located on the same rating scale by considering how often it is preferred to any re

ference signal. For instance, test signal A is presented together with each signal Bi out of the set of reference signals in repeated signal pairs ABiABi. Comparisons are performed in a random sequence with all the apirs of speech signals possible in both the forward and the reverse order of presentation.

The test results are expressed as preference rating numbers (PRN) between zero and ten on a dimensionless preference rating scale.

C.3.-Absolute Methods

C.3.1.-Category Judgment Method

The quality impression of a speech test signal is described by the listeners in terms of five response categories, Excellent(5), Good(4), Fair(3), Poor(2), and Unsatisfactory(1).

The experiment consists of two phases, familiarization and evaluation. The test start with the familiarization period during which the test signals are introduced to the listeners, and one or two reference signals are presented, of which the 'correct' category evaluation is identified to the listeners. The reference signals may be those defined by the experimenter as representative of the extreme categories, and by means of these the listeners can establish their points of reference for their responses.

In the evaluation phase the test signals are presented in a random order and the listeners are asked to mark the one category that correspond to their quality impression of the speech signals presented. From the listeners' responses a Mean Opinion Score (MOS) or Cumulative Preference can

now be calculated.

If there are many speech test signals to be evaluated, it is advisable to intersperse identified speech reference signals to refresh the standard of reference of the listeners.

C.3.2.-Absolute Preference Judgment Method

The main advantage of the category judgment method, its capability to describe preference in terms of a very small number of categories, can be construed to also be its main disadvantage. At first, the listeners sometaimes find it annoying to describe their complex impressions of speech quality in terms of only five categories. Often the listeners would prefer to get a chance to describe their impressions with higher accuracy. Second, a quantization of judgments to a small number of categories does not offer any advantage if the final goal of the absolute rating is a mean score, which is normally given as an unquantized decimal number.

The absolute preference judgment method avoids the above disadvantages of the category judgment method. Here the listeners are requested to evaluate the test signals in terms of numbers from zero to ten, where zero is the worst signal and ten correspond to the best signal. The listeners may use decimal fractions between integers if they think they can improve the accuracy of their judgments. Obviously, no quantization of these listeners responses is useful or necesary if the goal of the test is an evaluation in terms of a mean score.

REFERENCES

1.-C.L. Song, J. Garodnick and D.L. Schilling, 'A variable Step-size Robust Delta Modulator'. IEEE Trans. on Communication Technology, Vol. Com 19, Nº 6, Dec. 1971, pp 1033-1044.

2.-'IEEE Recommended Practice for Speech Quality Measurements' IEEE Trans. on Audio and Electroacoustics, Vol. AU 17, Nº 3, Sep. 1969, pp 225-246.

3.-J.L. Flanagan, M.R. Schroeder, B.S. Atal and other. 'Speech Coding, IEEE Trans. on Communications, Vol. Com-27, Nº 4 April 1979, pp 710-736.

4.-J.W. Bayless, S.J. Campanella and A.J. Goldberg, 'Voice Signals: bit by bit' IEEE Spectrum, October 1973, Vol 10 pp 28-34.

5.-J.L. Flanagan, Speech Analysis, Synthesis and Perception, Second Edition, Springer-Verlag, New York, 1972.

6.-J.A. Betts, Signal Processing Modulation and Noise, Hodder and Stoughton, London, 1976.

7.-A.J. Goldberg, H.L. Shaffer, ' A Real-Time Adaptive Predictive Coder Using Small Computers. IEEE Trans on Communications, Vol. Com 23, Nº 12, Dec. 1975, pp 1443-1451.

8.-R.E. Crochiere, S.A. Webber, and J.L. Flanagan, 'Digital coding of speech in sub-bands' Bell Syst. Tech. J., vol.55 October 1976, pp 1069-1035.

9.-R.E. Crochiere, 'On the design of sub-band coders for low-bit rate speech communication', Bell Syst. Tech. J.,vol.56 May 1977. pp 747-770.

10.-R. Zelinski and P. Noll, 'Adaptive transform coding of speech signals', IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-25, Aug. 1977, pp 299-309.

11.-R. Zelinski and P. Noll, 'Approaches to adaptive transform speech coding at low-bit rates', IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-27, Feb. 1979, pp 89-95.

12.-J.M. Tribolet and R.E. Crochiere, 'Frequency domain coding of speech', IEEE Trans. Acoust., Speech, Signal Processing vol. ASSP-27, Oct. 1979, pp 512-530.

13.-R.V. Cox and R.E. Crochiere, 'Real-time simulation of adaptive Transform Coding', IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-29, April 1981, pp. 147-154.

14.-M.R. Schroeder, 'Vocoders: Analysis and Synthesis of speech Proc. IEEE, vol. 54, Nº 5, May 1960, pp 720-733.

15.-M.C. Davie, 'Channel vocoder based on c.c.d. discrete-Fourier transform processors', IEE Proc., Vol. 127, Pt.F, Nº 2, April 1980, pp 132-143.

16.-B. Gold, P.E. Blankenship and R.J. McAulay, 'New Applications of Channel vocoders', IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-29, Nº 1, Feb. 1981, pp 13-23.

17.-C.K. Un, 'A low-rate digital Formant Vocoder', IEEE Trans. on Communications, Vol. Com-26, Nº 3, March 1978, pp 344-355.

18.-M.R. Schroeder, 'Correlation techniques for speech bandwidth compression', J. Audio Eng. Soc., vol. 10, 1962, pp 163-166.

19.-B.S. Atal and S.L. Hanauer, 'Speech analysis and synthesis by Linear Prediction of the speech wave', J. Acoustic Soc. Amer., vol. 50, Aug. 1971, pp 637-655.

20.-J.D. Markel and A.H. Gray, Jr., 'A Linear Prediction vocoder simulation based upon the autocorrelation method', IEEE Trans Acoust., Speech, Signal Processing, Vol. ASSP-22, Nº 2, April 1974, pp 124-134.

21.-E.M. Hofstetter, J. Tierney and D Wheeler, 'Microprocessor realization of a Linear Predictive vocoder', IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-25, Nº 5, Oct. 1977, pp 379-387.

22.-J.D. Markel, 'Digital Inverse filtering, a new tool for formant trajectory estimation', IEEE Trans Audio Electroacoustic, Vol. AU-20, June 1972, pp 129-137.

23.-B.S. Atal and M.R. Schroeder, 'Adaptive Predictive Coding of speech signals', Bell Syst. Tech. J., vol. 49, Oct. 1970, pp 1973-1986.

24.-J.D. Gibson, S.K. Jones and J.L. Melsa, 'Sequentially adaptive prediction and coding of speech signals', IEEE Trans. on Communications, Vol. Com.-22, Nov. 1974, pp 1789-1797.

25.-C.K. Un and D.T. Magill, 'The residual-excited linear prediction vocoder with transmission rate below 9.6 kbits/s.' IEEE Trans. on Communications, Vol. COM-23, Nº 12, Dec. 1975, pp 1466-1473.

26.-M. Nakatsui, D.C. Stevenson and P. Mermelstein, 'Subjective evaluation of a 4.8 kbits/s. Residual-excited linear prediction coder. IEEE Trans. on Commun., Vol. Com-29, Nº 9, Sep. 1981, pp 1389-1393.

27.-M.R. Schroeder and E.E. David, Jr., 'A vocoder for transmitting 10 Kc/s. speech over 3.5 Kc*s channel', Acustica, Vol. 10, Nº 1, 1960, pp 35-43.

28.-J.L. Flanagan and R.M. Golden, 'Phase Vocoder', Bell Syst. Tech. J., Vol. 45, Nov. 1966, pp 1493-1509.

29.-M.R. Portnoff, 'Implementation of the Digital Phase Vocoder Using the Fast Fourier Transform', IEEE Trans. Acoust. Speech, Signal Processing, Vol. ASSP-24, Nº 3, June 1976, pp 243-248.

30.-D. Malah, R.E. Crochiere and R.V. Cox, 'Perfomance of Transform and subband coding systems combined with harmonic scaling of speech', IEEE Trans. Acoust., Speech, Signal Processing, Vol.ASSP-29, Nº 2, April 1981, pp 273-283.

31.-M.F. Smith, 'PDP-11/60 - MC6800 Microprocessor Development System', Microprocessors and Microsystems, Vol. 4, Nº 5, Jun. 1980, pp 175-180.

32.-J.L. Payne, 'A software development workshop for programmable microelectronics', The Radio and Electronic Engineer, Vol. 48, Nº 3, March 1978, pp 122-127.

33.-D.R. Mc Glynn, 'Microprocessors, Technology, Architecture & Applications, John Wiley & Sons, Toronto, 1976.

34.-Digital Equipment Corporation. 'LSI - 11, PDP-11/03 user's manual', Maynard, Massachusetts, 1976.

35.-Fairchild, 'TTL Isoplanar Memory 93L412 - 93L422: 256x4-bit fully decoded random access memory', Preliminary data sheet, May 1976.

36.-M.H. Ackroyd. Reader in Telecommunications. The University of Aston in Birmingham. The Department of Electrical Engineering.

37.-T. Apelewicz and D.L. Schilling, 'Aprogrammable Microcomputer for real time speech processing', ICC Conference, Chicago, III, 1978, pp 13.7-314 - 13.7-316.

38.-Philips (Signetics), '8X300 Microprocessor', System Design Manual.

39.-Adam Osborne & Associates, Incorporated, 'The 8X300 (SMS 300)', Chapter 14, pp 14.1 - 14.27.

40.-Philips (Signetics), 'Microcomputer Cross Assembly Program (MCCAP)

41.-Signetics Microprocessors Training Department, '8X300 Programming Course', 1977.

42.-C.K. Un and H.S. Lee, 'A study of the Comparative Perfomance of Adaptive Delta Modulation Systems', IEEE Trans. on Communications, Vol. COM.-28, Nº 1, Jan. 1980, pp 96-101.

43.-V.R. Dhadesugoor, C. Ziegler and D.L. Schilling, 'Delta Modulators in Packet Voice Networks'. IEEE Trans. on Communications, Vol. COM-28, Nº 1, Jan. 1980, pp 33-49.

44.-R. Steele, 'Chip Delta Modulators Revive Designer's Interest', Electronics, October 1977, pp 86-93.

45.-R. Steele, 'Delta Modulation Systems', Pentech Press, London, 1975.

46.-D.L. Schilling, J. Garodnick and H.A. Vang, 'Voice Encoding for the Space Shuttle Using Adaptive Delta Modulation', IEEE Trans. on Communications, Vol. COM-26, Nº 11, Nov. 1978, pp 1652-1659.

47.-C.L. Song, J. Garodnick and D.L. Schilling, 'A Variable-Step-Size Robust Delta Modulator', IEEE Trans. on Communications Technology, Vol. COM-19, Nº 6, Dec. 1971, pp 1033-1044.

48.-W.P. Pachl, G.E. Urbanek and E.H. Rothauser, 'Preference Evaluation of a Large Set of Vocoded Speech signals',IEEE Trans. Vol.AU-19, Nº 3, Sep. 1971, pp 216-224.

Other References

49.-D.J. Tonge, D.L. Gaunt and J.P. Kendal, 'Programmable Logic and Microprocessors', POEEJ, Vol. 70, Oct. 1977..

50.-R.H. Eckhouse, Jr., 'Minicomputers System, Organization and Programming (PDP-11)', Prentice-Hall, Inc., New Jersey 1975.

51.-M.D. Rowe, 'The Selection of Microprocessors', The POEEJ, Vol. 71, part 2, July 1978.

52.-J.L. Flanagan, K. Ishizaka and K.L. Shipley. 'Signal models for low bit-rate coding of speech', the J. of the Acoustical Soc. of Am., Vol. 68, Nº 3, Sep. 1980, pp 780-791.

53.-R.E. Crochiere and Y. Kato, 'Speech waveform coding:Techniques and perfomance', The J. of the Acoustical Soc. of Am., Vol. 66, Nº 6, Dec. 1979, p 1627.

54.-J. Makhoul and M. Berouti, 'Predictive and residual encoding of speech', The J. of the Acoustical Soc. of Am., Vol. 66, Nº 6, Dec. 1979, pp 1633-1641.

55.-R.E. Crochiere and J.M. Tribolet, 'Frequency domain techniques for speech coding', .---. pp 1642-1652.

56.-A.J. Goldberg, 'Practical implementations of speech waveform coders for the present day and for the mid 1980s', .---. pp 1653-1657.

57.-T.P. Barnwell,III, 'Objective measures for speech quality testing', .---., pp 1658-1663.

58.-E.H. Rothauser, G.E. Urbanek, and W.P. Pachl. 'Isopreference method for speech evaluation', J. Acoust. Soc. Amer., vol. 44, 1968. pp. 408-418.

59.-____,'Acomparison of preference measurement methods', J. Acoust. Soc. Amer., April 1971.

60.-M.H.L. Hecker and C.E. Williams, 'Choice of reference conditions for speech preference tests', J. Acoust. Soc. Amer., vol. 39, 1966, p 946.