DESIGN CONSIDERATIONS OF A SPEECH RECOGNITION SYSTEM USING SPECIAL PURPOSE INTEGRATED CIRCUITS

by

Menahem Lowy

Memorandum No. UCB/ERL M83/50

25 August 1983

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Design Considerations of a Speech Recognition System Using Special Purpose Integrated Circuits

By

Menahem Lowy

B.S. (Technion-Israel Institute of Technology) 1960
M.S. (Technion-Israel Institute of Technology) 1975

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

Engineering

in the

GRADUATE DIVISION

OF THE

UNIVERSITY OF CALIFORNIA, BERKELEY

Approved: Robert W. Brodersen 8/22/83
Chairman Date

David A. Hodges

Michael J. Klass

....

Design Considerations of a Speech Recognition System Using Special Purpose Integrated Circuits

Ph.D. *Menahem Lowy* EECS Dept

Signature: [illegible]

Committee Chairman

ABSTRACT

High accuracy, large vocabulary speech recognition systems have extremely high computational requirements. In order to perform the computation fast enough to have real time response, an investigation was made of special purpose integrated circuits. These integrated circuits make it possible to obtain a system small and inexpensive enough to be generally useful. An analysis of the architecture for these circuits is presented and a circuit which implements a critical portion of the algorithm was realized. This integrated circuit implements a dynamic time warp algorithm which is required in the pattern matching part of a high performance speech recognition system. With this chip the system is capable of recognizing 500 words in real time with high accuracy.

to

Israel and Nicha

Marthe

Osnat

Carmit

Shira

ACKNOWLEDGEMENTS

I wish to express my sincere gratitude to Prof. Bob Brodersen for his support and encouragement throughout the course of this work. His vision and guidance concurred to keep this research on the right track. I am further indebted to him for his helpful comments concerning the writing of this manuscript. I would also like to thank Hy Murveit for undertaking this project with me. This thesis describes only a part of the speech recognition system. All the programs, endless simulations and determination of the numerous parameters of the system are his work. Without him the system would not have been realized.

My fellow students in the EECS Dept. have been extremely helpful and have contributed to this work in different ways. It is not possible to list all their names here. However, special thanks should go to David Mintz who build and debugged the hardware for the speech recognition system. I am also grateful to Alex Para and his group in the electronic workshop for building some of the hardware boards of the system.

Last and most important I would like to thank my late father, my mother, my wife Marthe and my daughters Osnat, Carmit and Shira for my education, the time they allowed me to devote to this work instead of sharing it with them and for their patience and encouragements during these long years of efforts. To them more than anybody else I am indebted for being able to complete this thesis. In acknowledgement of their dedication and sacrifices during these years I would like to dedicate this thesis to them.

TABLE OF CONTENTS

CHAPTER 1

Introduction

Improving man-machine communication has been a subject for research for many years. The ease of interfacing with machines by voice hold promises of increasing the use of computers or intelligent machines for the more menial daily tasks. This goal has sparked a wide interest in the field of voice or speech recognition [1]. The advances in large scale integration should make it possible to inexpensively implement the circuits and algorithms needed for voice communication with computers. This thesis describes the realization of such a speech recognition system.

A more general motivation behind this project is the investigation of LSI integration techniques as applied to real time signal processing. The speech application was chosen for investigation because:

(1) Speech recognition systems have reached a stage where they promise to be commercially feasible. It is possible to obtain very high recognition accuracy (above 99 percent) by the use of the dynamic time-warp algorithms [2],[3]. These algorithms solve the problem of aligning the time-warped speech utterances and enable accurate comparison between the different patterns. However they have the disadvantage of requiring a large amount of computation.

(2) The bandwidth of the speech signal is compatible with the processing rates of present-day digital MOS technology [4].

To appreciate the importance of these two points, the nature of speech and the problem associated with it must be explained.

1.1. Nature of Speech

Speech communication between human beings involves the generation and reception of a complex acoustic signal. This process may be thought of as a coding operation which takes place over a hierarchy of processing levels. The highest level is the one where thoughts are formed. These are encoded on a lower level, the *linguistic level*, in the form of words. The words are encoded in successively lower levels involving neural processing and articulatory movements until the lowest level is reached with the acoustic signal.

Redundant information is present in every processing level to correct ambiguities at that level. One form of this redundant information is the multiple perceptual "cues" used to distinguish between the different speech elements. As an example to distinguish between the sound /s/ and other speech sounds we use the presence of the fricative noise, the initial rate of rise in energy, the concentration of energy in certain frequency bands, the relative intensity of the sound, etc. . Another form of this redundant information involves interaction between levels. Thus an error in the identification of a speech sound from its acoustic pattern can be corrected by referring to physical or linguistic rules governing the production of such sounds, by knowledge about the vocal characteristic of the talker and also whether the total message "makes sense".

The cues, or sources of knowledge, are subconsciously used by talkers when trying to understand a sentence. These include the characteristics of speech sounds (*phonetics*), variability in pronunciation (*phonology*), the stress and intonation patterns of speech (*prosodics*), the sound patterns of words (*lexicon*

), the grammatical structure of language (*syntax*), the meaning of words and sentences (*semantics*), and the context of conversation (*pragmatics*).

In addition to the acoustic waveform, some of the additional cues present in speech can be used mostly for task oriented applications. The grammatical structure of sentences can be used to reduce search by restricting the number of acceptable alternatives. Syntactic structure imposes an ordering and mutually interdependent relationship among words such that only a subset of the vocabulary is searched. The accuracy for the sentence as a whole will increase as the system is restricted to smaller vocabularies. Additional level of information such as pragmatics are added in an AI (artificial intelligence) approach to improve recognition accuracy. [5]

At present our recognition system does not use these cues to improve recognition. The hardware uses only the lowest level, i.e. the acoustic waveform of the utterance, to recognize words . If these higher knowledge sources were added, this would further improve recognition performance.

1.1.1. Basic Speech Mechanism

Voiced speech is produced by the vibration of the vocal folds powered by air coming from the lungs during exhalation. The resulting pulse wave drives the acoustic system which is composed of the nasal and the vocal tract. The function of the vocal tract is to filter the spectrum of the excitation, depending on the position of the speech organs, to produce the sounds known as speech. The linguistic information in speech is conveyed by the modifications introduced in the vocal tract by the tongue, lips, jaws and velum. and by switching on and off the vibrations of the vocal cords. The linguistic information then appears as a time varying spectrum which depends on the particular configuration of the vocal tract at that time.

The changes in the shape of the vocal tract and the resulting changes in the spectra of the acoustic waveform correspond to different sounds. For certain sounds, the organs of speech do not take a fixed configuration but are in constant motion along a smooth trajectory. Therefore the result is a complex and continual motion transmitted to the emitted sound in the form of a constantly changing amplitude spectrum. To characterize speech we must then characterize this time varying spectrum. Due to the masses involved in the speech production mechanism, the rate of change in the shape of the vocal tract is limited. Therefore, for small intervals of time, of the order of 10 to 20 msec., the spectrum can be considered to be stationary.

This is the basis upon which speech can be transformed into a pattern and compared to other patterns to determine similarity between them.

1.1.2. Phonetic Considerations

The acoustic properties of the sounds in the English language influence the choice of the filters and other parameters in the realization of the feature extraction subsystem [6].

The design of the feature extraction block depends on the following characteristics of the English sounds:

(1) the frequency range of the spectrum of the acoustic waveform. This influences the number of channels of the spectrum analyzer and also the amount of computation in the system.

(2) the characteristic of the waveform of the different sounds. The shortest duration of the plosive consonants as */b/, /p/,/d/, /t/* influences the time during which the features are assumed to be constant.

(3) the range of intensity of the different sounds. This is necessary for defining the dynamic range of the filters and of the A/D converter.

(4) the variation in the pitch frequency of the vocal cords. The energy within a frequency band depends on the number of pitch harmonics contained in the band. In the low frequency part of the speech-sound spectrum, where the spacing between pitch harmonics is relatively high, a too narrow band-defining filter exhibits energy variations due to the variation in the number of pitch harmonics in its band. These variations in energy are mistakenly attributed to variations of the pattern.

The acoustic spectra cover the frequency range from below 200Hz up to 8Khz. However, information which permits discrimination between voiced and unvoiced sounds is also contained in the appearance of the fundamental of the pitch waveform. The sounds of highest frequency are the fricative consonants / ϑ /,/s/ and /*f*/. For the sound /s/, this frequency band extends from 4 Khz up to 8 Khz. The main noise energy for /*f*/ and / ϑ / is in the band from 6 Khz to 8 Khz.

The plosive consonants or stops, are characterized by short silence (or near-silence for voiced stops) followed by a short burst of noise when the stop is released. This burst of noise is of very short duration, about 10 to 15 ms, when the sound has no or little aspiration.

The relative intensity of the sounds in the English language has a range of about 30 dB. The vowels have the higher intensity levels whereas the weak fricatives and plosives have low intensity such as the sound of / ϑ / in *thin*. Therefore, in a single syllable like *thought*, the recognition system is faced with a change of intensity of almost 30 dB.

1.2. Problems in Speech Recognition

In general, template based recognition algorithms can be characterized as a processing and comparison of a received acoustic pattern with an expected pronunciation or template. The sources of errors in this matching are: acoustic variability, hardware and algorithm limitations in the comparison, and inadequacies in the stored templates [7]. Variability is due to the deviation of message pronunciation from time to time and talker to talker from the expected sound pattern. Due to this variability, and due to the fact that the only level of information is the acoustic waveform, the hardware must implement the time alignment algorithm between the patterns. Finally, storing adequate templates is a crucial factor in the accuracy of the system.

These problems are more difficult in a system which allows the input words to be spoken without pauses (connected word recognition system) rather than one which has pauses between the words (isolated word recognition system). This is because in connected speech it is difficult to determine where one word ends and another begins. In addition, due to the modification of the beginning and end of a word (co-articulation effect), acoustic characteristics of sound in a connected word recognition systems exhibits much greater variability. A more complex problem is addressed in a *speech understanding system* which makes use of sources of knowledge and of task-specific information to reduce search in the vocabulary. In such a system as long as the message is understood it is not important to recognize each phoneme or word correctly [8].

The accuracy of a system is very dependent on a variety of factors. By increasing the size of the vocabulary, making the system user independent or recognizing connected speech, we increase the number of recognition errors in the system. In addition, vocabularies with many similar sounding words are more difficult to recognize. For small vocabulary systems, trained for each user and recognizing isolated words, it is possible to obtain virtually error free recognition.

1.3. Brief Review of Speech Recognition Efforts [9],[10]

The research on automatic speech recognition is based on an understanding of the generation and physics of speech, the mechanisms of hearing, and the tolerance of the ear. A great deal of useful research has also been concerned with relations between the perceptual and acoustics aspects of speech communication[11].

The first truly successful recognizer was reported in 1952 by Davis, Biddulph and Balashek of Bell Laboratories [12]. This device could recognize the ten digits, spoken over the telephone by a single talker, with an accuracy approaching 100 percent. Since then, many systems have been reported . In 1956 Wiren and Stubs [13] produced a device based on phoneme classification. This classification was based on voiced/unvoiced, turbulent/ nonturbulent, stop/fricative, and acute/grave determinations. In 1959 Forgie and Forgie [14] at Lincoln Laboratories reported a study where they used a 35-channel filter bank whose outputs were envelope-detected, sampled and fed to a computer. The computer program made use mainly of the frequency positions of the formants F1 and F2 and of fundamental voice-frequency measurements. For 21 male and female talker, with no adjustment for the talker, the vowel recognition accuracy was 93 percent.

The availability of computers has led to it being used in place of the hardware concerned with classifying the utterance on the basis of output signals from a hardware spectrum analyzer. A number of practical recognizers were demonstrated in Japan. The high recognition scores obtained in these systems are sometimes attributed to the phonetic simplicity of the Japanese language. A study of vowel recognition was reported in 1961 by Suzuki and Nakata [15]. They used a 26 channel spectrum analyzer and separate wideband, formant-related channels. The channel outputs were separately grouped and connected to individual vowel-decision circuits. The speech was segmented into voiced and unvoiced segments and envelope-intensity and fundamental voice-frequency measurements were also used. To reduce the errors due to formant movement during vowel sounds, separate recognition decisions were made at intervals throughout voiced segments. Final classification was made by observing the phoneme most frequently recognized. Sakai and Doshita reported in 1962 [16] using separate circuits for segmenting speech into vowels and consonants and for classifying the segmented phonemes. Zero-crossing analysis was combined with measurements of the variations of energy in various frequency regions. The segmentation operation made use of measures of the "stability" of, and the "distance" between the digital patterns generated. It was claimed that 90 percent correct recognition was obtained on vowels and 70 percent on consonants.

Pols et al. have published results [17],[18] obtained by principal components analysis . Using this method the three first formants F1, F2 and F3 are found from the spectral analysis of the speech waveform . Their preprocessor used 16 channels . The dimensionality was reduced by finding the best three-dimensional plane defined by the 16 dimensional vectors. The word-recognition system based on this method obtained more than 90 percent correct recognition.

In 1971 Sakoe and Chiba [2] published results obtained when using dynamic programming for solving the problem of time alignment between the patterns to be compared. This was a major breakthrough as it improved the recognition accuracy from about 90% obtained until then up to 99% and above. The algorithms based on dynamic programming are now a major technique in speech recognition systems.

The speech-understanding project sponsored by the advanced research project agency (ARPA) in 1971 gave renewed impetus to research in the field. The goal was to achieve continuous speech understanding for a 1000-word vocabulary for a specific task for a small number of speakers in near real time on a big computer (a capability of 100 MIPS). The absence of specifications concerning the applicability of these systems to real-world problems and that the systems be cost effective helped to focus it on scientific and computational issues. But unfortunately this resulted in ignoring the work needed to develop future practical systems [19].

Five systems were originated at: Bolt Beranek and Newman Inc. (BBN), at Carnegie-Mellon University (CMU), at Lincoln Laboratories (LL), at Stanford Research Institute (SRI) and at System Development Corporation (SDC). After two years the project focused on the three systems at BBN, CMU and SDC.

The three ARPA speech-understanding systems use semantic, grammatical and phonological sources of knowledge to overcome acoustic ambiguity. The approach which was chosen consisted basically of the following steps: the first step was to process the acoustic input to obtain a phonetic transcription of what was said. This consists of a sequence of states called phonetic segments. The second step consisted of trying to find the most likely candidate words and word sequences that might be present in this phonetic transcription.

A description of the systems and of the results obtained in the different groups which participated in the project is described in [19].

Only the Harpy system from CMU met or exceeded in 1976 the goals set forth in 1971. The recognition accuracies obtained is between 80 and 90 percent for a constrained vocabulary of 200-300 words for a specific task. Harpy is an extension of a Markov model of a sentence decoding originally employed by Baker [20] in a sentence recognition system called Dragon. In that system, time alignment is obtained using Markov models. Other knowledge domains like syntactic, semantic and acoustic can also be modeled as Markov processes. The HEARSAY-II system also from CMU, exhibited the best performance of the systems other than Harpy. In this system a hierarchy of speech units is used : subphonemes, phonemes, surface phonemes, syllables, words and phrases represent knowledge sources to achieve significant economy in the computation. To deal with multiple knowledge sources in the system the following strategy was defined: whenever any knowledge source has something to say, it writes it on a structurally uniform data base for all other knowledge sources to see. The control of the computer processor is taken by the knowledge source that has the highest match between this pattern matching and one of its templates.

Since the ARPA project the main effort in speech recognition has been pursued at Bell Labs, at IBM, and in Japan. Many speech recognition systems have been developed and are available commercially. Doddington in [21], has made a comparative survey of some of the systems. Continuous efforts at IBM have resulted in a continuous speech recognition system as described in [22]. The IBM system consists of an *acoustic processor* (AP) followed by a *linguistic decoder* (LD). The speaker and the AP are conceptually viewed as combined in a acoustic channel which provides the LD with information y. The LD tries to find that word string w which maximizes the probability $P(w,y)$ of the joint observation of the input-output pair w,y at the terminals of the channel. The system can correctly identify 93 percent of the words contained in sentences of a 1000 continuous word special purpose vocabulary.

1.4. Speech Recognition System Requirements

This thesis describes a speech recognition system which has the potential to be of sufficiently low cost and small size to make it possible to significantly increase the use of such systems.

The important characteristics of this system are:

- very high recognition accuracy, typically > 99% for a cooperative talker using the system in isolated word recognition mode.
- recognizes words spoken both in isolation and in continuous speech
- speaker dependent system that is, each user trains the system by storing one or more templates representing a word in the vocabulary.
- real time recognition of a vocabulary of about 500 templates (more precisely, the vocabulary being searched in real time amounts to 250 sec. of speech).

1.4.1. Processing of Speech Signals in Real Time

It has been stated that the main problem in speech recognition systems is how to use the massive amount of data pertaining to different sources of knowledge and to process them in real time to maximize recognition accuracy [23].

A word recognition system is basically a pattern recognition system in which an incoming pattern is compared to the patterns stored in the system. The model of such a system is illustrated in Fig. 1.1.

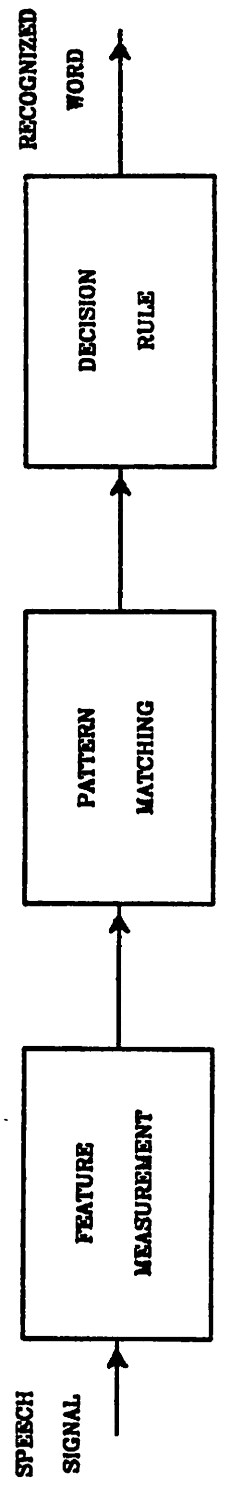


Fig. 1.1 Basic model of the speech recognition system

processor for all the computations in the system. Due to overhead operations like fetching and decoding of instructions, such a system is not optimized for the high rate of computation needed for the feature extraction done in the front end and for the dynamic time warp algorithms. Therefore, the dynamic time warp algorithms used are modified in order to reduce the amount of computation needed. Many of these modifications introduce recognition errors. It seemed necessary therefore to implement the highest performance algorithms without compromise and solve the computation speed problem by using another approach.

To increase the rate of computation without having recourse to expensive high performance processors, we use custom designed integrated circuits. Fig. 1.3 shows the block diagram of the system using this approach. The front-end and the realization of the dynamic time-warp algorithm are distinct functions and use special purpose circuits. Flexibility in the system is still retained by the use of a standard microprocessor (which can have relatively low performance), as a controller.

The custom designed circuits perform the necessary operations in the required time without any difficulty. Improvement in the computational rate is done using pipelining techniques and parallel circuits. These circuits use smaller area and lower clock rates then the generally used micro-processors. In addition, it is possible to increase the capability of the system by connecting other processing I.C.'s in parallel together with their associated memories.

Our goal is to realize a speech recognition system using custom designed integrated circuits for the feature extraction block and for the pattern matching one. With the addition of a low-cost microprocessor the complete system will consists of only three chips with additional memory IC's.

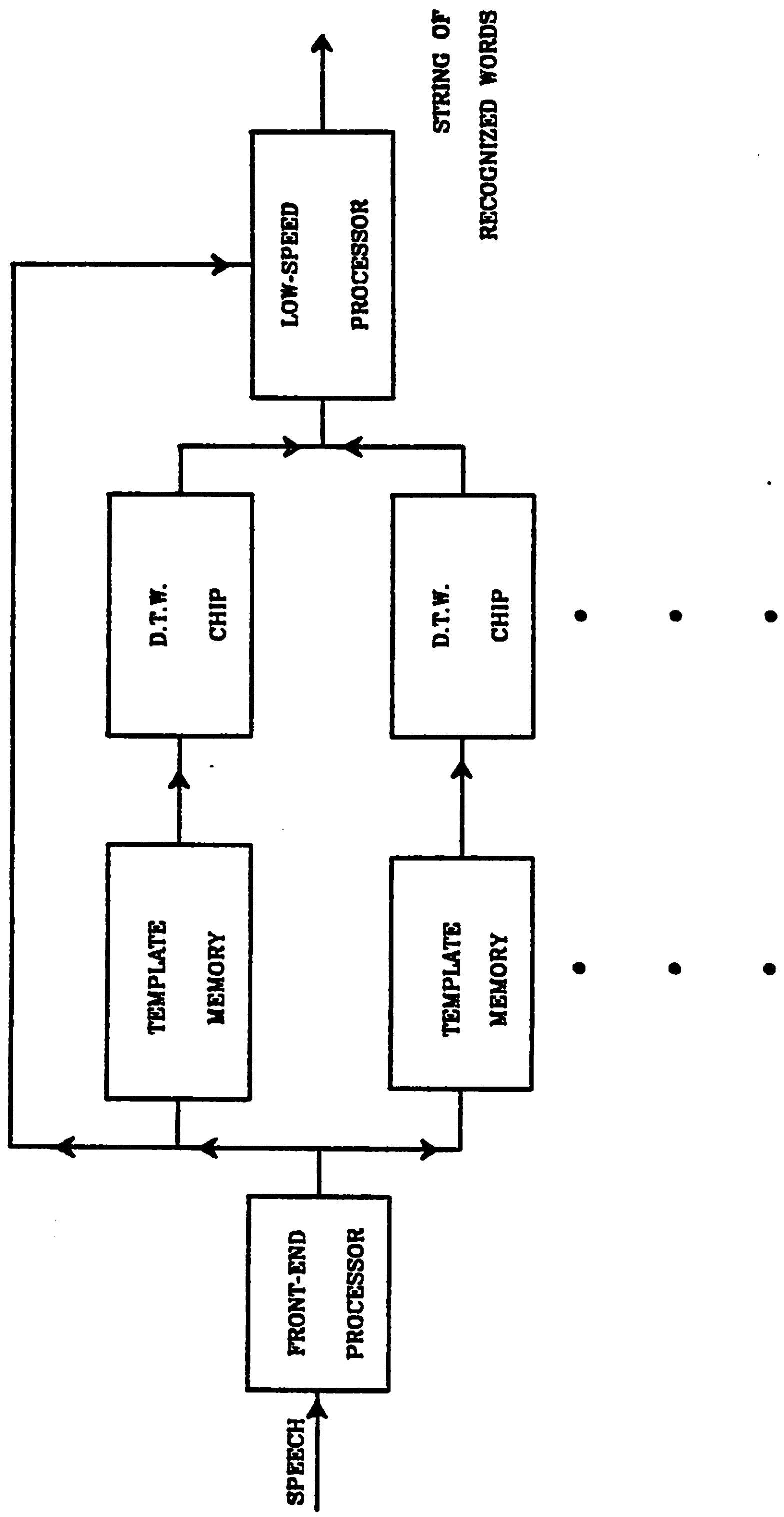


Fig. 1.3 Speech recognition system using custom-designed I.C.'s.

The input to the system is the acoustic waveform of the spoken utterance. This model performs the following basic steps:

a) The feature extraction of the input waveform is done in the ***front end*** part of the system. One or more coded representation of the input utterances are stored as templates in the system memory during the training phase.

b) The ***pattern matching*** part determines the similarity between the incoming word and the templates. This determination is based on the "distance" obtained between the incoming word and each reference template in the vocabulary of the system. The smaller this "distance", the more similar the reference is to the incoming word.

c) Based on the number of templates stored for each word, different ***decision rules*** are applied to determine which templates most closely matches the incoming word.

The main problem encountered when matching patterns representing the incoming words to those representing the words in vocabulary is the correct time alignment of the different parts of the incoming word to those of the reference one (the template). This non-linear time alignment is necessary due to the variation of pronunciation of the same word at different times. To obtain high accuracy we use the dynamic time warp algorithms to optimally time align the incoming patterns to the stored ones. However these algorithms require a large amount of computation. The processing of the algorithm, the large vocabulary and the capability of real time recognition all combine to require a rate of computation of the order of 30 million additions/sec. .

Realizations of speech recognition systems [24],[25] use general purpose processors to compute the D.T.W. algorithms. Fig. 1.2 shows the basic block diagram of a typical speech recognition system using a general purpose

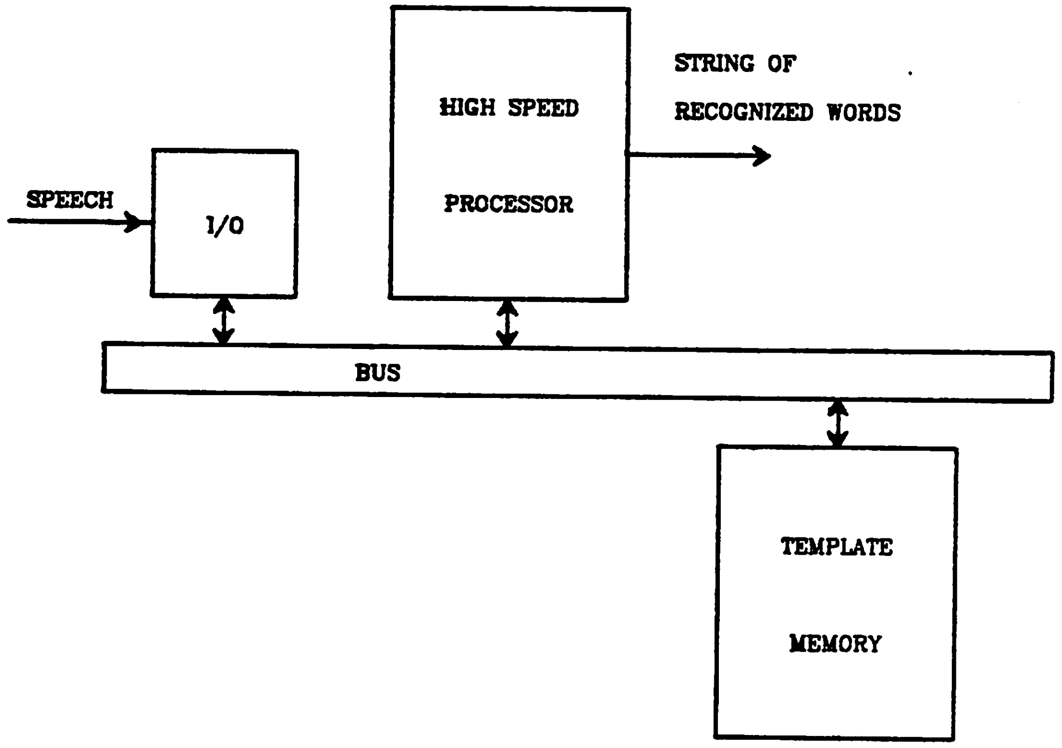


Fig. 1.2 Standard architecture of a speech recognition system

1.5. Description of the Thesis

An overview of the system is given in Chapter 2. Chapter 3 discusses the different implementations of the feature extraction circuit and different structures for realizing the filters in digital form. Chapter 4 describes the operation of the dynamic time-warp circuit and the different cells used in the realization of the chip. Chapter 5 lists the modifications and additions done on the integrated circuit which further integrate the recognition system.

CHAPTER 2

Overview of the Speech Recognition System

This chapter describes the basic considerations affecting the design of the speech recognition system. The main parts of the system are shown in Fig. 2.1.

2.1. Feature Extraction Subsystem

The main operations of the feature extraction block, as described in Fig. 2.2, are:

(1) pre-conditioning of the input signal by amplification and lowpass filtering

(2) characterization of the short-time energy spectrum based on band-pass filters and energy averaging

(3) logarithmic compression of the samples. This reduces the data rate in the system and simplifies the normalization procedure.

(4) amplitude normalization. This is done by dividing each feature by the overall energy of the speech signal.

(5) realization of the end-point detection algorithm. For correct recognition, the system finds the beginning and ending of the utterance by separating speech from background noise.

In [26], different techniques have been proposed to analyze speech. From these, different features set have been obtained. The feature sets most frequently used to represent speech are those based on linear predictive analysis (LPC) and on the filter-bank analysis.

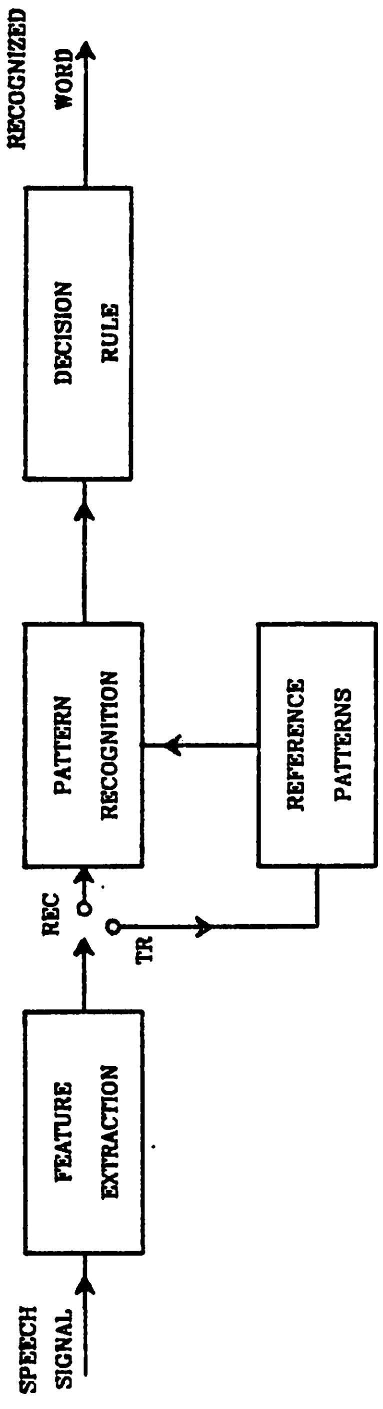


Fig. 2.1 Block Diagram of the system

White and Neely have shown in [3] that the use of either feature set results in similar recognition accuracy. They have found that an LPC analysis of the fourteenth order is equivalent to a set of 20 one-third octave band-pass filters. Each type of analysis used its appropriate metric or "distance measure" .

In the LPC based feature set, speech is represented as a series of frequency-smoothed power spectra. The speech samples forming the power spectra are obtained as a linear combination of past speech samples. By minimizing (over a finite interval) the squared differences between the actual samples and the linearly predicted ones, a unique set of predictor coefficients is obtained.

In the filter bank analysis of speech we use the output of a bank of filters to characterize the short-time energy spectrum of the waveform.

Our system uses a bank of bandpass filters to obtain the set of features. The reasons for this choice are:

a) the distance measure, characterizing the dissimilarity between speech segments, is simple to implement.

b) filters are easy to simulate, design and implement either in digital or analog (switched capacitor) form.

c) bank of filters are available in integrated form [27].

2.1.1. Filter bank Analyzer

The time-varying spectra of the acoustic waveform is obtained from the short-time energy of the speech signal. Because the properties of speech change relatively slowly with time, short segment of speech signal can be processed as if they were short segments from a sustained sound with fixed properties.

The feature extraction operation transforms the large amount of raw data into a smaller set which represents as faithfully as possible the identifying properties of the acoustic waveform. This smaller set of features, representing discrete time segments of speech, serves as a basis for comparing the time segment of the spoken utterance to those of stored templates.

After preamplification and lowpass filtering, the speech signal is first linearly filtered by a bank of bandpass filter to characterize the spectrum in different frequency bands. The output of the spectrum analyzer is sampled every 10 msec. to obtain a set of features representing the input waveform during this time interval.

The disadvantage of the average magnitude computation is that the dynamic range is approximately the square root of the dynamic range for the standard energy computation given by:

$$E_n = \sum_{m=-\infty}^{\infty} x^2(m)\, h(n-m)$$

Because the short-time energy overemphasizes large levels and because of ease of implementation, we use the short-time average magnitude [28]. defined as:

$$M_n = \sum_{m=-\infty}^{\infty} \left| x(m) \right| \cdot w(n-m)$$

where: $x(m)$ represents a sample of the speech signal, and $w(n)$ is a window positioned at time of sample index n.
This process is illustrated in Fig. 2.3 . These operations are realized using bandpass filters, full wave rectification and a lowpass filter representing the windowing operation.

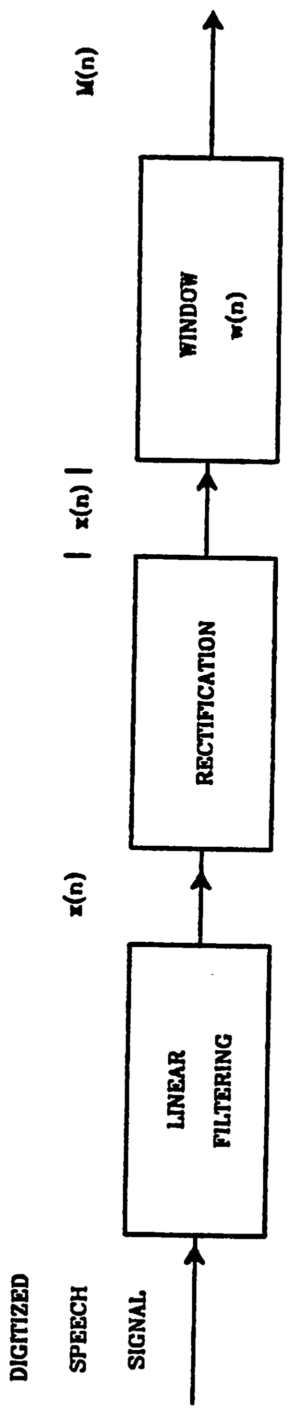


Fig. 2.3 Computation of the short-time average magnitude

2.1.1.1. Definition of the Bandpass Filters

The number of frequency bands in the spectrum analyzer depends on two conflicting requirements: accurate representation of speech spectrum on one hand and minimizing computation and memory requirements on the other. The continuously changing pitch frequency imposes another constraint on the choice of filters. Each channel is integrated during a short time interval which is unrelated to the pitch frequency. Therefore the energy in the bandpass filters fluctuates as a particular pitch harmonic moves in or out of the filter passband. This fluctuation is severe in narrow band filters where the energy may change considerably from frame to frame when in fact it should represent the same average magnitude. Therefore there is a limit to the minimum width we can use in the bandpass filters.

Accurate representation of speech spectrum depends on:

(1) the extend of frequency coverage desired

(2) the number of filters in the analyzer, which depends on the bandwidth of the individual filters.

In addition, the computation and memory requirements also depend on:

(3) the speech sampling frequency

(4) the order of each filter

(5) the accuracy of filter realization.

(6) the desired dynamic range of the filters

The selection of the bandpass filters is frequently done by duplicating the frequency response of the ear.

The ear makes a crude Fourier analysis of an acoustic signal - much as a set of contiguous bandpass filters does. This frequency analyzing behavior is reflected in the mechanical response of the basilar membrane. The basilar membrane in the inner ear exhibits nonsymmetric (and also nonlinear) selectivity at different frequencies [29]. Fig. 2.4 in [30] shows the variation of the equivalent filter bandwidth of the ear as obtained from measurements on:

(1) critical bandwidths [31]

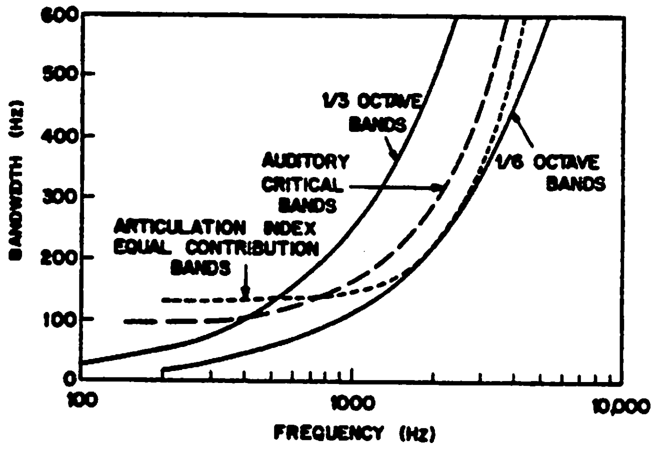


Fig. 2.4 Measurements on the selectivity of the ear

(2) bands of equal contributions to articulation index [32]

and comparison with 1/3 and 1/6 octave bandwidths curves.

From these measurements, the ear seems to behave like linearly spaced filters with a bandwidth between 100 and 140 Hz up to a frequency of 800 to 1200 Hz. From that frequency and up, the filters have logarithmic spacing with bandwidths between 1/3 to 1/6 octave.

Other factors which influenced the choice of filters in our system are:

(1) Good recognition results obtained by White, [3], using third-octave filters up to 10 kHz. His comparison with other recognition systems indicates the necessity for good high frequency discrimination.

(2) possibility of using commercially available switched-capacitor bandpass filters.

In the present system we are using 12 1/3 octave fourth-order Butterworth filters. The bandwidths of the bandpass filters used in the present system appear in Table 2.1. The frequency range spans the band from 200 Hz to 5 kHz. The first filter is one octave wide. The reason is that using 1/3 octave filters in that range would result in filters narrower than the spacing between pitch harmonics. Pitch variations would then give unconsistent energies in that band.

Preliminary investigation [33] has shown that there is little difference in recognition accuracy when using filters having two or three complex poles pairs. However when using one complex pole pair filters, a noticeable degradation in recognition accuracy occurs.

2.1.1.2. Definition of the Lowpass Filter

The window w(n) integrates the incoming signal to yield the slow changes, - the envelope - of the speech spectrum. The duration of the window w(n) is

filter no.	f_l [Hz]	f_u [Hz]	bw [Hz]
1	200	400	200
2	400	500	100
3	500	630	130
4	630	800	170
5	800	1000	200
6	1000	1250	250
7	1250	1600	350
8	1600	2000	400
9	2000	2500	500
10	2500	3150	650
11	3150	4000	850
12	4000	5000	1000

Table 2.1 Frequency Bands of the Analyzer

chosen to be short enough to respond to rapid amplitude changes but long enough to provide sufficient averaging for obtaining a smooth energy function. The duration of the impulse response should be on the order of one to two pitch periods. Pitch frequencies, however, vary for different speakers. The average fundamental pitch frequency is 120 Hz for men, 225 Hz for women and 265 Hz for children. The total range of fundamental frequencies encountered in speech extends from about 60 Hz to about 500 Hz.

It is obvious then that one window will not have adequate duration for all speakers. A compromise is reached for a filter having a duration of the order of 10 to 20 msec. Another solution is to realize different filters for male and female users. For digital filter realization of the windows, the additional coefficients require only a few more memory locations, depending on the order of the filter.

In our system, the window is realized as the impulse response of a three pole, lowpass filter having a cutoff frequency of about 25 Hz.

2.1.2. Logarithmic Compression and Amplitude Normalization

The samples obtained from each channel are passed through a logarithmic compressor. This reduces the dynamic range of the signal and simplifies the normalization procedure. The circuit realizing this operation is a 256 μ-law integrated coder. Each time frame is then characterized by a set of N numbers $M_n^{(i)}$ for i = 1,2,...,N where each number is proportional to the log energy of the speech signal in a given frequency band. N , the number of channels in the spectrum analyzer is typically between 12 to 20. The system uses at present 12 channels.

As the log energy depends on the loudness of the speech, each number in the set must be normalized before using it for comparison. The normalization is done for every frame by subtracting from each feature the average sum of the features. This is equivalent to dividing the energy of the speech waveform in that band by the geometric average of the total energy present during that time frame and then compressing logarithmically. The justification for this operation is the following:

In order to compare two log frequency spectra, L_1 and L_2 respectively, we have to find a constant a minimizing the squared distance between them.

Defining the distance between the two spectra as:

$$D(\, a + L_1, L_2) = \sum_{i=1}^{p} [\, (a + l_1^{(i)}\,) - l_2^{(i)}\,]^2$$

where p is the the number of features ,12 in our system, and $l_1^{(i)}$ and $l_2^{(i)}$ are the i th feature appearing in the log energy of L_1 and L_2 respectively.

For minimum D:

$$\frac{\partial D}{\partial a} = \sum_{i=1}^{p} 2[(a + l_1^{(i)}) - l_2^{(i)}] = 0$$

from which we obtain:

$$a = \frac{\sum_{i=1}^{p} l_2^{(i)} - l_1^{(i)}}{p}$$

The constant a is then the difference between the average of the log features of the spectra. The normalized features which we use in our computations of pattern matching are then of the form:

$$f_1'^{(i)} = f_1^{(i)} - \frac{\sum_{i=1}^{p} f_1^{(i)}}{p}$$

In practice, a constant is added to the normalized features to place it in the numerical range desired.

2.1.3. Endpoint Detection

A word recognition system is basically based on pattern recognition. Therefore errors in endpoint location will result in errors in the recognition process.

When the background noise is significantly lower than the weak fricatives, the location of the beginning and the end of an isolated utterance can be determined by the use of a simple energy threshold test and temporal constraints on the word.[34] This ideal condition very rarely exists. In practice, the utterance is corrupted by:

(1) background noise, like machinery hum, door slamming, etc.

(2) speaker generated artifacts like clicks, lip smacking and breathiness.

These types of noise add pulses at the beginning and at the end of the word making it difficult to determine the correct endpoints.

The basic endpoint detector is based on the double threshold technique [35]. The actual circuit has been modified by Davies and is described in [36]. The operation of the algorithm is shown in Fig. 2.5 . Two thresholds designated T_1 and T_2 respectively are defined at the beginning of each training and recognition session. For frames to belong to speech, they must be above the two thresholds for a certain amount of time larger then the minimum word width. The gap allowed between two such pulses, if they belong to the same word, should not be larger than the maximum gap time. A counter then looks back and selects as the beginning point the first frame having risen above T_1 for which these conditions are true. The ending point is at that first frame falling under T_1 for which the next pulse rising over T_2 is at a distance greater than the maximum gap time.

Heuristic considerations are used to choose the time durations and the thresholds appearing in the algorithm. We use values of 3 db and 10 db above the background noise for the first and second threshold respectively. The minimum word width is 80 msec and the maximum gap between pulses belonging to the same word is 180 msec.

Improvements can be added to this algorithm to discriminate against certain types of noise. To reduce the effect of breath noise, for example, the slope between the two thresholds is constrained by the use of another counter.

Detecting the ending of a word is complicated by:

(1) weak fricatives with low trailing slope like the ending sound /*v*/ as in *five*.

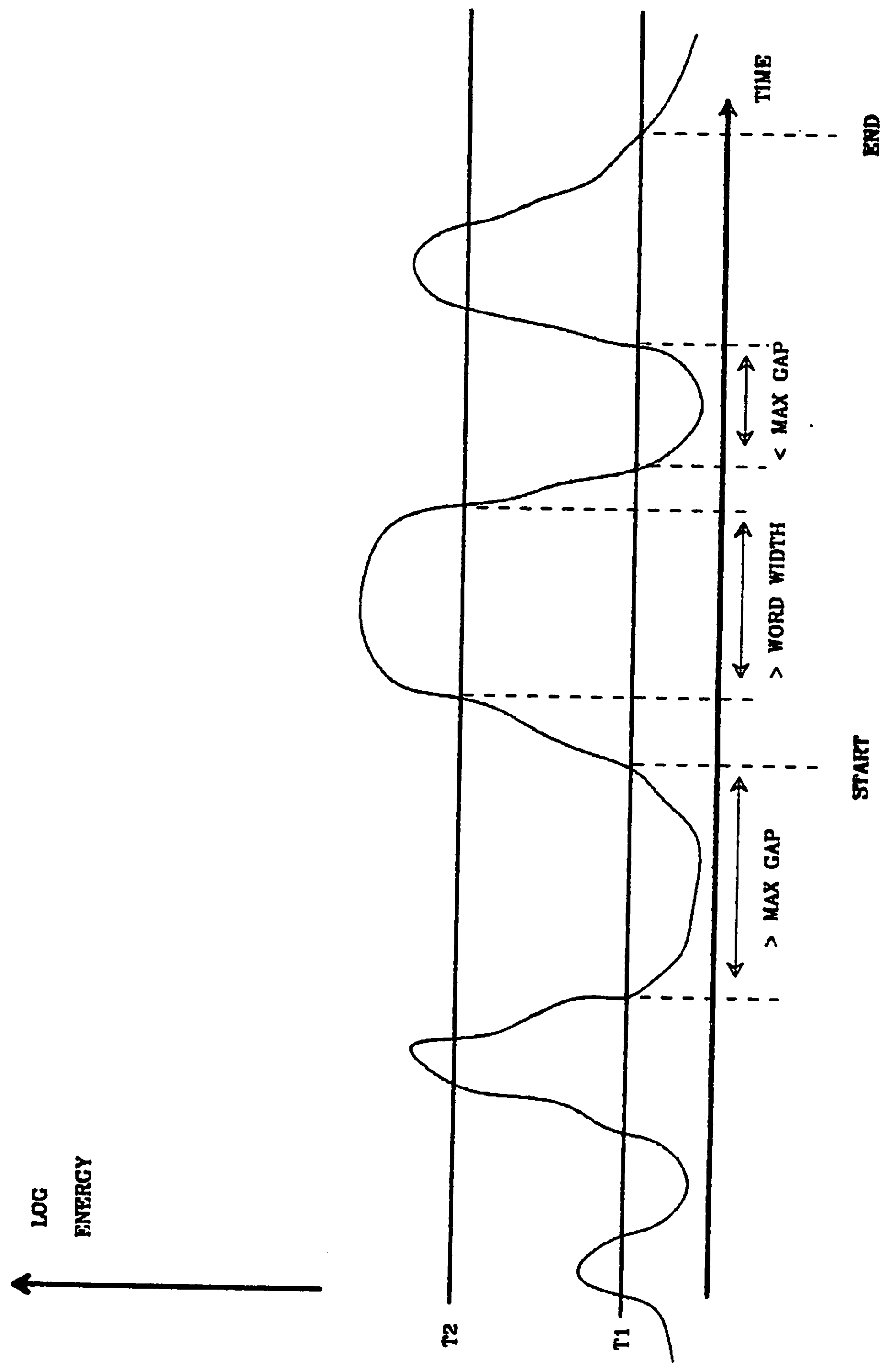


Fig. 2.5 Endpoint detection algorithm

(2) breathiness added at the end of the word.

In the first case the word tends to be shortened as the endpoint detection algorithm will delete the trailing end of the word which blends into noise. In the second case the opposite effect occurs as the breathiness will be included within the endpoints.

The problems mentioned above can be reduced by:

(1) extending the range of frequency of the bank of filters to provide better characterization of the input waveform

(2) improving the endpoint detection algorithm to include differentiation between voiced, unvoiced and silence sounds

(3) incorporating temporal clues in the algorithm.

Including elements of the hybrid technique as proposed by L.Lamel [37] improves endpoint detection. When using this technique, the endpoint detector supplies several estimates for each endpoint with the final decision as to the "correct" endpoints being made by the recognition stage. Alternatively, feedback can be used from the recognition stage to obtain a revised estimate of the endpoints.

2.1.4. Downsampling

After endpoint detection the normalized filter values are either stored in the system dictionary (the template memory) during the training mode, or sent to the pattern matching circuit during the recognition mode. To decrease the data rate through the system, frames which are similar to the previous one are not transmitted. Also the previous frame is not repeated. In this way the system dynamically varies the spectral sampling rate. This method increases the accuracy of the system by reducing the emphasis the system would otherwise place on long steady sounds during pattern matching. Reducing the frame rate also increases the size of the vocabulary that can be processed in real time. Another advantage is that more words can be stored in the vocabulary of the system.

This selective downsampling is explained in more detail in [38]. Basically, a new frame is transmitted only if the spectral distance between it and the previous frame is greater then a predefined threshold. With the threshold at its present value, the spectral sampling rate is decreased from once every 10 msec. to about every 23 msec. when averaged over long periods of speech. For short periods the spectral sampling rate can be as fast as the original one or very low as during steady vowels.

2.2. Pattern Matching

The output of the endpoint detector is a string of N dimensional vectors. Each vector characterizes the state of the vocal tract during a short time interval of the order of 20 msec. This string of vectors represents a pattern. During the training phase, these patterns are the references or "templates", which are stored in the memory of the system. During the recognition phase, the incoming utterance must be matched against each template. The template having the smallest "distance" (most similar) to the incoming utterance is selected as the recognized word.

The success of this operation is dependent on three steps:

(1) correct pattern of the utterances

(2) efficient distance computations

(3) flexible matching procedure.

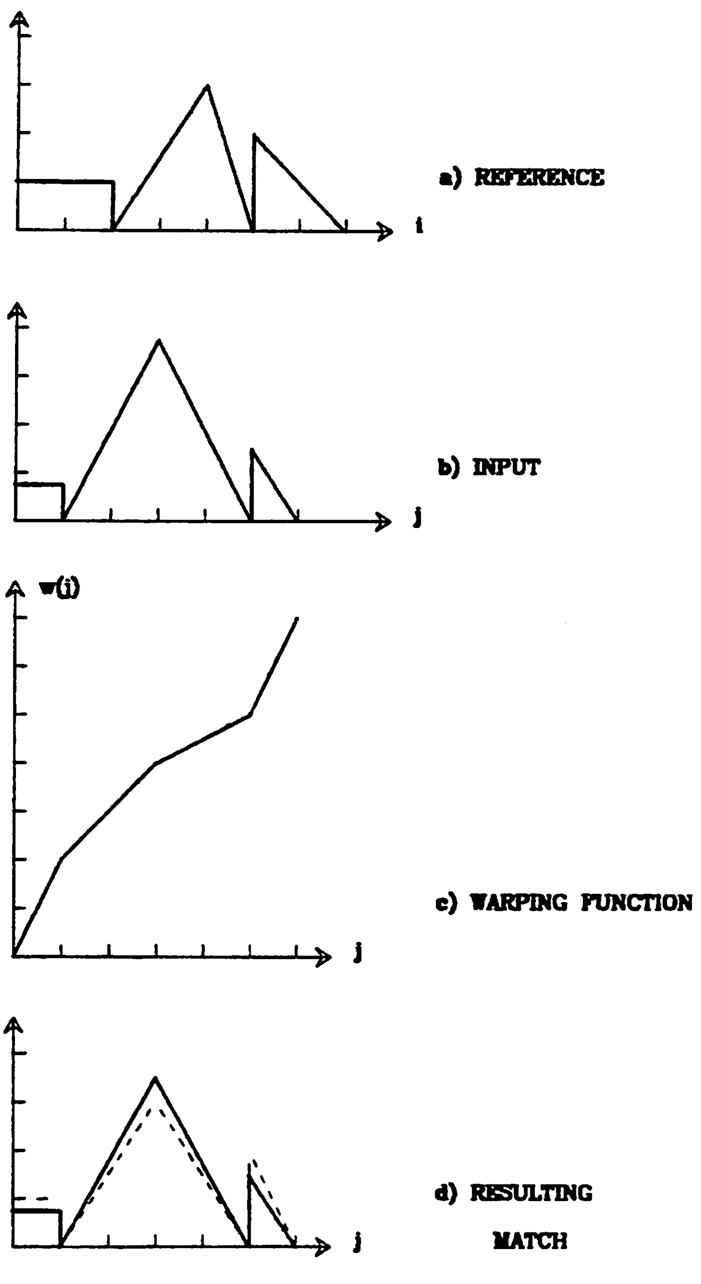


Fig. 2.6 Time alignment between Test and Reference patterns .

The first step is dependent on feature extraction and endpoint detection algorithms. The second and third steps involve both distance computation and time alignment. These operations are strongly related and are done simultaneously.

The main problem in matching the speech patterns is that due to varying speech rates, the repetition of a word has parts which have different duration than their corresponding parts in other repetitions. The repetition may have parts which are compressed and others which are expanded as compared to the original word. The problem is illustrated in Fig. 2.6. To compare the two patterns it is therefore needed to align the corresponding parts of the words using a non-linear time alignment. Sakoe and Chiba proposed the use of dynamic time warping to improve the fit between references and test patterns [2]. In this method, a nonlinear expansion and/or compression of the time scale is used to provide an optimal fit between the patterns. Sakoe and Chiba also suggested the use of dynamic programming, as developed by Bellman [39], for the efficient implementation of the time warping algorithm.

Dynamic programming is a method for finding efficiently an optimal path between two points. This path is determined by minimizing a weight function. The operation of the dynamic programming (DP) algorithm can be best explained with reference to Fig. 2.7a.

The optimal path between the two points A and P is found by applying the principle of optimality [39] stating that the globally optimal path is also locally optimal. Therefore starting from point A we find the best (local optimum) path to point E by comparing the two paths existing from A to E . That is comparing 5 + 2 with 8 + 3 and choosing the path with the smaller weight. The weight at point E is then 7 with the path passing through point B. The optimal path from A to H is obtained by comparing the sum of the paths from A to D and from D to H with the accumulation of the paths from A to E and from E to H. As all the accumulated weights of these paths are known, the optimal path to H is easily found. The accumulated weight at point H is 9 with the path passing trough point E. In this way the optimal path to every point in the matrix is successively found until we reach point P.

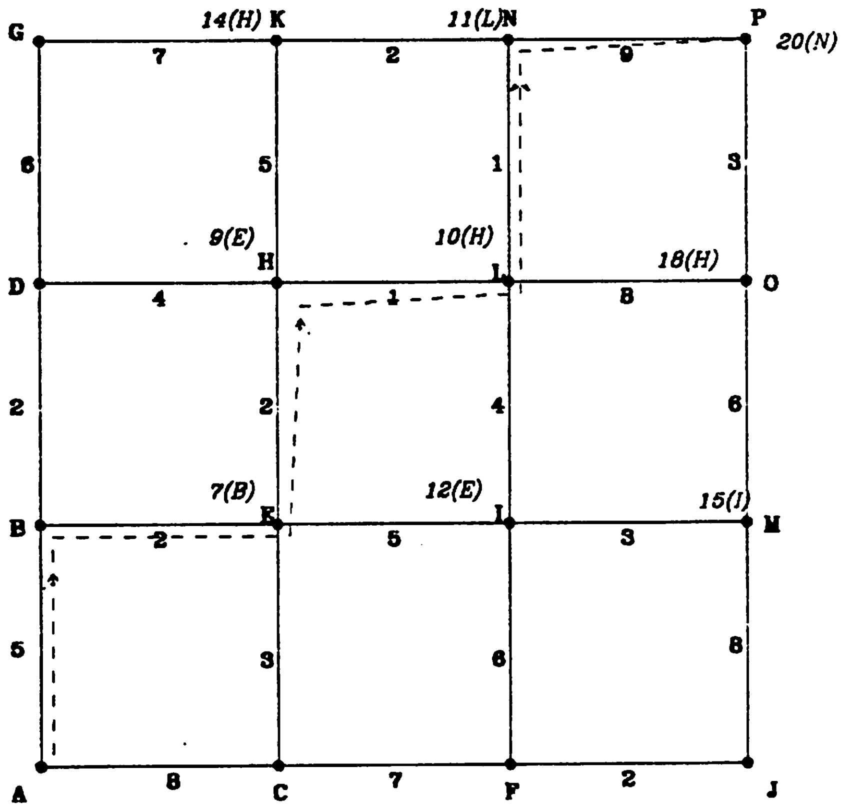


Fig. 2.7a. Principle of operation of the dynamic time-warp algorithm.

This method is applied for the time alignment of speech patterns as shown in Fig. 2.7b :

Each time interval, during which the speech is considered to remain stationary, is represented by a vector of 12 features. The horizontal segments represent the time intervals - or (time) frames - of the incoming pattern denoted by R(m), $0 \le m \le M$. The vertical segments represent the frames of the reference pattern denoted by T(n), $0 \le n \le N$. The weight attributed to each cell in the two dimensional grid is obtained by computing the local distance between the corresponding reference frame [R(m)] and (incoming) test frames [T(n)].

The distance between the two patterns is the sum of the local distances along the optimal time-alignment path. The optimal time-alignment path is a function relating the m time axis of the reference pattern to the n time axis of the test pattern and has the form:

$$m = w(n)$$

where w(n) is restricted to begin at the point n=1, m=1, to pass through the grid of points (n,m), where n and m are integers, and to end at the point n=N, m=M. The required function is the solution to the minimization problem posed by:

$$D = \min_{w(n)} \left[\sum_{n=1}^{NT} d(\, T(n), R(w(n))\,) \right]$$

where d(T(n),R(w(n))) is the "distance" between frame n of the unknown pattern, and frame w(n) of the reference pattern.

This function is found using the dynamic programming technique. Fig. 2.7b shows the matching between the patterns of two repetitions of the word " four" having different durations. The numbers in parenthesis indicate the (local) distance between the corresponding frames of the reference and incoming test patterns. The optimal path, found using the dynamic programming method, passes through the cells indicating the best match (smallest distance) between the segments of the reference and test patterns. The distance between the two patterns is obtained by summing the local distances along its path.

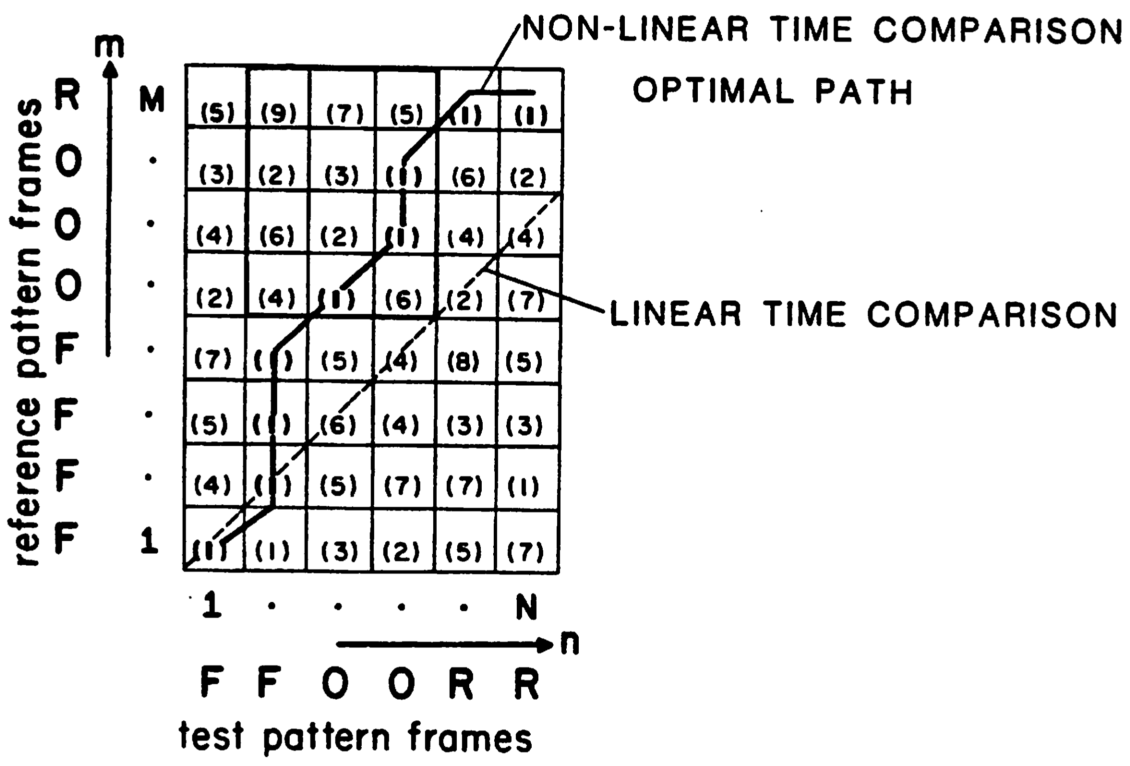


Fig. 2.7b Speech pattern alignment using DTW algorithm.

2.2.1. Distance Computation

The distance between two patterns depends on:

(1) the time alignment

(2) the measure we use to compute the distance between them.

The "distance" between the two patterns is defined as the accumulation of the spectral distances between their aligned frames. Several measures have been proposed for calculating the spectral distances [40]. Among the most widely used are the LPC log likelihood distance for LPC based analysis and the Euclidean distance. The distance used depends on the features set chosen to describe the speech waveform.

As described earlier, we use the following as measure of distance between two spectra of energy S_A and S_B :

$$d(S_A,S_B) = \sum_{\omega} \left| \ln S_A(\omega) - \ln S_B(\omega) \right|^2$$

The features we are using represent the in-band log energy. Substituting the unknown and reference patterns in the distance measure we obtain the square of the Euclidean distance:

$$d(T,R) = \sum_{i=1}^{i=16} [l_A^{(i)} - l_B^{(i)}]^2$$

where $l_A^{(i)}$ and $l_B^{(i)}$ are the normalized log energy estimate of the i^{th} bandpass filter of spectra A and B respectively. We have found that euclidean distance gives generally better recognition accuracy then the distance based on absolute value of the difference between features [3].

This distance is computed between every frame of the unknown pattern and every frame of the reference pattern. Thus for N frames in the unknown and M

frames in the reference pattern, we have NM calculations when comparing the test pattern with one reference pattern. For a vocabulary consisting of W reference words, the number of distance calculations is WNM. This distance calculation is the step requiring the most computation in the word recognition system.

2.2.2. Dynamic Programming for Time Warping

Dynamic programming uses a succession of simple operations to find the optimal path. Beginning from frame (1,1), the optimal path to a point in the grid comes from a previous point for which the accumulated distance is minimum. Thus, if D(n,m) represents the minimum accumulated distance to the point (n,m) we have:

$$D(n,m) = d(T_n,R_m) + \min_{q \leq m}\left[D(n-1,q), D(n,m-1)\right]$$

where: $D(n,m)$ is the minimum accumulated distance to the grid point (n,m), $d(T_n,R_m)$ is the local distance between these two frames as described in the previous section and q is the number of points from which a permissible path exists to the point (n,m).

Many variations of the basic algorithm have been proposed [41].Local and global constraints on the path have been added to reduce the amount of computation. To obtain a better fit, the endpoint constraints have been modified to take into account conditions appearing during generation of speech patterns[42].

We are using the algorithm shown in Fig. 2.8 . The minimum accumulated distance to any point (n,m) is given by:

$$D(T(n),R(m)) = d(n,m) + \min\left[D(n-1,m), D(n-1,m-1), D(n,m-1)\right]$$

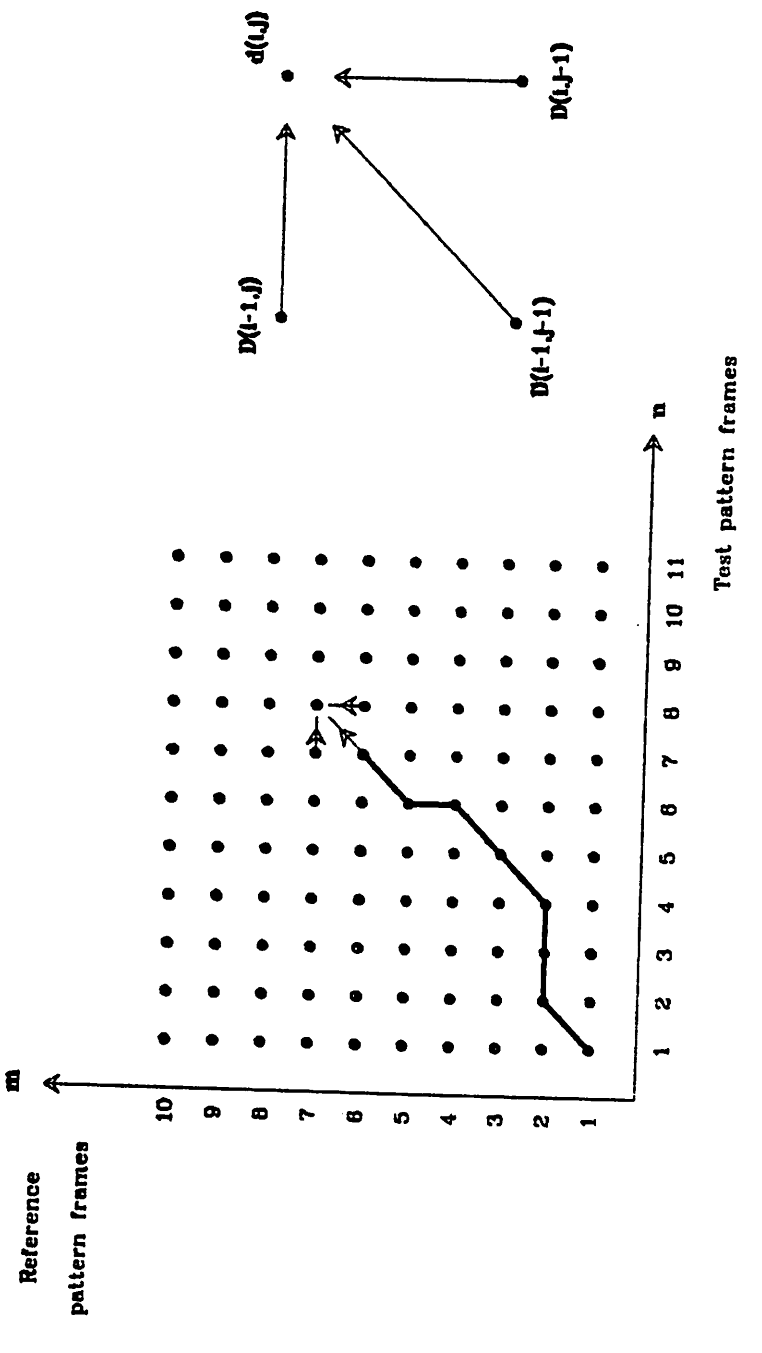


Fig. 2.8 Dynamic time warp algorithm

where D(n-1,m) ,D(n-1,m-1) and D(n,m-1) are the accumulated distances, along an optimal path, at grid points adjacent to the point (n,m). There is no need to find the optimal path but only the accumulated distance along it.

This computation is done for every point in the grid, up to the last point *(N,M)* and is repeated for every reference word in the vocabulary. The reference pattern which has the smallest accumulated distance to the incoming pattern is selected as the incoming word.

Justification for using this algorithm is the following: although the incoming pattern is time warped when refered to the correct reference template, the order of the characteristic frames remain the same. Therefore the optimal path which matches the corresponding frames can only increase monotonically The path can continue only along the three directions as indicated by the algorithm.

When computing the j^{th} column the values of D (i,j-1) , $1 \le i \le M$, are needed. Therefore the amount of memory required by this algorithm for the computation of one path through the grid is one column and one cell of memory. Previous values of accumulated scores are discarded.

An example of the computation of the accumulated score for two successive frames in the same column is shown in Fig. 2.9a. In each cell, the numbers in parenthesis indicate the local distance whereas the upper number indicate the accumulated distance along an optimal path up to that point. through the grid. An example showing the optimal path and all the accumulated distances in the grid appears in Fig. 2.9b.

2.2.3. Time Normalization

To compare correctly the accumulated scores obtained for the different templates, it is necessary to normalize them to the same time length. This

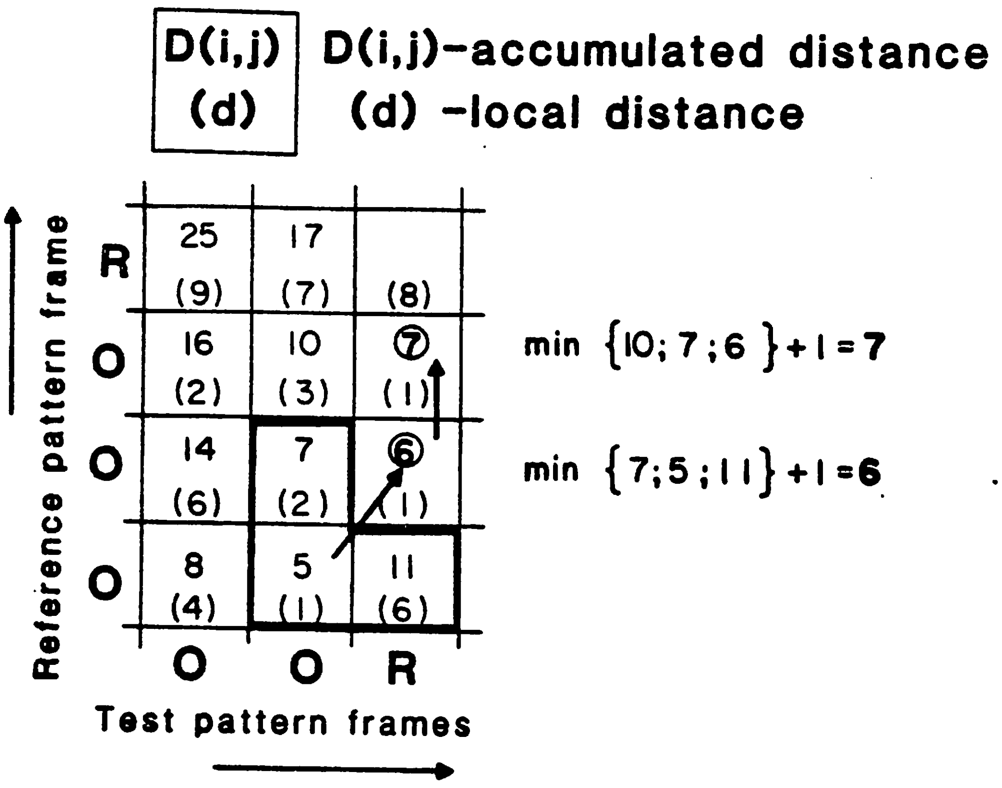


Fig. 2.9a Computation of the accumulated distance

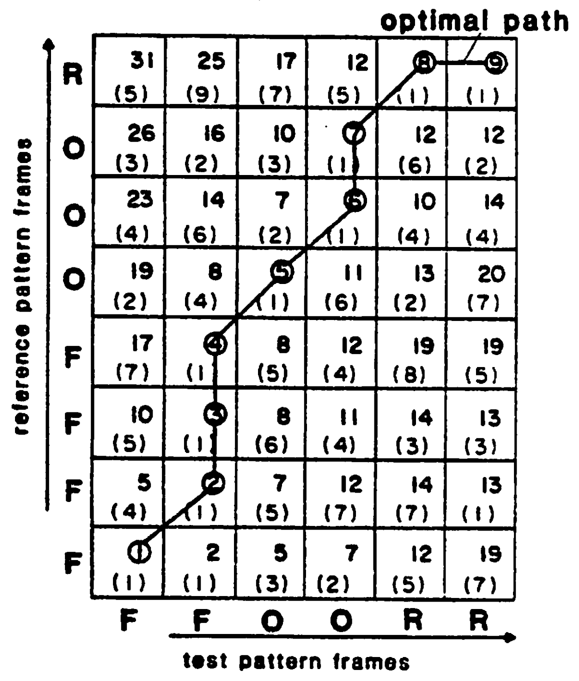


Fig. 2.9b The optimal path

normalization should be done by dividing the accumulated score by the length of the optimal path. However, to simplify the computation, the length of the reference pattern is used as measure of normalization. The recognized pattern is then selected by comparing the different $\frac{D_N}{L_N}$ where D_N is the accumulated score obtained with reference template N and L_N is its length, in number of frames. Experiments done on different normalization measures [38] show no difference in recognition accuracy.

2.2.4. Decision Procedure

To reduce the probability that an unwanted sound may trigger the recognition system and that the smallest normalized distance, although incorrect, will be recognized, a numerical threshold is defined. Any normalized distance above this threshold will be rejected. The threshold should be adjusted to obtain a good comprise between the number of errors and between the number of rejection. This operation, and others like I/O, is performed by the microprocessor in the system and is adjustable by the user.

2.3. Compatibility with Connected-Speech Recognition Algorithms

Connected-speech recognition systems are more attractive than isolated-word reoognition one's since they are easier to use. The recognition system described here is also capable of recognizing connected speech. The feature extraction subsystem is used for both isolated words and connected speech. The reference templates are the same ones that are used for isolated word recognition. The endpoint algorithm detects the beginning and ending of a string of words. This string of words passes through a dynamic time-warp algorithm similar to the one used for isolated words. The algorithm finds the best string of templates that when concatenated matches the input string better than any other concatenation of templates.

The implementation of the algorithm is describe in detail in [38] and is similar to other algorithms used for connected-speech recognition [43]. Basically the algorithm concatenates the reference templates having the smallest normalized topscores. To normalize the topscores of the string of concatenated templates, the path length of this string must be found. This is done by keeping track of the number of the individual frames in the optimal path. Each frame has now both an accumulated score as in the isolated-word algorithm and a path length associated with it. The normalization and comparisons are done using the standard microprocessor in the system. The circuit realizing the path length is implemented by a counter-like circuit. This circuit is integrated with the circuit implementing the isolated-word algorithm and is described in detail in chapter 4.

2.4. Training Procedure

One or more reference template must be obtained for each word in the system's vocabulary. In a speaker dependent system, this must be done for every user. It is very important to obtain a template which is the typical pronunciation of the word. The quality of the template influences greatly the performance of the speech recognition system. Different training algorithms have been used with the system. Among them are:

- repeating each word only once
- repeating the word several times, typically from 2 to 10 times until the repetitions are similar and chosing that word which is closest to the others

- using more than one template to represent a vocabulary entry
- constructing an artificial template which is the average of the repetitions

These different methods have been tried and are described in detail in [38,44].

CHAPTER 3

Circuitry for Feature Extraction Using Spectral Analysis

3.1. Current Feature Extraction Circuits

The function of each chip in the integrated recognition system has been defined in the previous chapter. The front-end chip has not yet been realized, but its integration is straightforward as most of its functions either have already been implemented or do not pose any problem of implementation due to size or frequency rate. Once completed, this special purpose integrated circuit will replace the feature-extracting breadboard used in the present system. Due to the modifications made in the original circuitry [45], the version of the circuits presently in use is described.

Basically the feature extraction system performs a twelve channel spectral analysis every 10 ms. The output of each channel is averaged by a lowpass filter which has a window of 20 ms duration. The system consists of the following parts:

(1) input amplifiers, pre-emphasis and anti-aliasing filters

(2) bandpass filters

(3) half-wave rectifier and lowpass filter

(4) sample-and-hold circuit and analog multiplexer

(5) logarithmic A/D converter

(6) serial-to-parallel data conversion circuit

Fig. 3.1 shows the block diagram of the system.

3.1.1. Input Amplifiers, pre-emphasis and anti-aliasing filters

The input signal to the feature extraction circuit comes from a low output impedance line driver operational amplifier. The input circuit consists of an input-protected, second order, active lowpass filter with a 3 db cut-off frequency of about 10 kHz. This is followed by an amplifier having an adjustable gain between 1 and 16. The output of this amplifier drives a pre-emphasis circuit. This circuit is needed as the speech signal has a 6 db roll-off characteristic. The pre-emphasis circuit is a high-pass filter with a single zero at 500 Hz and a single pole at 2 kHz. The output of the pre-emphasis circuit drives two anti-aliasing filters. These filters limit the bandwidth of the input signal and cancel the aliases generated by the switched-capacitor bandpass filters. The first anti-alias filter is composed of two second order lowpass active filters realizing together a 3 db cutoff frequency of 1.4 kHz. The output of this filter is connected to the first four channels of the spectrum analyzer. The second anti-aliasing filter is a single, second order, lowpass filter having a 3 db frequency of 10 kHz. Its output is analyzed by the eight other channels of the analyzer.

3.1.2. Bandpass filters

The spectrum analyzer uses twelve bands defined by third order, switched-capacitor, Butterworth filters. The cutoff frequencies of the filters are as indicated in table 2.1 of the previous chapter. Originally the analyzer had 19 channels. Eighteen 1/3 octave filters spanning the range from 100 Hz to 6.4 kHz and one full octave filter indicating the high frequency energy from 5 kHz to 10 kHz. It was soon apparent that the three lowest filters - from 100 to 200 Hz - were too

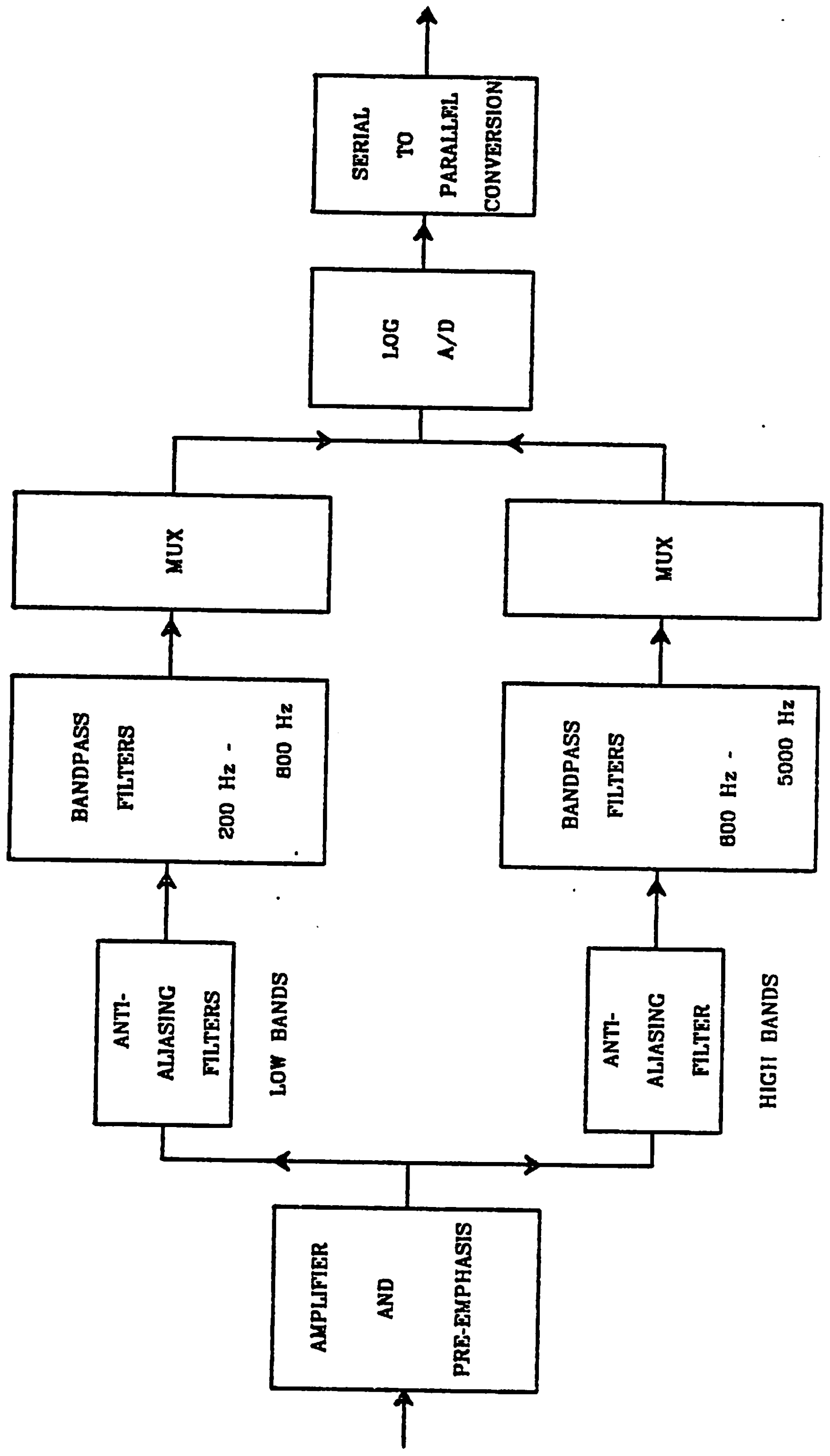

AMPLIFIER AND PRE-EMPHASIS
ANTI-ALIASING FILTERS
LOW BANDS
ANTI-ALIASING FILTER
HIGH BANDS
BANDPASS FILTERS 200 Hz - 800 Hz
BANDPASS FILTERS 800 Hz - 5000 Hz
MUX
MUX
LOG A/D
SERIAL TO PARALLEL CONVERSION

Fig. 3.1 Block diagram of the feature-extraction subsystem

narrow. They were discarded. The three next filters from 200 to 400 Hz were combined into one full octave filter. The high frequency filters above 5 kHz were also discarded as it was found that they did not increase the accuracy of the recgnition system.

Two kinds of integrated filters are used, the first one has one full octave filter per chip and realizes the first filter as a 1/3 octave bandwidth would result in filters being narrower than the pitch harmonics. The second one has three 1/3 octave filters per chip and needs only one clock signal for the three filters [46]. Four such circuits are used to realize the eleven 1/3 octave filters.

The clock signals for the integrated switched-capacitor filters are derived from the master clock used in the data conversion circuit.

3.1.3. Half-wave Rectifier and Lowpass Filter

The half-wave rectifier realizes the absolute value function needed in the computation of the short-time average magnitude of the speech signal while the lowpass filter averages the energy over a duration of about 20 msec.

The output of each bandpass filter is a.c. coupled to an active half-wave rectifier . The output of the rectifier is buffered and connected to a three pole Butterworth lowpass active filter. This filter has been modified. Its actual realization is shown in Fig. 3.2a, while its impulse response appears in Fig. 3.2b.

3.1.4. Sample-and-hold and Multiplexer

The smoothed output of each channel is sampled every 10 msec. by a sample-and-hold circuit using the LF 398 monolithic circuit. The output of each channel is connected to a 16 channel analog multiplexer [47]. The output of the multiplexer is buffered and serves as input to the logarithmic converter.

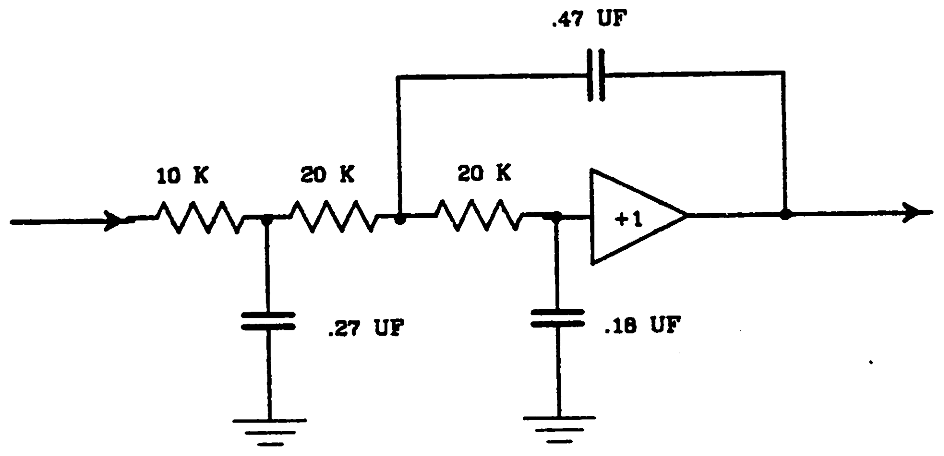


Fig. 3.2a Lowpass filter circuit.

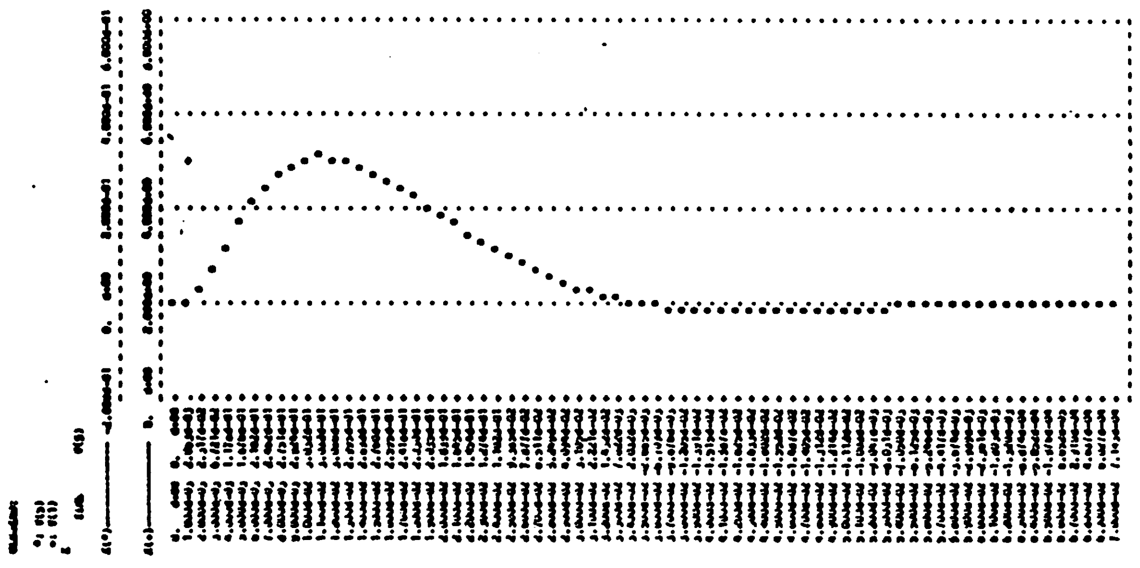


Fig. 3.2b Impulse response of lowpass filter

3.1.5. Logarithmic A/D Converter

The logarithmic compression of the sampled signal was done using a μ-255 law analog-to-digital converter. Eight bits are used to represent the piece-wise logarithmic transfer function of the A/D converter. The first bit is a sign bit. The next three bits indicate one of the eight possible logarithmic chords of the transfer function. The last four bits indicate the linear arc within that chord [48].

3.1.6. Serial-to-Parallel Conversion Circuit

The coder outputs the eight bits representing the speech sample in series. A circuit is needed to obtain this code as an eight bit-word. The eight bits are loaded in a shift register. A counter signal the availability of a new word. Another counter keeps track of the channel being sampled. A flag is issued at the beginning of each new sampling cycle. All the timing signals are derived from one master clock. These signals select the correct channel from the analog multiplexers, provide the digital clock for the A/D converter, and the different clock frequencies for the switched-capacitor filters.

The output of this conversion circuit are eight-bit words representing 10 msec. pseudo-logarithmic samples of the speech signal. These samples are sent to the 11/23 microprocessor for the following operations:

(1) true 4-bit logarithmic representation

(2) amplitude normalization

(3) realization of the end-point detection algorithm

The purpose of these operations was explained in the previous chapter. The details of the operations as done in software is explained in [38].

3.2. Realization of the Future Spectrum Analyzer

Implementation of the spectrum analyzer in digital form is attractive as all the filters can realized by the same circuit using time-multiplexing . The dynamic range of the filters can be tailored to one's need as it depends on the number of bits used in their internal computation. Also additional digital computations can readily be added to the circuit due to the flexibility of digital processing.

Integrated analog-to-digital converters (ADC) needed for such a digital approach have already been realized. Fotouhi [49] has realized a high-resolution integrated ADC realizing 12 bit monotonicity with a differential nonlinearity (DNL) of less than 1/2 LSB. This monotonicity is obtained with only 8 bit ratio accurate circuit elements thereby reducing the area needed by the circuit as compared to other integrated ADC's. The size of the complete ADC, including the logic, is about 15 000 mil^2. The same technique can be extended to obtain resolution greater than 12 bits.

When using this digital approach, the speech should be sampled at a 14 kHz rate. The new spectrum analyzer is designed to have sixteen channels. This increase in the definition of the spectrum is done to enhance recognition accuracy. The sixteen fourth order Butterworth bandpass filters have been chosen according to the behaviour of the ear that is: a constant bandwidth up to about 1 kHz and a constant Q up to a frequency of 6 kHz. The cutoff frequencies of the sixteen bandpass filters appear in table 3.1 .

Digital filters however require multiplications that are time consuming operations and require relatively large area for implementation.

Two approaches are popular to solve this problem. The first one is the ROM-Accumulator type of circuit as proposed by Peled and Liu [50]. The second

filter no.	f_l [Hz]	f_u [Hz]	bw [Hz]
1	75	220	145
2	180	380	200
3	380	580	200
4	580	780	200
5	780	980	200
6	980	1180	200
7	1180	1402	222
8	1402	1677	275
9	1677	2006	329
10	2006	2400	394
11	2400	2793	393
12	2793	3250	457
13	3250	3781	531
14	3781	4400	619
15	4400	5138	738
16	5138	6000	862

Table 3.1 Frequency Bands of the Analyzer

approach is to use a dedicated signal processor.

3.2.1. ROM-accumulator Circuit

The basic ROM-ACC circuit is shown in Fig. 3.3. Its essential feature is that all the multiplications and additions of each sections are performed simultaneously through the use of a ROM together with an adder-subtractor circuit.

Considering the second-order difference equation for a bandpass filter

$$y(n) = b_0 x(n) + b_0 x(n-2) - a_1 y(n-1) - a_2 y(n-2)$$

and assuming that x(n), y(n), a_i and b_i are represented in two's complements

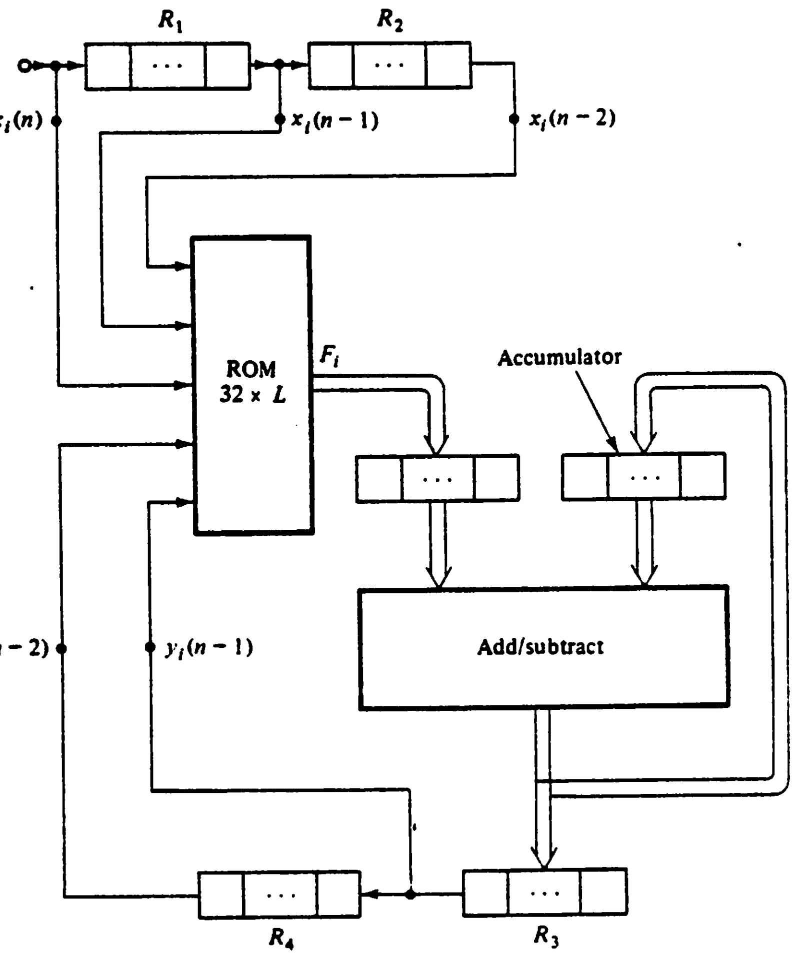


Fig. 3.3 Basic ROM-Accumulator section (after [50])

$$\tilde{x}(n) = x_0(n).x_1(n)x_2(n) \cdots x_L(n)$$
$$\tilde{y}(n) = y_0(n).y_1(n)y_2(n) \cdots y_L(n)$$

where $\tilde{x}(n)$ and $\tilde{y}(n)$ are the two's complement of x(n) and y(n) respectively. Then we can represent the numbers *x(n)* and *y(n)* as:

$$x(n) = -x_0(n) + \sum_{i=1}^{i=L} x_i(n)2^{-i}$$

$$y(n) = -y_0(n) + \sum_{i=1}^{i=L} y_i(n)2^{-i}$$

Equation (1) can now be written in the form

$$y(n) = \sum_{i=1}^{L} 2^{-i}\left[b_0x_i(n) + b_0x_i(n-2) - a_1y_i(n-1) - a_2y_i(n-2)\right] - \left[b_0x_0(n) + b_0x_0(n-2) - a_1y_0(n-1) - a_2y_0(n-2)\right]$$

Or in a more compact form

$$\tilde{y}(n) = \sum_{i=1}^{L} 2^{-i}F_i - F_0$$

The computation of y(n) is done beginning from the LSB (i = L). F_i is evaluated and added to the content of the accumulator. The content of the accumulator is then shifted one bit to the right using a two's-complement shift. This computation does not require any multiplication as the ROM is programmed to output the result of F_i for every combination of *1*'s and *0*s at the input of the ROM. This step is repeated for i = L-1,L-2, ... ,1. Then F_0 is evaluated and is subtracted from the content of the accumulator by a two's-complement subtraction.

The disadvantage of this method is that the number of steps needed to evaluate the output of one filter is equal to the number of bits in the word. Direct implementation of digital filters requires 22 bits to obtain a signal-to-

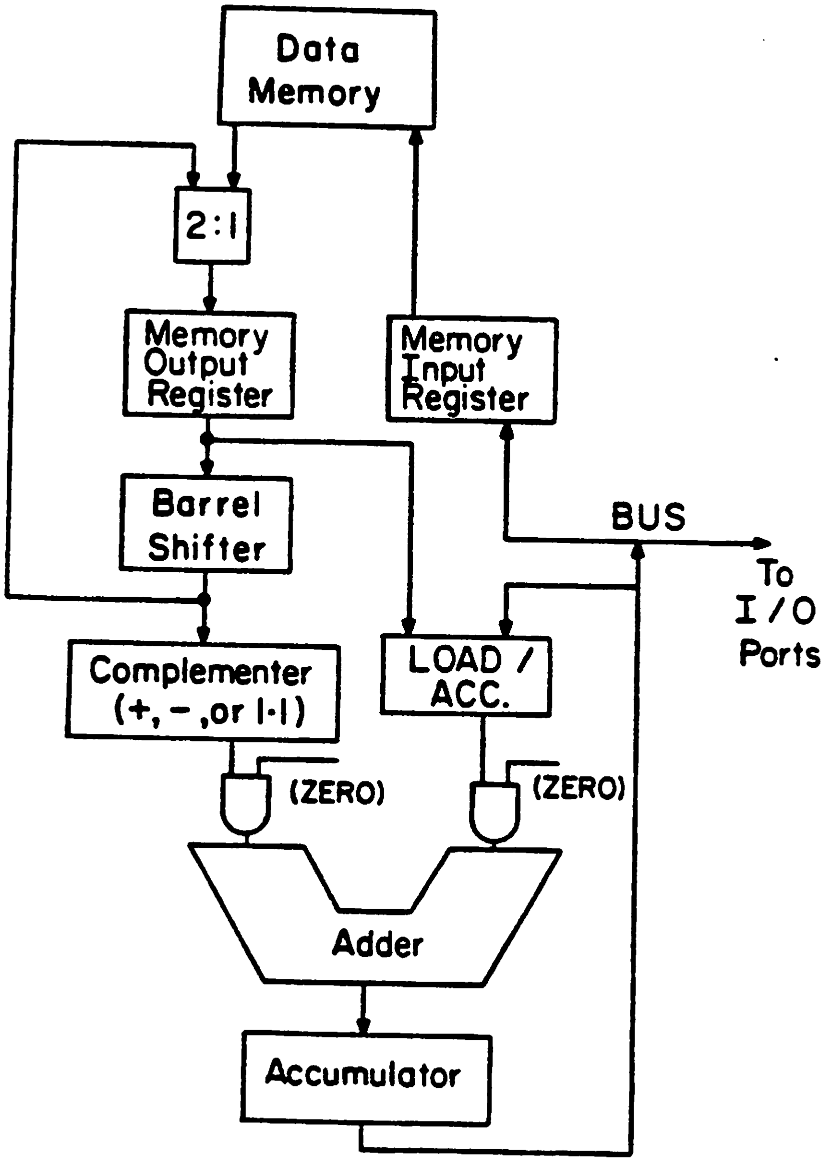


Fig. 3.4 Block diagram of the signal processor.

noise (S/N) ratio of approximately 80 db. Another disadvantage is the size of the ROM. The equation of a fourth - order bandpass filter is:

$$H(z) = \frac{b_0(1 - 2z^{-2} + z^{-4})}{1 + a_1z^{-1} + a_2z^{-2} + a_3z^{-3} + a_4z^{-4}} \tag{3.1}$$

The numerator can be computed beforehand as it is the same for all the filters. For 16 fourth order, bandpass filters we need then 4 inputs to the ROM for the selection of the filter and 5 inputs for the five coefficients of the filter. The size of the ROM is then: $2^9 \times 22 = 11264$ bits.

For these reasons, the approach using a dedicated processor based on a parallel-serial (P-S) multiplier is preferred.

3.2.2. Signal Processor for the Realization of Digital Filters

To implement the short-time spectrum analyzer we are using a processor specially designed for signal processing [51]. The block diagram of the processor appears in Fig. 3.4. The processor consists of a data memory, an arithmetic unit and an I/O section. Signals are represented in the data memory as fixed-point two's complement values.

The arithmetic unit consists of pipeline registers, a complementer, a barrel shifter, an adder and multiplexers and gating circuits. Pipeline segmentation allows concurrent memory access, addition and invert or shift operation. The memory output register (MOR) is a master-slave register which is loaded each cycle with the output of a multiplexer. The multiplexer selects either the output of the data memory or the output of the barrel shifter. The microcode controlled barrel shifter is a parallel shifter array which is used to present the terms of the canonical signed-digit representation of the coefficients to the accumulator. By using the recirculating path from the output of the shifter to the multiplexer, we can reduce the requirements of the depth of the shifter (and hence its size) at the expense of occasional additional clock cycles. The shifter has been selected to have a depth of six as this provides a good compromise between size reduction on one hand and minimum additional clock cycles (for shifts greater than six) on the other. The output of the MOR feeds the complementer which under program control will give the true value or the one's complement or the absolute value of its input. The adder is a saturating adder: if an overflow is detected the output will be the maximum or the minimum value that can be represented, for positive or negative overflow respectively. The output of the adder is loaded into the accumulator (ACC). According to the possible values for the inputs to the adder, the following are the values which can be loaded into the accumulator:

acc := 0

acc:= $[\,+\,,\,-\,,\,|.|\,]\,.\,2^i \times$ MOR for i = 1, 0, -1, -2,...

acc:= $[\,+\,,\,-\,,\,|.|\,]\,.\,2^i \times$ MOR + acc for i = 1, 0, -1, -2,...

acc:= $[\,+\,,\,-\,,\,|.|\,]\,.\,2^i \times$ MOR + MOR for i = 1, 0, -1, -2,...

acc:= acc

acc:= MOR

Additional features are incorporated in the processor as for example provision to allow for a parallel-serial division operation by a "accumulate if positive" control. When this feature is used, the output of the adder is loaded into the accumulator only if it is positive.

The output of the accumulator drives the M-bus, which feeds the memory input register (MIR). This register is a transparent latch, which may either load or hold data under program control. All memory write operations store the output of this register; by holding the latch transparent, the accumulator is written directly into the memory.

All I/O transfers are also through the M-bus. Provided are parallel input and output ports, two serial output ports, and a serial input port.

A multiply operation of a data word by a signed constant is performed in the arithmetic unit by a sequence of shift-and-add operations. beginning with the MSB. This is shown in the following example: Consider an eight bit constant in sign-magnitude format:

-.1100001

First the data is read from memory into the MOR.

MOR := x(n) The multiplication proceeds as in the following microcode segment:

```
acc  := -MOR/2 ; MOR := MOR/2
acc := acc      ; MOR := MOR/2
acc := acc - MOR; MOR := MOR/2
acc := acc; MOR := MOR/2
acc := acc; MOR := MOR/2
acc := acc; MOR := MOR/2
acc := acc; MOR := MOR/2
acc := acc + MOR
```

The accumulator now contains the original data multiplied by the above constant. Thus the multiplication by an externally available coefficient in serial form can be accomplished by routing the coefficient into the appropriate control inputs for the processor. To reduce the number of operations the constant coefficients are represented in the canonical signed-digit (CSD) form [52]. In this scheme, a string of constant of the form:

$\pm k_0.2^{-1} + k_1.2^{-2} + k_2.2^{-3} + \ldots$

where the k's have values of 0 or of 1, is represented as:

$g_0.2^{-1} + g_1.2^{-2} + \cdots$
where the g's have values of ± 1.

This is a pseudoternary representation in which $g_i = 0$ gives a shift, $g_i = 1$ is an add-and-shift, and $g_i = -1$ is a subtract-and-shift operation. The sum of non-zero g_i gives the number of additions and subtractions implied by the coefficient. It is known, [52], that using the CSD representation there is a unique representation for any coefficient such that the sum of non-zero g_i is a minimum, that:
$g_i g_{i-1} = 0$ for all valid i
and that $prob\,(g_i \neq 0) \to 1/3$ as the length of the coefficient becomes large. Therefore the use of the CSD form greatly reduces the number of operations in the processor.

The advantages of the processor approach over the ROM-ACC realization are: flexibility and small size (approx. 5 mm^2) [53] .

(1) Due to the large amount of computation needed for the analyzer, two processors are used to realize the spectrum analysis over sixteen channels in real time. The size of the memory remains the same.

(2) The processor realizes all the operations needed in the spectrum analyzer and others needed in the front-end as well without the need of additional special circuits.

3.3. Bandpass Filter Realization

An efficient realization of the system requires to minimize the number of clock cycles needed for each operation. In digital filter implementations, the multiplication is the most time consuming operation. The time neeed to compute the multiplications depends on the number of non-zero bits in the coefficient words. It is therefore desirable to reduce the number of these bits to minimum while still realizing the desired transfer function. This reduction of the coefficient is done in two steps: in the first one we use digital structures having low coefficient-sensitivity. This enables us to use shorter coefficient words and still maintain a transfer function close to the desired one. A short coefficient word however does not automatically ensure a small number of non-zero bits. To reduce the numbers of non-zero bits in the coefficients, we use the pseudo-ternary notation which makes use of non-zero bits with negative sign, so that the bits can obtain values of =1, 0 and -1. Using this notation, the number of non-zero bits is reduced in the average to a third of the original number. The second step is to modify the coefficients such that the number of non-zero bits is reduced. This can be done until the transfer function of the filters exceeds allowable distortions. As there is no well established relation between the shape of the transfer function and the degradation in recognition accuracy, it is necessary to compare the results obtained with a certain sets of bandpass filters to the results obtained with simulated ideal filters.

3.3.1. Low-Sensitivity Digital Filter Structures

Other considerations minimizing the number of clock cycles in the realization of a digital filter structure are:

(1) minimum number of additions

(2) minimum number of memory read and write cycles

The fourth-order Butterworth bandpass filters specifying the frequency bands of interest are realized as a cascade of second order filters. This realization is less sensitive than the direct realization of the fourth order equation because the coefficient accuracy requirements increase with the order of the difference equation [54]. The cascade realization is also preferred over the parallel one as the former requires only five coefficients as can be seen in equation (3.1), whereas in the parallel form the transfer function is decomposed as:

$$H(z) = \frac{A + Bz^{-1}}{1 + a_1 z^{-1} + a_2 z^{-2}} + \frac{C + Dz^{-1} + Ez^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}}$$

This form requires 9 coefficients. Although the parallel form is less sensitive to coefficient quantization as indicated in [55], the number of shift-and-add operations is higher than in the cascaded form.

Other configuration which were considered are the lattice filter [56] and the Fettweis ladder filter [57]. The lattice filter transfer function is sensitive to coefficient quantization due to the fact that all the coefficients have a limited range (between +1 and -1). It requires 5 addition for each second-order section and also has a high number of memory read/write cycles. The Fettweis ladder filter has not been chosen as it requires 8 additions for each second order section.

3.3.2. Low Frequency Filters

The highest coefficient sensitivity appears for filters having their poles close to the point $z = 1$ in the z plane and close to the unit circle. This occurs when the sampling frequency is much higher than the passband frequency (low-frequency filters) and for high Q filters respectively. This high sensitivity is due to the low density of allowable positions for the poles in these regions of the z plane . The position of allowable pole position with positive imaginary part inside the unit circle of the polynomial equation $z^2 + a_1 z + a_0 = 0$ for a five bit quantization is shown in Fig. 3.5

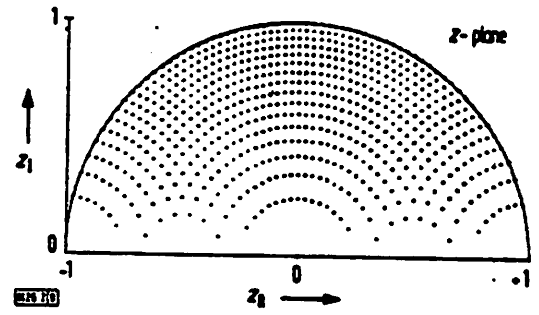


Fig. 3.5 Possible root positions inside the unit circle for $z^2 + a_1 z + a_0 = 0$.

The number of allowable pole positions is proportional to the length of the coefficient word. For a second order equation, the poles are the roots of

$$z^2 + a_1 z + a_0 = 0$$

and the distance from the pole position to the point $z = 1$ is:

$$R = \sqrt{(z_R - 1)^2 + z_I^2}$$

where z_R and z_I are the real and imaginary parts of the roots respectively given by:

$$z_R = \frac{-a_1}{2}$$

$$z_I = \sqrt{a_0 - \frac{1}{4}a_1^2}$$

From which we obtain:

$$R = \sqrt{1 + a_1 + a_0}$$

The quantized coefficients a_1 and a_0 are restricted to discrete values, i.e.

$$a_i = l_i \, . \, Q$$

where l_i is an integer, Q is the quantization step defined as

$$Q = 2^p$$

and p is the least significant bit of the coefficient word. Then the minimal distance R will be proportional to $\sqrt{Q}$ or $2^{\frac{p}{2}}$.

Thus after truncation or rounding of the (infinite precision) coefficients, each set of pole has to move to a nearby legal position to become a pair of pole of a new (distorted) transfer function. The possible position of the poles on the z-plane are evenly spaced over the real axis but not over the imaginary one because the real part of each pole is a_1, while the imaginary part is $\pm\sqrt{a_2 - a_1^2}$.

The possible locations of poles close to $z = 1$ is particularly sparse. Therefore for the low frequency filters, where the sampling frequency is much higher than the frequencies of interest, and for high Q filters, which have poles close to the unit circle, the distortion in the transfer function due to coefficient quantization is severe.

Second-order sections have been developed which have very low sensitivities to coefficient quantization. In [58], the distribution of the possible roots near the point z=1 is maximized by making the distance between the point z=1 and the nearest pole location to be proportional to the smallest possible step. By making the distance R proportional to Q we effectively double the wordlength.

To do this, we set the radicand of R as a product of two coefficients:

$$R = \sqrt{e_1 \,.\, e_0} = \sqrt{l_1 \,.\, Q \,.\, l_0 \,.\, Q}$$

This value must be small, that is:

$$|e_1| \,,\, |e_0| \ll 1$$

By chosing $e_1 = a_1 + 2$ we are satisfying the previous condition as near the point $z = 1$ the coefficient a_1 is close to -2.

By equating the values of R

$$\sqrt{1 + a_1 + a_0} = \sqrt{e_1 \,.\, e_0}$$

we obtain

$$a_0 = 1 - e_1 + e_1 e_0$$

The polynomial has now the form

$$z^2 + (e_1 - 2)z + 1 - e_1 + e_1 e_0$$

The resulting structure and the distribution of the possible root locations are shown in Fig. 3.6a and Fig. 3.6b respectively.

Another approach to increase the density of possible root locations near the unit circle is to transform the origin of the z- plane to the point $z = 1$ through a linear change of coordinates as described in [59]. The resulting structure has the same low-sensitivity as obtained in [58]. This particular filter has also been developed from the continuous state-variable filter [60]. Other low-sensitivity structures for high-Q filters are also described in [61]. An analysis of both the coefficient sensitivity and roundoff noise shows the superiority of these structures.

For realizing a bank of bandpass filters we need to obtain a transfer function which is close, within a pre-defined error range, to the required one while using a minimum number of non-zero bits in the coefficient words.

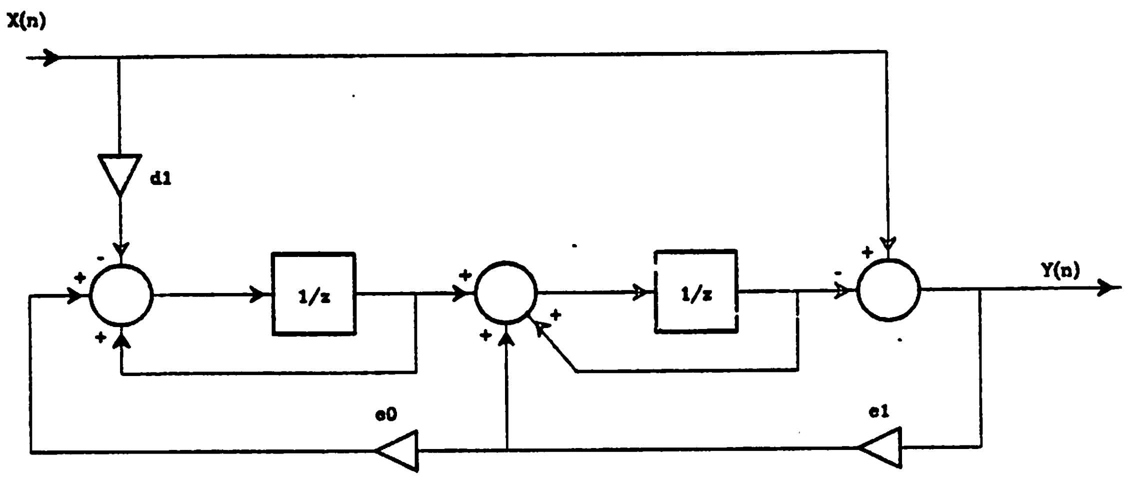


Fig. 3.6a Low sensitivity structure after [58]

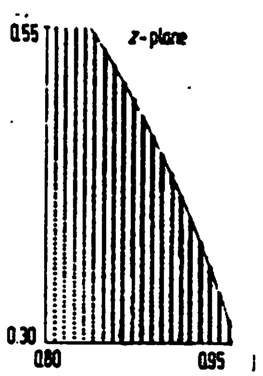


Fig. 3.6b Possible pole locations of the structure

To analyze the deviation of a parameter F of a second order section as a function of its coefficients we use :

$$\frac{\Delta F}{F} = S_a^F \frac{\Delta a}{a} + S_b^F \frac{\Delta b}{b} \qquad (9)$$

where a and b are the coefficients of the second order denominator and S_a^F and S_b^F are the sensitivity of the parameter F to a and b respectively. Low-sensitivity to coefficients is a desirable characteristic which enables us to use truncated coefficients.

We compare three structure having comparable sensitivities and requiring the same number of operations for implementation. We shall then choose that structure yielding a transfer function which is closest to the required ones and having the smallest number of non-zero bits in its coefficients. Although the structure has not the minimal sensitivity at that frequency, its realization requires the least amount of operations.

The choice of a low sensitivity structure depends on the position of the poles of the particular filter. In the realization of the bank of bandpass filters, different structures will therefore be needed according to the pole positions of the different filters. In [61], the structures used for the 16 filters proposed for the bank of filters is described. A sensitivity analysis results in optimization in the choice of structures.

3.3.3. Sensitivity Comparison

Three different structures were investigated to implement the low frequency bandpass filters. The first is the structure proposed by Agarwal and Burrus [59] as shown in Fig 3.7a. This will henceforth be designed as the AB structure . The second structure is the state variable structure [61] and is

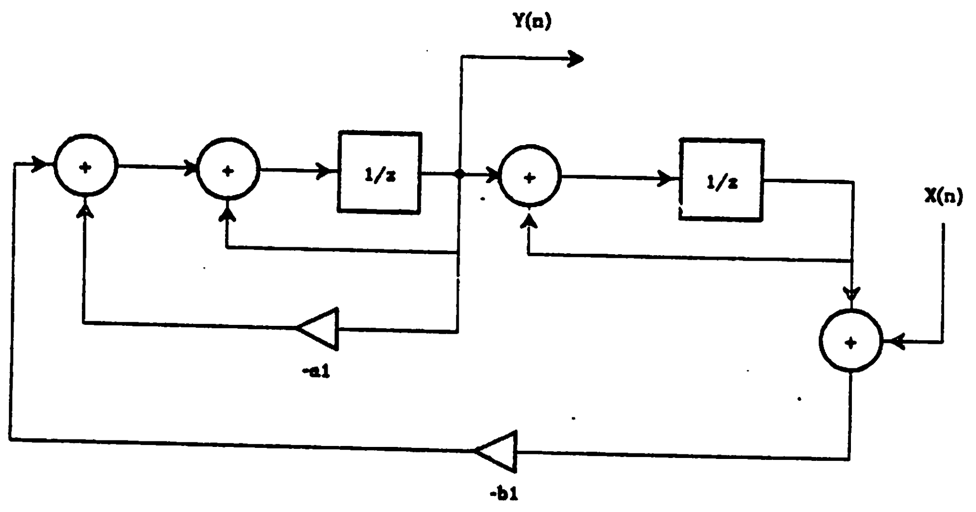


Fig. 3.7a Second order AB structure

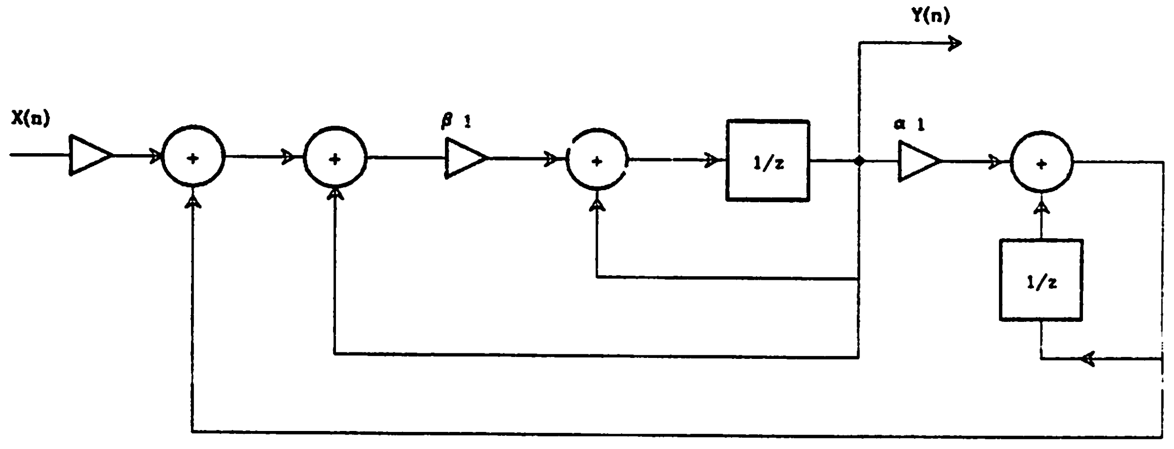


Fig. 3.7b Second order SV structure

shown in Fig. 3.7b. This will be called the SV second order structure. In this structure the second delay does not serve as an integrator. This yields a higher sensitivity of the center frequency with respect to the two coefficients in the denominator than obtained with the previous structure. The third structure is a modification of the state variable one. This is shown in Fig. 3.7c This structure will be designated as MSV.

The transfer function of the three structures as described in Fig. 3.7 are:

for the AB structure:

$$H(z) = \frac{-b_1 G z^{-1}(1 - z^{-1})}{1 + (a_1 - 2)z^{-1} + (1 + b_1 - a_1)z^{-2}}$$

for the SV structure:

$$H(z) = \frac{-\beta_1 G z^{-1}(1 - z^{-1})}{1 + (\beta_1 + \alpha_1\beta_1 - 2)z^{-1} + (1 - \beta_1)z^{-2}}$$

and for the MSV structure:

$$H(z) = \frac{-\alpha_2 G z^{-1}(1 - z^{-1})}{1 + (\alpha_2 - 2)z^{-1} + (1 + \alpha_2\beta_2 - \beta_2)z^{-2}}$$

where the zero at $z = 1$ has been ignored due to its small influence at low frequencies.

We compare the three structures by computing the sensitivity of the center frequency of the second order section with respect to the coefficients in the denominator.

The denominators of the second-order sections AB, SV and MSV are given respectively by:

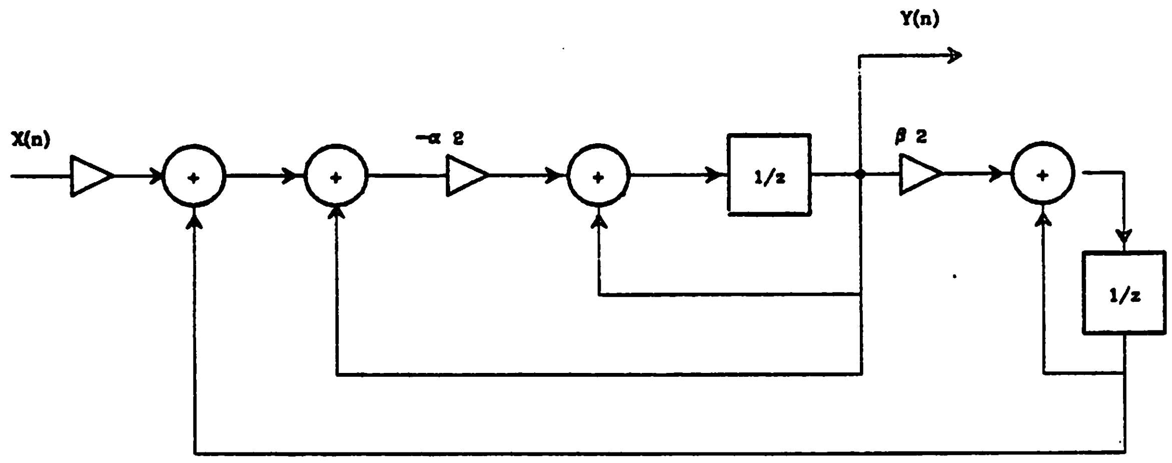


Fig. 3.7c Second order MSV structure

$$z^2 + az + b = z^2 + (-2 + a_1)z + (1 + b_1 - a_1) \tag{3.10a}$$

$$= z^2 + (-2 + \beta_1 + \alpha_1\beta_1)z + (1 - \beta_1) \tag{3.10b}$$

$$= z^2 + (-2 + \alpha_2)z + (1 + \beta_2\alpha_2 - \alpha_2) \tag{3.10c}$$

A second order section is determined by the position of its two complex poles. The coordinates of the poles in the z plane are given in terms of the angle ϑ and the radial distance R. It is easier to do the sensitivity analysis in terms of ϑ and δ. ϑ, the angle of the pole, is found from: $a = -2R\cos\vartheta$, while δ, the radial distance of the z-plane pole from the unit circle is given by: $\delta = 1 - R$ The coefficient b is related to δ by the following relation:

$b = R^2 = (1 - \delta)^2$.

The center frequency ϑ_0 is given by (see appendix):

$$\cos\vartheta_0 = \frac{1 + R^2}{2R}\cos\vartheta$$

Using the relation between R and δ in this equation gives us:

$$\cos\vartheta_0 = \left[1 + \frac{\delta^2}{2(1-\delta)}\right]\cos\vartheta$$

As we are dealing with poles very close to the unit circle (small δ) the term in δ^2 is neglected and to a good approximation ϑ_0 is equal to ϑ.

The sensitivity $S_{a_1}^{\vartheta}$ is defined as:

$$S_{a_1}^{\vartheta} = \frac{\partial\vartheta}{\partial a_1}\frac{a_1}{\vartheta}$$

For the AB structure ϑ can be expressed as:

$$\vartheta = \cos^{-1}\frac{-a}{2\sqrt{b}} = \cos^{-1}\frac{2 - a_1}{2\sqrt{1 + b_1 - a_1}}$$

Then, $\frac{\partial\vartheta}{\partial a_1}$ is found to be:

$$\frac{\partial\vartheta}{\partial a_1} = -\frac{1}{\sin\vartheta}\cdot\left[\frac{a_1 - 2b_1}{4(1 + b_1 - a_1)\sqrt{1 + b_1 - a_1}}\right]$$

The sensitivity $S_{a_1}^{\vartheta}$ is found by multiplying the previous expression by $\frac{a_1}{\vartheta}$. In terms of the coefficients a and b, the sensitivity is:

$$S_{a_1}^{\vartheta} = \frac{(a + 2b)(a + 2)}{2\vartheta b\sqrt{4b - a^2}}$$

where $\sin\vartheta$ is expressed as:

$$\sin\vartheta = \frac{\sqrt{4b - a^2}}{2\sqrt{b}}$$

Using the expressions relating a and b to ϑ and δ and the relation

$\cos\vartheta \approx 1 - \frac{\vartheta^2}{2}$,

a and b can be expressed as:

$$a = (\vartheta^2 - 2)(1 - \delta)$$

$$b = (1 - \delta)^2$$

So that we obtain:

$$S_{a_1}^{\vartheta} = \frac{\vartheta^2}{2(1-\delta)\sqrt{4-\vartheta^2}} \tag{3.10a}$$

The sensitivity to the coefficient b_1 is obtained in a similar way and is given by:

$$S_{b_1}^{\vartheta} = \frac{2-\vartheta^2}{2(1-\delta)\sqrt{4-\vartheta^2}} \tag{3.10b}$$

For the SV circuit the sensitivities are:

$$S_{\alpha_1}^{\vartheta} = \frac{1}{\sqrt{4-\vartheta^2}} \tag{3.11a}$$

$$S_{\beta_1}^{\vartheta} = \frac{1}{(1-\delta)\sqrt{4-\vartheta^2}} \tag{3.11b}$$

While for the MSV circuit the sensitivities are:

$$S_{\alpha_2}^{\vartheta} = \frac{1}{(1-\delta)\sqrt{4-\vartheta^2}} \tag{3.12a}$$

$$S_{\beta_3}^{\vartheta} = \frac{(\vartheta + \frac{1}{Q})(2-\vartheta^2)}{\vartheta\, 2(1-\delta)\sqrt{4-\vartheta^2}} \tag{3.12b}$$

The following can be deduced from equations (10) to (12):

(1) at low frequencies, i.e. for small values of ϑ, the sensitivity $S_{a_1}^{\vartheta}$ is close to zero so that the coefficient a_1 can be approximated very coarsely while having very little influence on the position of the poles.

(2) at low frequencies the sensitivities S_{b_1}, $S_{a_1}^{\vartheta}$, $S_{\beta_1}^{\vartheta}$ and $S_{a_2}^{\vartheta}$ approaches 1/2 as ϑ tends to zero.

(3) as ϑ increases the sensitivity $S_{b_1}^{\vartheta}$ decreases.

(4) for small and medium selectivity filters, i.e. small or medium Q's, the sensitivity $S_{\beta_2}^{\vartheta}$ is high. For high selectivity filters this sensitivity is equal to $S_{b_1}^{\vartheta}$ and tends to 1/2 for small ϑ .

(5) comparing particularly the AB and SV circuits, we observe that the sum of the sensitivities in the AB circuits is $\frac{1}{(1-\delta)\sqrt{4-\vartheta^2}}$. This is about half the sum of the sensitivities in the SV circuits.

(6) as ϑ increases the sensitivities of the AB and SV configurations increases. Therefore beginning from a certain frequency we need to use another configuration for the realization of the filters.

The AB circuit has lower center frequency sensitivities than the two other circuits. However as mentioned earlier the deviation of the center frequency of the filter is given by (9) where ϑ now replaces F. The absolute deviation of the desired coefficient from its closest possible approximation is therefore also important.

The three different circuits give rise to different coefficients. The choice which structure to use for the different filters depends on the sum of the products of the deviation and the corresponding sensitivity for the coefficients of that particular filter.

In the realization of a bank of bandpass filters the center frequencies of the different filters must remain constant so that no gap or overlap will be created due to non-ideal coefficients. The variations in the bandwidth of the filters due to

Filter	section	circuit	actual coefficient		approximated coefficient	
0	higher	MSV	α_2=0.076939	β_2=0.1247065	0.078125 ($2^{-4}+2^{-6}$)	0.125 (2^{-3})
	lower	SV	α_1=0.064948	β_1=0.0313311	0.0644531 ($2^{-4}+2^{-9}$)	0.03125 (2^{-5})
1	higher	AB	a_1=0.100354	b_1=0.022989	0.09375 ($2^{-3}+2^{-5}$)	0.0234375 ($2^{-5}-2^{-7}$)
	lower	AB	a_1=0.053098	b_1=0.0077411	0.0546875 ($2^{-4}-2^{-7}$)	0.0078125 (2^{-7})
2	higher	AB	a_1=0.128054	b_1=0.0577532	0.125 (2^{-3})	0.05859375 ($2^{-4}-2^{-8}$)
	lower	AB	a_1=0.084442	b_1=0.031831	0.078125 ($2^{-4}+2^{-6}$)	0.03125 (2^{-5})
3	higher	AB	a_1=0.175312	b_1=0.1077579	0.1875 ($2^{-3}+2^{-4}$)	0.109375 ($2^{-3}-2^{-6}$)
	lower	MSV	α_2=0.126723	β_2=0.5628157	0.125 (2^{-3})	0.5625 ($2^{-1}+2^{-4}$)
4	higher	AB	a_1=0.238580	b_1=0.1725217	0.25 (2^{-2})	0.171875 ($2^{-3}+2^{-4}-2^{-6}$)
	lower	AB	a_1=0.182767	b_1=0.1258528	0.1875 ($2^{-3}+2^{-4}$)	0.125 (2^{-3})
5	higher	AB	a_1=0.311452	b_1=0.2489966	0.3125 ($2^{-2}+2^{-4}$)	0.25 (2^{-2})
	lower	AB	a_1=0.250569	b_1=0.1948139	0.25 (2^{-2})	0.1875 ($2^{-3}+2^{-4}$)
6	higher	SV	α_1=4.727919	β_1=0.0742053	4.5 ($2^{2}+2^{-1}$)	0.078125 ($2^{-4}+2^{-6}$)
	lower	SV	α_1=4.133265	β_1=0.0665588	4 (2^{2})	0.06640625 ($2^{-4}+2^{-8}$)

Table 3.2 Structures and coefficients for the first seven filters

variations in the "Q" are in our case of much less significance. A change of a few dB in either direction in the -3 dB overlap points between adjacent filters does not show an influence in the recognition accuracy.

Table 3.2 indicates the structures used in the realization of the lower and higher sections of the seven lower bandpass filters and also the correct and approximate coefficients of the filters.

The table also indicates the number of non-zero bits used for the approximated coefficient. As can be seen only one coefficient uses three non-zero bits. All the others use one or two non-zero bits. The transfer function of these seven filters appears in Appendix A.

The other nine bandpass filters are realized using the direct form. Their coefficients are described in [61].

3.3.4. Filter Structure as a Function of Frequency

As the sensitivity increases with the frequency ϑ, other structure having lower sensitivity to the coefficients must be used for the higher frequency bandpass filters. The structure called realization 1B in [59], and shown in Fig. 3.8 has the following transfer function:

$$H(z) = \frac{z^2}{z^2 + (-2 + a_2)z + (1 - b)}$$

For this structure the sensitivities of the center frequency are:

$$S^{\vartheta}_{a_2} = \frac{2b(a + 2)}{2\vartheta b\sqrt{4b - a^2}}$$

$$S^{\vartheta}_{b_2} = \frac{a(1 - b)}{2\vartheta b\sqrt{4b - a^2}}$$

Expressing these sensitivities in term of ϑ and δ, results in the following expression for the numerator of the previous sensitivity functions respectively:

$$2b(a + 2) \approx 2\vartheta^2(1 - \delta)^3 + 4\delta(1 - \delta)^2$$

and

$$a(1 - b) \approx 2\delta(\vartheta^2 - 2) + \delta^2(\vartheta^2 + 2)(\delta - 3)$$

All the sensitivities have the same denominator: $2b\vartheta\sqrt{4b - a^2}$.

Expressed in terms of ϑ and δ, it is:

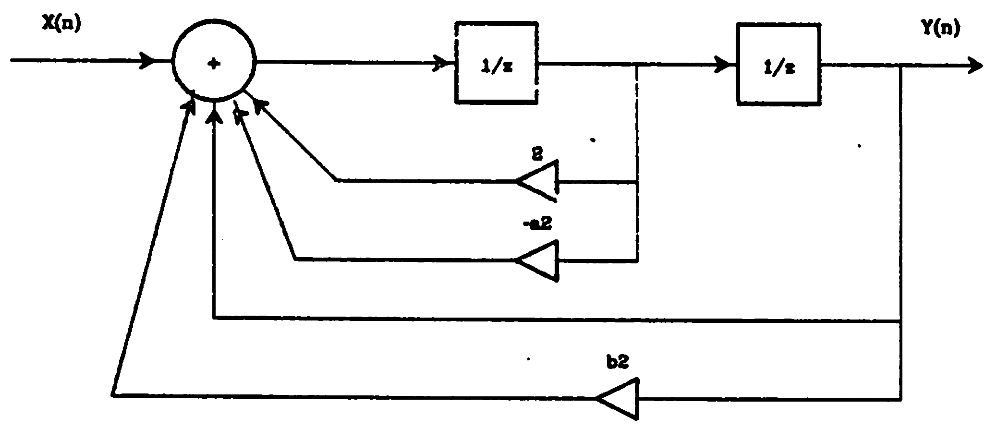


Fig. 3.8 Realization 1B in [59]

$$2b\vartheta\sqrt{4b - a^2} = 2(1-\delta)^3\vartheta^2\sqrt{4+\vartheta^2}$$

The denominators of the sensitivity functions to a first approximation for the three different structures: Agarwal-Burrus (A-B), realization 1-B and direct realization are as follows:

A-B: $S_{a_1}^{\vartheta} \approx \vartheta^4$

$S_{b_1}^{\vartheta} \approx \vartheta^2 (2 - \vartheta^2)$

1-B: $S_{a_2}^{\vartheta} \approx 2\vartheta^2$

$S_{b_2}^{\vartheta} \approx 2\delta\vartheta^2$

direct: $S_a^{\vartheta} \approx \vartheta^2 - 4$

$S_b^{\vartheta} \approx \vartheta^2 - 2$

Comparing these expressions with the one for the previous filter shows that the numerator is proportional to $2\vartheta^2$ and $2\delta\vartheta^2$ instead of ϑ^4 and $\vartheta^2(2-\vartheta^{2)}$ respectively. For increasing frequencies, the direct realization shows less sensitivity as the denominator of the sensitivity functions are proportional to $\vartheta^2 - 4$ and $\vartheta^2 - 2$ respectively.

The frequency at which another structure is used depends on the coefficients. At a certain point, different structures have to be tried to find the coefficients resulting in minimal non-zero bits.

3.4. Modification of the Coefficients

Further reduction in the number of operations is done by reducing the number of non-zero bits in the coefficients of the filters. This modification of the coefficients results in distortion of the transfer function. One approach to this reduction is described in [62]. The algorithm used there is the following: The CSD (Canonical signed digit) representation of each coefficient of the filter is restricted to a specified number of non-zero CSD digits. For selected fraction of the Nyquist frequency, $0 \leq W_1 < W_2 .. < W_m \leq 1$, the filter is optimized by minimizing the sum of the squared differences between the desired transfer function at the different frequency bands, $I(W_i)$, and the actual transfer function of the filter $H(e^{jW_i\pi})$ so that the minimization is done on the expression:

$$\sum_{i=0}^{m} \left| I(W_i) - H(e^{jW_i\pi}) \right|^2$$

To find the solution to this optimization problem the optimal filter with infinite wordlength is first designed and then a random search in the N-dimensional parameter space is performed in a neighborhood surrounding the point corresponding to the coefficients of the optimal infinite wordlength filter. In this second phase, the random search method is used to find the trial values for the coefficients. This random search method has constraints on both the the coefficient wordlength and the maximum number of CSD bits set in each coefficient. This search does not guarantee that an optimum will be found although it probably will come very close.

The approach as described in the appendix is different. Due to the use of a barrel shifter in the hardware, there is no more need to consider the wordlength. Rather, the number of non-zero bits is important. So that a coefficient represented by a 3-bit word with the first and last bit being non-zero is less advantageous from the point of view of number of computation than a 5-bit word where only the last one is non-zero.

The method was developed for maximally flat bandpass filters. In our application we are dealing with four pole filters built as a cascade of two second-order sections. We are dealing only with the polynomial coefficients of the denominator of the individual transfer function. In this case we have four coefficients which have to be found so as to reduce the number of shift-and-add operations their non-zero bits represents and still provide a satisfactory transfer function. The method to judge this transfer function consists on using the program DINAP interactively [63]. The detailed algorithm and an example are described in the appendix .

The main idea consists in reducing the number of the non-zero bits in two of the coefficients and to modify the two remaining one so that the -3 db points of the transfer function remain the same. First the coefficient closest to the smallest combination of powers of two in each section is chosen. The deviation from the infinite wordlength coefficient are found by simple subtraction. Using the formulas which were developed for keeping the center frequency and bandwidth of the desired filter unchanged, we can find the deviations in the two other coefficients. Using the new coefficients thus obtained we use DINAP to simulate the filter. The filter thus obtained is not the desired one as :

(1) the formulas are only correct for the first approximation

(2) only the changes in the denominator are taken into account.

The deviations in the center frequency and bandwidth from the desired filter are introduced in the formulas and new coefficients are obtained. Using DINAP, the process of simulating the filter, compensating the formulas by the amount of deviations, is repeated by compensating alternatively the center frequency and the bandwidth. Using these compensated equations, we find the last two coefficients. The discrete coefficients are then approximated to the ideal coefficients. This approximation depends on the amount of deviation that can be tolerated in the bandwidth and center frequency.

The disadvantages of this method are:

(1) This method is interactive. Therefore it is difficult to make it automatic.

(2) The method is not general. The formulas are derived for second order sections. In higher order filters the expressions for the center frequency and the bandwidth are increasingly complex and therefore difficult to use in this form.

(3) No provision has been made to give a weight to different portions of the transfer function.

(4) Although good results are obtained for some examples, there is no proof that this method yields optimal coefficients.

The advantage of the method is that we are not bound to a certain wordlength. The coefficients of the filters have in general different length.

The results obtained for the example described in the appendix show that it is possible to reduce the number of non-zero bits in the four coefficients from 10 in the case of almost ideal transfer function to 6 with a tolerable degradation. A lesser degradation is possible with an increase to 7 or 8 non-zero bits. This is illustrated in Fig. B1 and Fig. B2 in appendix B.

3.5. Special Purpose Circuits for Spectral Analyzer

The DSP cannot take advantage of the reduction in the number of non-zero bits as these low-sensitivity structures still require more operations in the realization than the direct form. The reason is illustrated in the following comparison between the computation required for a bandpass filter in a state variable configuration and in direct form. The SV configuration appears in Fig. 3.7b .

Denoting x(n) as the input variable, y(n) as the output variable and P(n) as a state variable, the implementation of a SV BPF is the following:

$$y(n) = -\beta 1\{ Gx(n-1) + P(n-1) + y(n-1) \} + y(n-1)$$
$$P(n) = a1y(n) + P(n-1)$$

The direct form has a transfer function given by:

$$H(z) = \frac{1}{z^2 + A1z + (1-b)}$$

The implementation is straightforward:

$y(n) = (b-1)y(n-2) - A1y(n-1) + x(n)$

Therefore a larger number of additions and read and write cycles must be done to implement a low-sensitivity structure. This means higher speed needed in the processor for real time implementation and also more ROM memory for the microcode. The DSP is an efficient structure as the adder is used almost every clock cycle. Still, the large number of operations in the computation of the 16 channels requires two processor operating in parallel.

To reap the advantages of the low-sensitivity structures a third approach for the implementation of the spectrum analyzer is proposed: using special purpose circuits.

To reduce the computations due to multiplication of data by a coefficient, a common operation in signal processing, parallel multiplier circuits have been developed. Their disadvantage is the large area required for their implementation. Commercially available general purpose signal processors like the 7720 of NEC and the TMS320 from TI use such parallel multiplier. These multiplier have little advantage over the parallel-serial (PS) multiplier as the number of nonzero bits in the coefficients of the filters, and hence the number of clock cycles, is less than two in the average. Therefore, the special purpose circuits use the PS multiplier approach.

The spectrum analyzer requires the computation of 32 second order sections, for the fourth order bandpass filters, and 16 full-wave rectification. The computation of the third-order lowpass filter is done in two steps: The first order section is computed at the 14 kHz rate. The second order section is computed at a rate eight times slower [61].

Two observation are made on the present digital spectrum analyzer chip :

(1) the area occupied by the data processing circuits is much smaller than the area occupied by the memory. Thus increasing data circuits increases only slightly the size of the whole analyzer

(2) the absolute value operation and the lowpass filter are the same for all 16 channels. Therefore special purpose circuits are indicated for their realization.

To increase the throughput of the circuit the following architecture is proposed:

(1) using parallel circuits for the additions and for the multiplications in the computation of the second order section. Refering to the previous equation for the implementation of the state variable structure, the additions are done in parallel using additional registers and adders. To reduce the number of cycles for the computation of the multiplications, an additional barrel shifter and adder can compute part of the shift-and-add operation. The microcode is made wider but with a reduced depth. Additional control lines are required to control both barrel shifters. This section computes alternatively y(n) and P(n).

(2) implementing the full wave rectifier using a dedicated circuit. Basically this circuit is an "exclusive or". The data appears in two's complement. Therefore when the MSB is 0, the data is passed as is. When the MSB is 1, all the bits are inverted. This introduces a small error as the correct two's complement operation requires adding a 1 at the LSB. This error has negligible effect on the result of the operation.

(3) implementing the first order section of the lowpass filter using a dedicated circuit with a simplified barrel shifter. The output $y(n)$ of this section is given by:

$$y(n) = py(n) + x(n)$$

where p is the constant representing the value of the pole of the section, and $x(n)$ is its input sample. This circuit requires an adder, a delay register and a constant multiplier. As this constant is known, the barrel shifter is simplified to shifting the data. The second order section of the lowpass filter requires much less computations and therefore will not be discussed here.

(4) increasing the depth of the barrel shifter. Presently the barrel shifter has a depth of 6. This means that for non-zero bits separated by more than 5 0's, another shift is needed for the shift-and-add operation. By increasing the depth of the barrel shifter to 10, no second cycle is needed. A difficulty in increasing the depth of the barrel shifter is that the loading on the lines increases as n^2. However a depth of 10 is easily feasible.

These changes basically use the same amount of hardware but in a different configuration. The analyzer is then a cascade of optimized circuit for realizing a second order section, a full wave rectification and a lowpass filter with adequate shift registers between them.

CHAPTER 4

Integrated Dynamic Time-Warp Circuit

4.1. Introduction

The digital features representing the spectrum are stored in the memory of the system and used as references against which the incoming speech - in the same representation - is compared. The information which is relevant for our purposes is the frequency transfer function of the vocal tract. This curve computed every 10 msec is characterized at 12 different frequencies. The problem of speech recognition then is now reduced to compare an incoming string of digital words to the ones stored in memory or, in short, reduced to a pattern matching problem.

In matching speech patterns the incoming pattern differs every time even for the same utterance. This variation in spectral information is a result of the particular state of the speaker such as: tiredness, emotional state etc... . Therefore we have to expend and compress non-linearly the incoming patterns until we obtain an optimal time alignment of the incoming test pattern with the one stored in memory.

The comparison of speech patterns uses the dynamic time-warp (DTW) algorithm to align the incoming pattern with the reference along a non-linear time axis. The operation of the algorithm was explained in chapter 2.

The choice of a computationally efficient DTW algorithm performing this alignment, is described in the next section. Section 3 describes the operation and architecture of the circuit. The chip layout and specific circuits are described in section 4. Section 5 presents the performances obtained with this circuit.

4.2. DTW Algorithm Considerations

4.2.1. Algorithm Selected

The advantage of the DTW algorithm is the small amount of operations it requires in the computation of the optimal path. However when implemented on a general purpose computer the DTW algorithm can be the most computational intensive part of the recognition process. By implementing a relatively simple circuit dedicated to the algorithm we can implement all the computation of the algorithm in real time.

We choose an algorithm which does not compromise recognition accuracy and needs as little silicon area as possible when implemented for real time recognition.

All the DTW algorithms used have the following points in common :

[1] the "distance" or degree of dissimilarity between two patterns is the sum of the distances of some measure of the spectral energy between frames of the test pattern and the reference patterns after optimal time normalization.

There are many proposals for this local distance based on the method used to extract the speech features. A local distance measure giving good results in terms of accuracy of recognition when using bandpass filters in the spectral analyzer, is the Euclidean distance. This distance is computed as the sum of the squared differences between the energy of each channel of the incoming frame and the energy of the corresponding channel in the frame of the reference template.

[2] the accumulated distance at any point of the two-dimensional grid spanned by the reference and test pattern, is obtained by summing the local distance at that point with the minimum of the accumulated score coming from the locations from which this path can come.

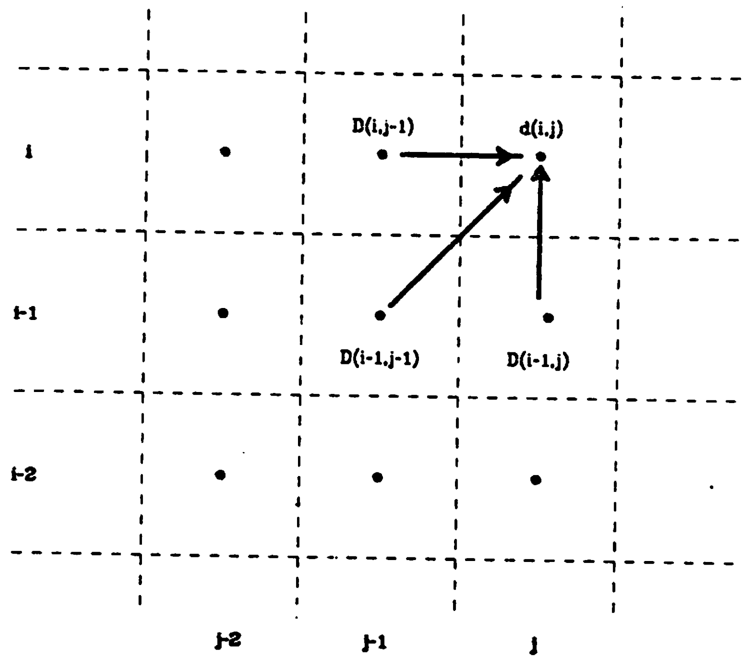
The algorithms differ between them on the weighting function , the position and number of states from which the path can originate and on the normalization factor.

To reduce the amount of computation, Sakoe and Chiba restricted the warping of the time axis [64] by imposing constraints on the permissible slope of the paths . A slope constraint of 2 to 1 for example allows only two frames of one word to be warped with one frame of the other word. Experiments [38] showed that these restrictions decrease recognition accuracy . As described in paragraph 2.5.2 we have chosen the basic algorithm without slope constraints as shown in Fig. 4.1 .
The equation for this algorithm is:

$$D(i,j) = d(i,j) + \min\left\{D(i-1,j), D(i-1,j-1), D(i,j-1)\right\} \tag{4.1}$$

where index j refers to the column and index i to the row. d(i,j) is the local distance between frame j of the incoming test word and frame i of the reference word, D(i,j-1) , D(i-1,j-1) and D(i-1,j) are the accumulated scores adjacent to frame (i,j).

This algorithm is simple to implement in IC technique as the memory requirements are small. Only the accumulated scores of the previous column - the D(i,j-1) - are needed and one additional memory cell for D(i-1,j).



D(i,j) = d(i,j) + min { D(i,j-1), D(i-1,j-1), D(i-1,j) }

Fig. 4.1 Implemented DTW algorithm

The distance we are using for d(i,j) is the Euclidean distance given by eq. (2.8).

$$d(i,j) = \sum_{n=1}^{16} [\, l_A^{(n)} - l_B^{(n)} \,]^2 \tag{4.2}$$

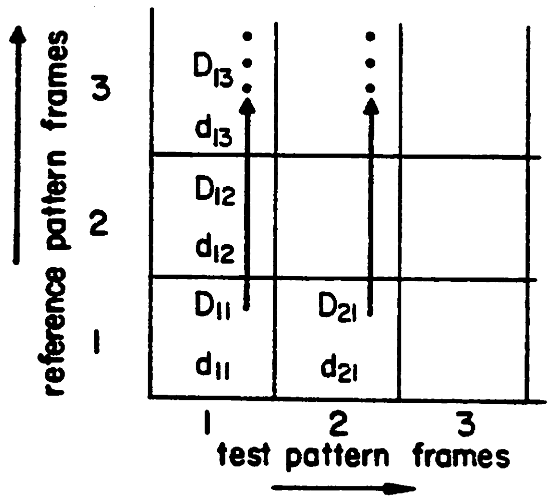
The algorithm chosen has a bias towards paths on the diagonal. This happens because the sum of the two local distances along a vertical (horizontal) and horizontal (vertical) paths - two frames - is always greater than the distance along the diagonal - one frame . The sum is equal only in the case that one segment has a local distance of zero. The path along the diagonal has less weight then a path along a vertical (horizontal) frame and horizontal (vertical) frame and therefore is the one which will be chosen. Favorizing diagonal paths introduces an error in the pattern matching procedure. This error was found to be negligible. When our algorithm was compared with others as done in [38], it performed as well or better. There was no need therefore to correct this effect .

The time normalization needed for correct comparison of the results of the algorithm and their adaptation to continuous speech recognition is explained in chapter 2.

4.2.2. Order of Computation

To obtain the optimal path, it is necessary to compute the accumulated scores at all locations (i,j) , $i \leq N$, $j \leq M$, of the two-dimensional grid spanned by the frames of the reference template and those of the incoming word, where N,M are the numbers of frames in the reference template and incoming word respectively. To compute the accumulated score at location (i,j) we need the three adjacent accumulated scores. The computation of the path must be repeated for all templates in the reference memory.

Different strategies are used to compute the optimal path in a grid for all the templates as shown in Fig. 4.2 In Fig. 4.2a, the accumulated scores are computed along the columns of the template for every frame of the incoming word. This computation is continued up to the last frame of the incoming word. This sequence is repeated for each template in the memory. In this method all the frames of the incoming word must be acquired before it is possible to begin the computation of the optimal paths for the templates except the first one. The computation of each path is done serially and the frames of the incoming word must be stored as they are used in the computation of the optimal path for every template in memory. The recognition is therefore not in real time. This



a) Computation of accumulated scores along columns for each template separately

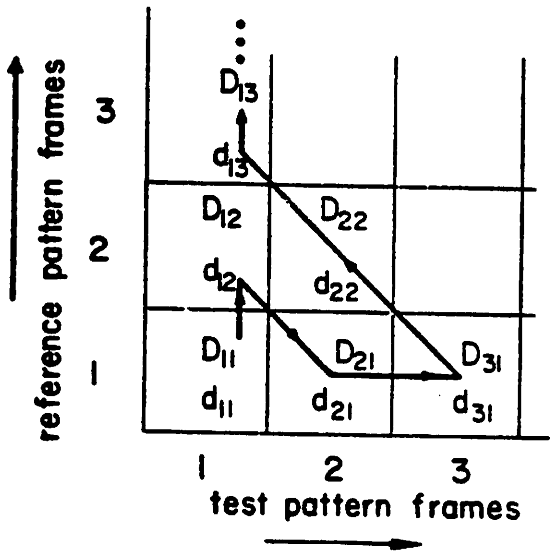


Fig. 4.2b Computation of accumulated scores along diagonals

technique is used in the speech recognition systems described in [24]and [25].

In 4.2b the accumulated scores are computed along a diagonal. As in the previous method, all the frames of the incoming word must be acquired before it is possible to begin the computation of the optimal paths. Here too, the computation of each path is done serially, and the frames of the incoming word must be stored. [65] describes a system using this strategy.

The method used in our system is shown in Fig. 4.2c. We compute the accumulated scores - the D(i,j) - column after column, however unlike the method described in 4.2a, the accumulated scores are computed for all the frames of all the templates in the memory. The computation of all the optimal paths begins as soon as the first frame of the incoming word is obtained. All the paths are computed in parallel and there is no need to store all the frames of the incoming word, only one frame at a time.

The advantage of this method is obvious for the recognition of large vocabularies. We can obtain the accumulated scores for all the templates in the reference memory by stacking them in one column and by repeating the sequence of computations along the column. These computations are done before the next frame is available. Thus the top scores for all the paths are available after the last incoming frame. This provide our system with real time recognition capability.

A code word marks the end of each template and the beginning of the next one. An additional code word indicates the last frame in the TP memory. Thus all the paths are computed in parallel by time multiplexing the hardware.

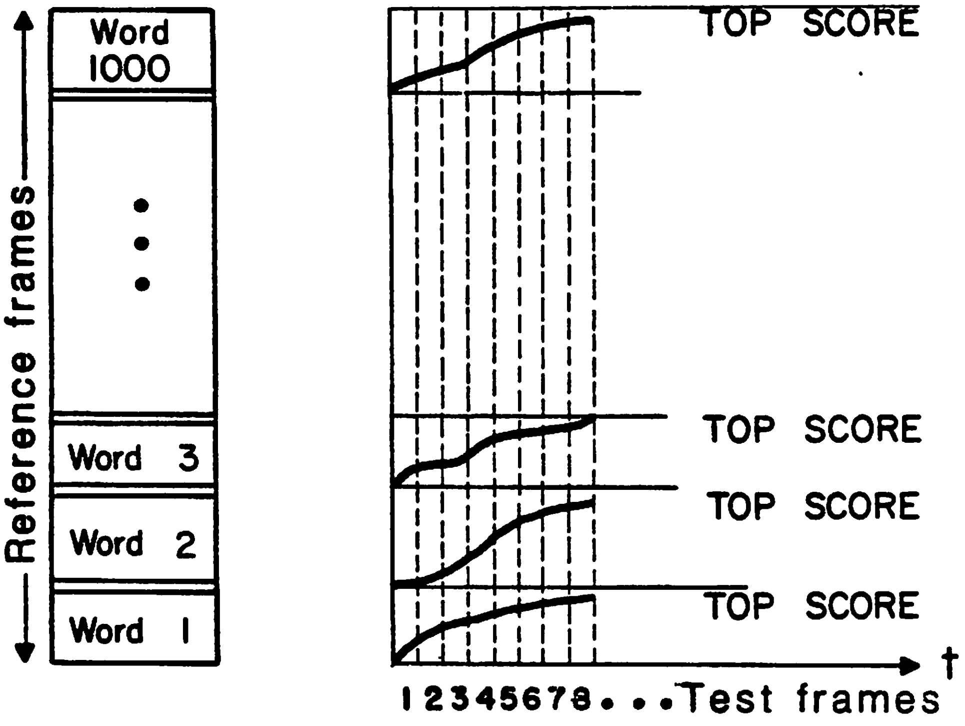


Fig. 4.2c Computation of accumulated scores for all templates in parallel

4.2.3. Data Word Length Requirements

The determination of the number of bits needed to represent the local distance d(i,j) and the accumulated distance D(i,j) is based on the coded features as obtained in the front-end chip. Each frame is represented by 12 features. Each feature is 4 bits wide. All features represent a positive number. The local distance is the sum of 12 squared differences between 4 bits wide words. In the worst case the value of the local distance is 12 times 256. Good matching between frames is indicated by local distance scores in the order of 20 to 40. In connected speech mode the scores are higher, up to approximately 80, as the matching is done between strings of words. As there is no need to represent precisely the values of local distances which are much larger than the values expected for good matching, it was found practical to limit the value of the local distance to 127, thus using only 7 bits .

Using 127 as a maximum value and assuming an utterance of 100 frames - about 2.5 sec. - the maximum accumulated score would then be 12,700. However the same argument for choosing the small number of bits in the local distance is also true here. There is no advantage of representing the distances which represent an extremely poor match between words. A good match even for such a long utterance would be of the order of 200 to 1000. Furthermore the final decision making software of the system includes a threshold level above which the smallest accumulated score is recognized as "no match". Therefore supposing that the threshold would be about half the maximum or about 6350, we choose a word which is 13 bits wide.

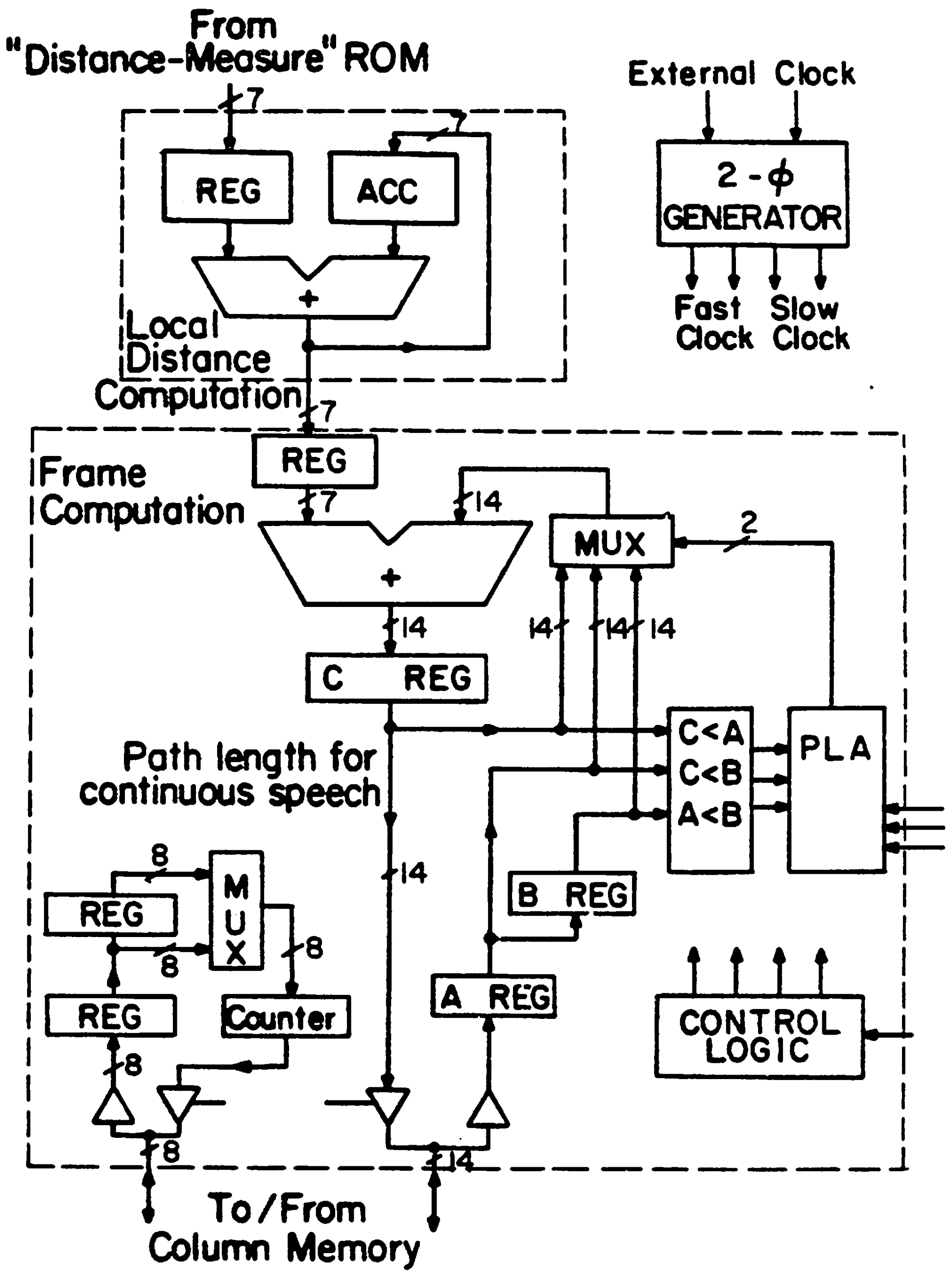


Fig. 4.3 Block diagram of the DTW chip

4.3. Hardware Architecture and Operation

4.3.1. DTW IC Architecture

Our recognition system underwent algorithm improvements which resulted almost continuously in changes in the hardware. The system, including all peripherals is described in [66]. The description presented here refers to the special purpose integrated circuit in the system realizing the DTW algorithm. The block diagram of the DTW IC is shown in Fig. 4.3 .

The computation of the squared difference between the corresponding features of the reference and incoming word is done using a look-up table implemented by a ROM. The twelve words obtained from the ROM - one for each channel - are added in the "Local Distance" block. At the end of twelve such accumulations, the output from the adder is the local distance. This local distance is then stored in the first register of the "Frame Computation" block.

The comparison between the three adjacent accumulated scores takes place during the clock cycles- also called fast clock - in which the computation of the local distance is done. As we are doing this comparison along the columns, the accumulated scores of the previous columns D(i,j-1) and D(i-1,j-1) are read from the scratch memory also called "DP memory" and stored in registers "A REG" and "B REG" respectively. The accumulated score D(i-1,j) computed in the previous cycle resides in register "C REG". The output of the three registers appear both at the input of three comparators and at the input of the multiplexer.

The magnitude comparators output the results of $C<A$ $C<B$ and $A<B$ to the PLA where A, B and C represents the magnitude of the values stored in the corresponding registers. The PLA controls the multiplexer to compute the

correct word. Because the conditions along the first column or along the first row differ from the conditions existing in the rest of the grid, additional information is needed. This information includes : whether the computation takes place in the first row , whether the computation is along the first column and whether the recognition is done for isolated or connected words. The detailed equations for the PLA appear in section 4.4.7. The correct accumulated score appearing at the output of the multiplexer is then added to the local distance and is the accumulated score for this frame. Following the algorithm, this result is used in the computation of the next cell. Therefore the result is stored in a master/slave register "C REG" and also sent to the "DP MEMORY". The address counter of this memory is then incremented and the content is written out and latched into "A REG" to repeat the same sequence of operations.

4.3.2. Slow-Loop Operations

The operations needed to move from one frame to the other along a column can be understood by referring to Fig. 4.4 .
At a certain time the three registers A,B and C contain the appropriate accumulated scores as described in Fig. 4.4a. The computation of the accumulated score for this cell yields a results denoted by D(i,j). To compute the accumulated score for the cell above it in the column the following operations must be done as shown in Fig. 4.4b:

T1: $R \rightarrow W$; the DP memory is in write mode

$C \rightarrow M$; the content of C are written in DP memory

$A \rightarrow B$; the content of A are transferred to B

T2: $W \rightarrow R$; the DP memory is in read mode

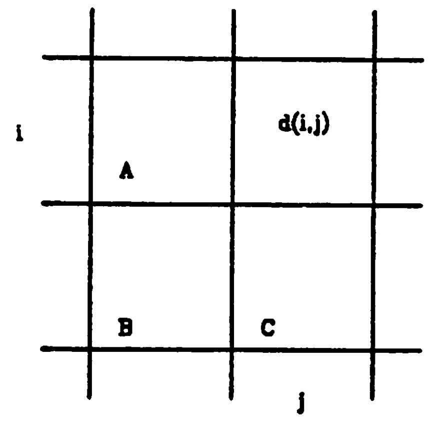


Fig. 4.4a Initial state

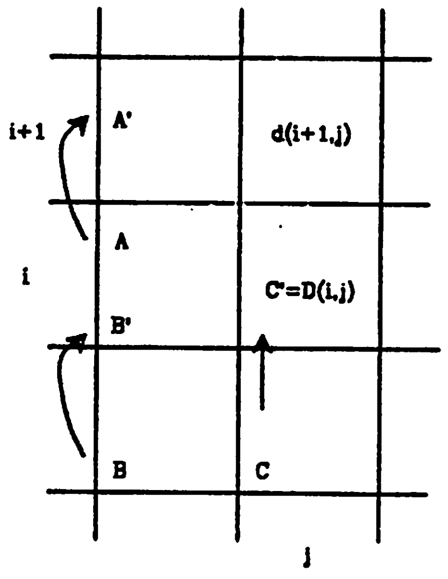


Fig. 4.4b Slow-loop operations

T3: Ad = Ad + 1 ; the next address is selected

T4: M $\rightarrow$ A ; the accumulated score is read to A

Clocking of the registers , enabling of the tri-state output buffers, enabling and disabling the DP memory and incrementing the address counter must also be done in accordance with this sequence of operations. The slow-loop has 6 fast clock cycles to do all the computation. This time is sufficient to do the sequence of operations needed without any difficulty.

4.3.3. Clock and Memory Requirements

Equation 4.1 can be rewritten in the following form:

$$D(i,j) = \sum_{1}^{16}[f_A^k - f_B^k]^2 + \min\left\{D(i-1,j), D(i-1,j-1), D(i,j-1)\right\} \qquad (4.3)$$

Assuming a frame for every 20 msec. of speech, and words of average length of .5 sec., we have 25N frames for a vocabulary of N words. The computation of eq. (4.3) is being done for all the frames in the template memory during the 20 msec. interval between incoming frames.

The clock frequency is given by:

$$f_{clock} = \frac{N\,.\,25\,.\,12}{2.10^{-2}} = N1.5 * 10^4$$

For 500 words of template memory the clock frequency is then: $f_{clock} = 7.5MHz$.

From the above formula we need 7.5 million operations/sec. or 7.5 Mips for the subtraction, 7.5 Mips for the squaring operation and the same number for the additions. Thus we need 22.5 Mips just for the computation of the local distances. This large number can be reduced by the use of parallel operations and

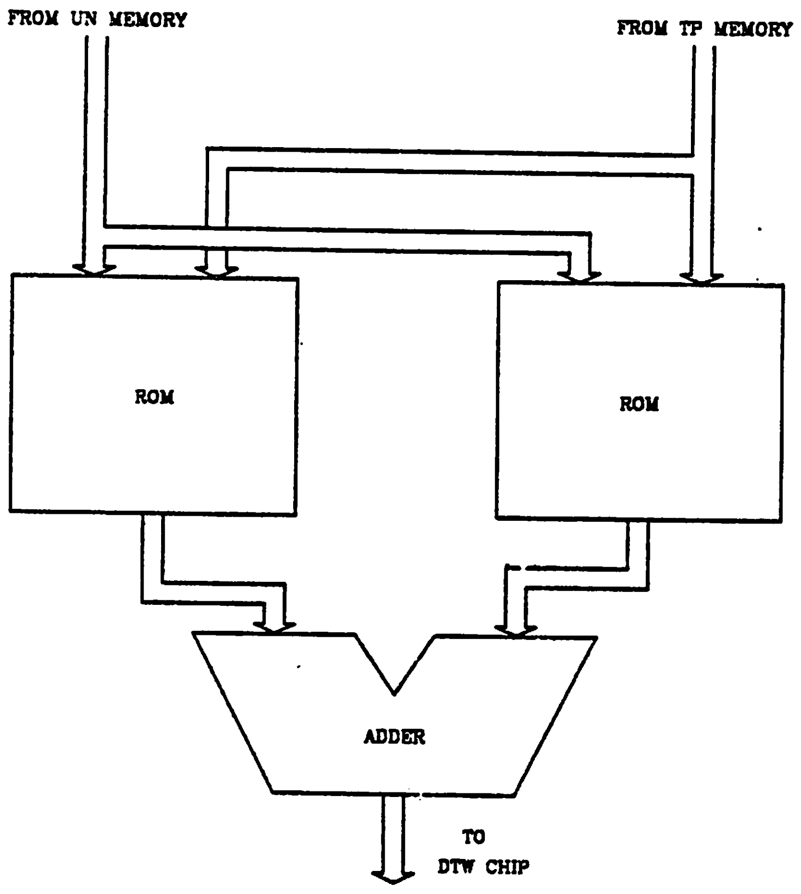


Fig. 4.5 Parallel computation for local distance

the computational rate of the system can be increased by the use of pipeline registers. The features of the frames are stored in groups of 2. Therefore the local distance involving the 12 features of the incoming word frame and of the reference word frame can be computed for the two features in parallel. This is shown in Fig. 4.5. The operations of subtraction and of squaring can be achieved in one clock cycle using a table look-up ROM. Such a ROM would output the

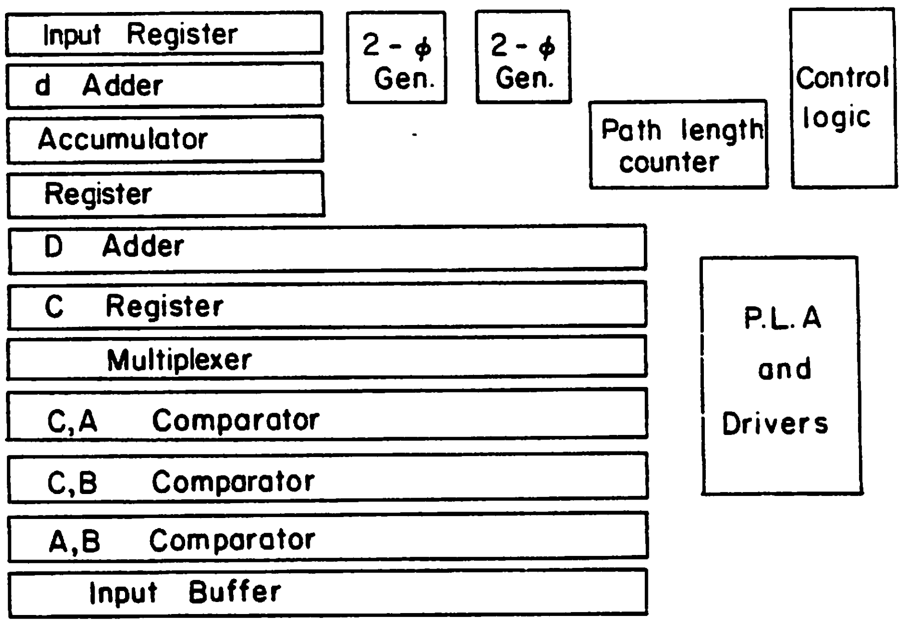


Fig. 4.6 Floor plan of the DTW chip

squared difference between the two 4-bits words representing the features of the incoming and reference frame correspondingly.

The clock frequency is then reduced to 3.75 MHz for the same vocabulary of 500 words. The size of the template memory is 75 k x 8 bits.

The depth of the DP memory is six times less then the template memory as there is one accumulated score per frame. The width of the memory will be 13

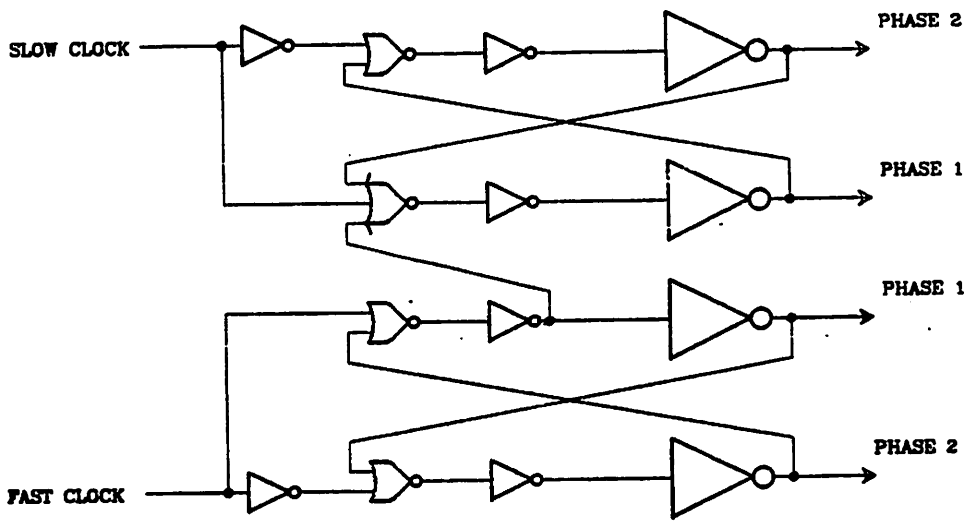


Fig. 4.7 Two two-phase generators

bits in addition of one bit for slope indication and 8 bits for the path length.

4.4. Description of the Circuits of the DTW Chip

The functional blocks of the chip are shown in the floor plan as indicated in Fig. 4.6
The data path consists of a cascade of register and adders. The slow-loop consists of an adder, registers, a multiplexer, three comparators and a PLA. It was decided initially to use two-phase non-overlapping clocks and semi-static registers in the implementation of the circuit. Two such phase generators are needed in the circuit for the fast and for the slow clock. A path length counter and a control logic circuit are also realized.
To increase testability it was decided to make the connections between the different sub-systems using the external pins of the chip. In case of an error in one sub-system, it would be possible to implement its function using an external circuit.

The input to the chip are partial sums of the square differences between the feature of the incoming frame and the corresponding feature of the reference frame. six such partial sums are serially input to the adder at a 3.75 MHz rate. The accumulation of these partial sums form the local distance. The output of the chip are the 13 bit wide accumulated distance for each frame, one bit indicating slope constraint and eight bits indicating the path length associated with this frame when in connected speech recognition mode.

4.4.1. Two-Phase generators

The chip uses two clock rates. One for the computation of the local distance at a rate of 3.75 MHz. The other for the computation of the accumulated dis-

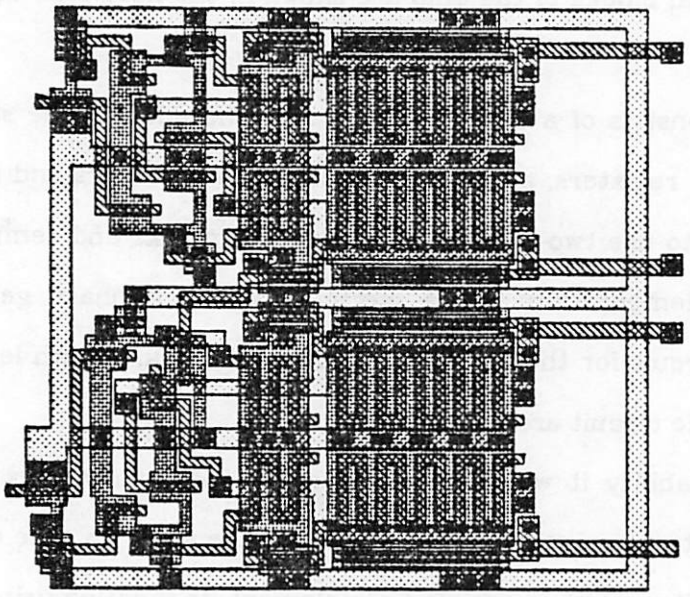


Fig. 4.8 Plot of the two two-phase generators

tance which is 8 times slower. The master clock and the divide-by-eight counter deriving the slow clock are outside the chip. Thus we have two single phase clocks as input to the chip and two two-phase generators on chip.

To prevent error in the operation of the chip due to possible skew between the clocks, the phases of the slower clock are synchronized to those of the faster clock. This is shown in Fig. 4.7 The clock drivers were designed to drive about 5 pF load capacitors on each phase. The output of the phase generators are brought back to external pins of the chip.

The plot of the two two-phase generators is shown in Fig. 4.8

4.4.2. Registers

The registers on the chip are of the Master/Slave semi-static variety The electrical configuration is shown in Fig. 4.9a while the layout is shown in part b.

The semi-static configuration enables us to test the operation of the register when the clock is stopped. In this case, the phase $\Phi 1$ is high and the output of the slave section is latched. This configuration was used with varying modifications for every M/S register. Thus it was used for implementing the input register, the pipe line 2 register, the accumulator, the C register, the F/F used for the controller and for the register in the Lx counter.

The accumulator used in the d_i adder needs a provision for clearing its content. The clear operation is done using a NOR configuration instead of a buffer in the Slave section. This is shown in Fig. 4.10.

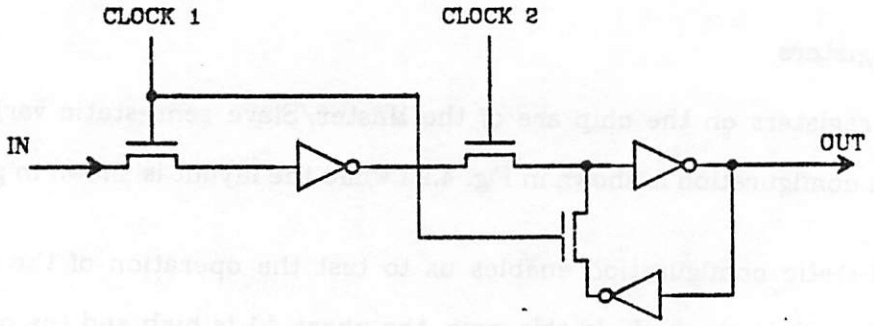


Fig. 4.9a Schematic diagram of M/S register

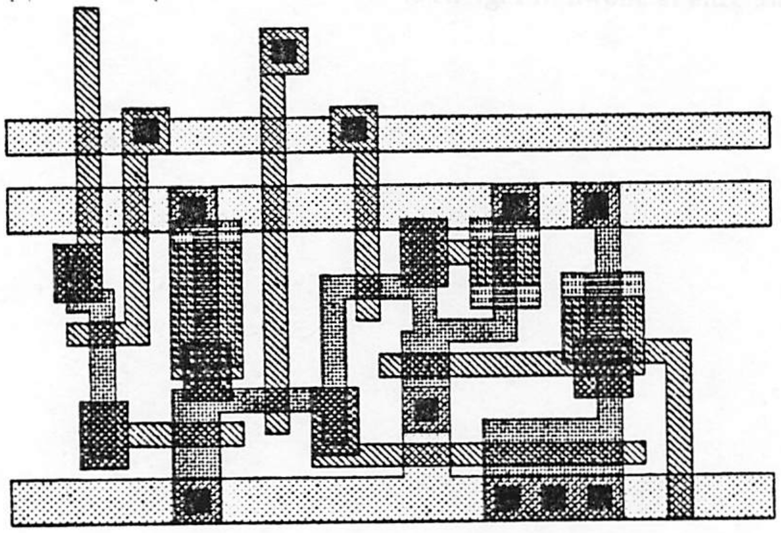


Fig. 4.9b Plot of layout of M/S register

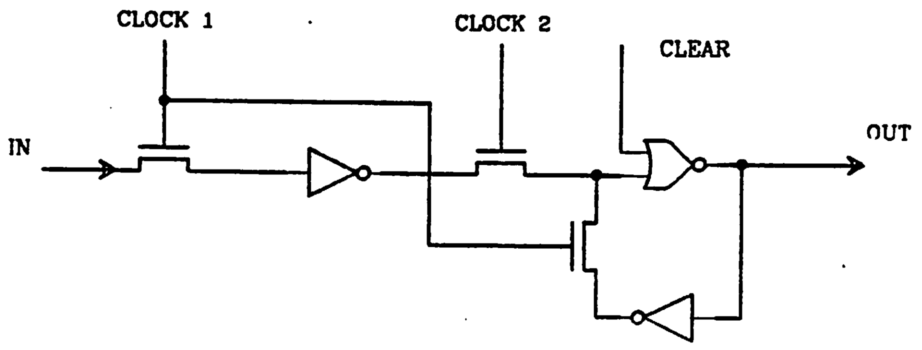


Fig. 4.10 Accumulator

The "CLEARAC" signal clearing the accumulator occurs when the input register issues the first partial sum to the "d ADDER". This signal is derived from the external control unit.

4.4.3. Latches

Registers "A REG" and "B REG" can be implemented using latches. The latches are also of the semi-static type. The latch configuration used for register A and B appears in Fig. 4.11. The advantages of using latches are their small size and easy control. This configuration requires only one clock line.
The output of "A REG" and "B REG" are connected to the comparators and to the multiplexer. A plot of one slice of latch A and B appear in Fig. 4.12.

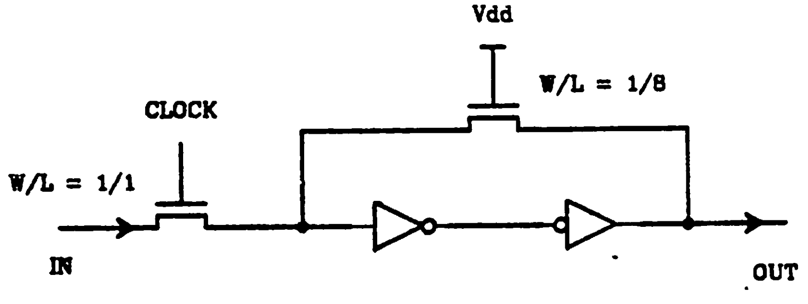


Fig. 4.11 Semi-static latch

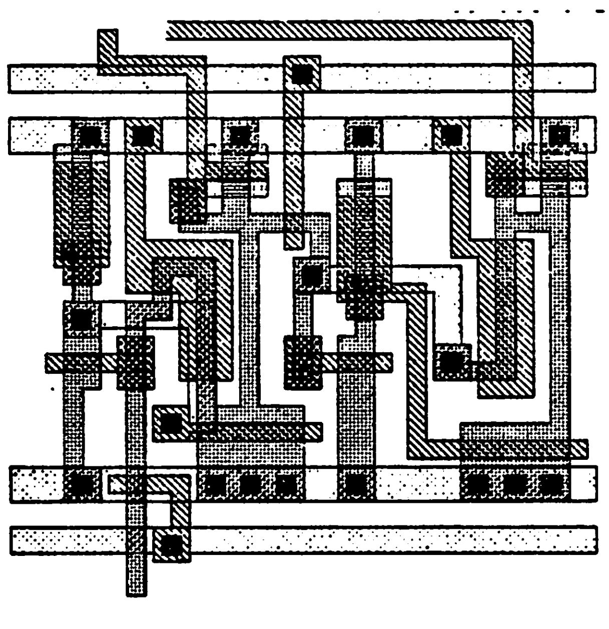


Fig. 4.12 PLot of "A B REG".

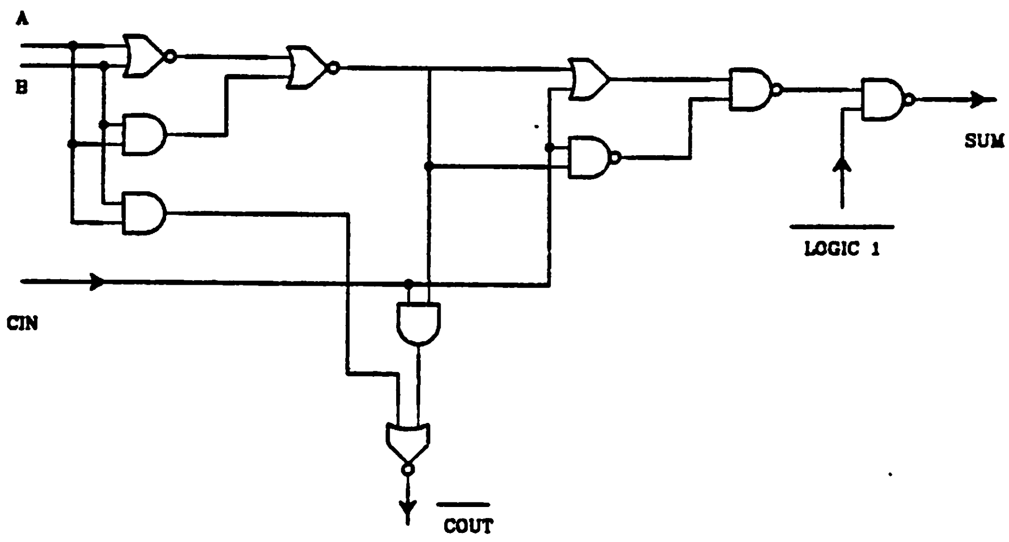
4.4.4. Adders

Both adders on the chip are ripple adders. Reduction of the propagation time of the carry is done by having one inversion only between the carry-in and the carry-out. This can be achieved by designing full-adder cells having carry-in and carry-out of opposite polarity. We need then two different cells: one for odd bits and one for even bits. For small adders this configuration is as fast as the Manchester carry chain. The two full adder cells are shown in Fig. 4.13. The adders are designed to be of the saturating type. This means that with the appearance of an overflow at the output of the adder the sum outputs of all the cells is forced high. This was obtained by adding a circuit activated by the carry out of the last cell. The output of this circuit controls a gate added to the sum output of all the cells. This signal is labeled $\overline{\text{LOGIC}}$ in Fig .4.13 and Fig. 4.14. The plots of the two cells is shown in Fig. 4.14.

4.4.5. Comparators

The three comparators needed on the chip compare the amplitude of the content of the three registers "A REG", "B REG" and "C REG". The comparators are also of the ripple type. The ripple time of the carry is reduced using the same technique as in the adders. In this case the configuration is the same for both the even and odd cells. The comparator is shown in Fig. 4.15 The two cells differ only by the signals at their inputs.

4.4.6. Multiplexer



a)

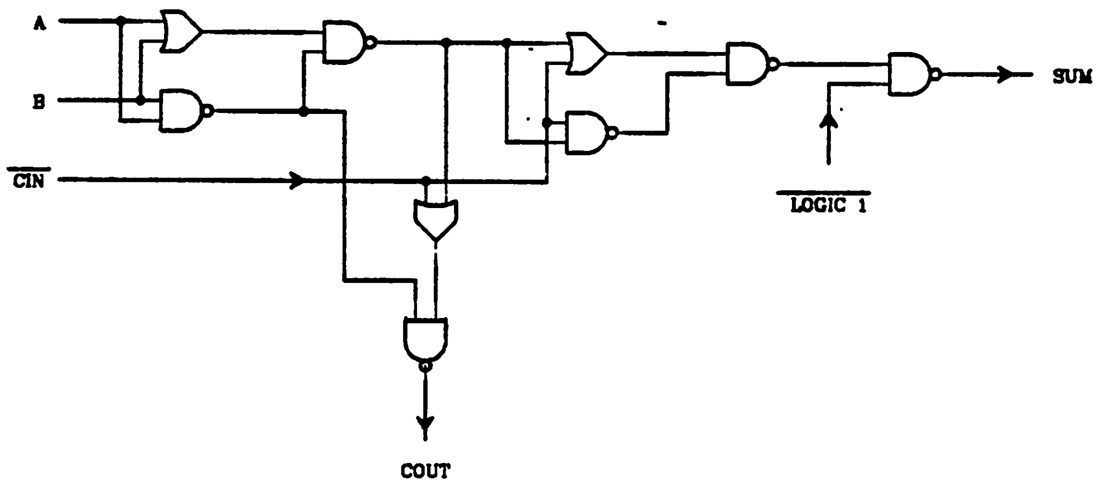
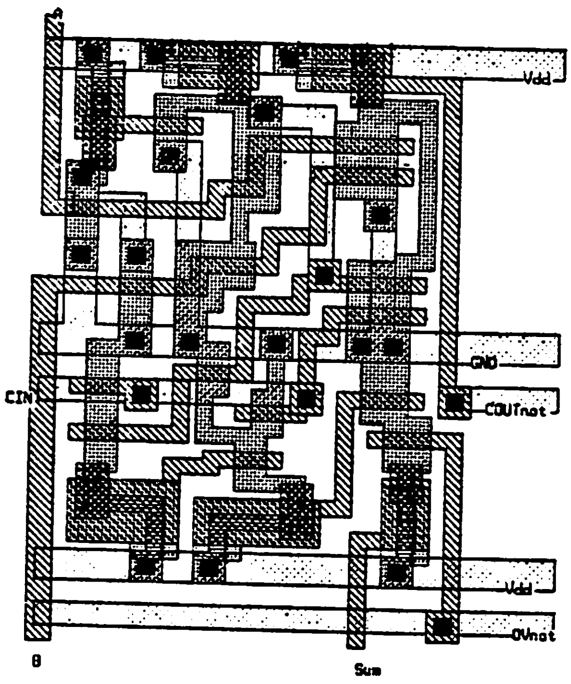


Fig. 4.13 Full adder cells a) odd bit b) even bit



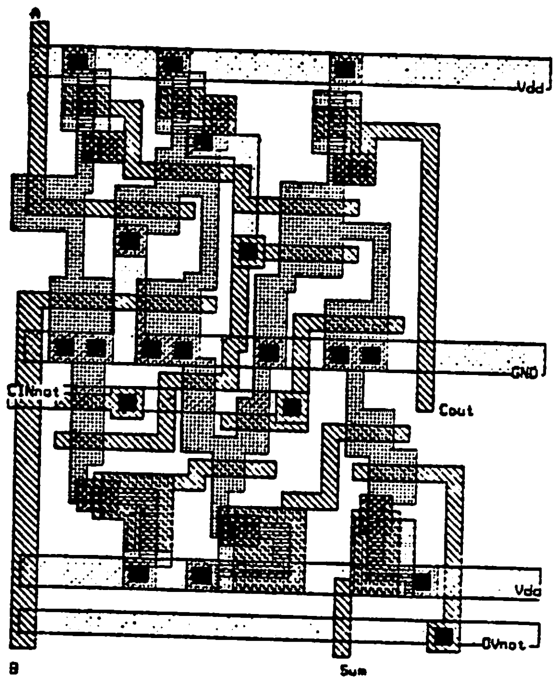
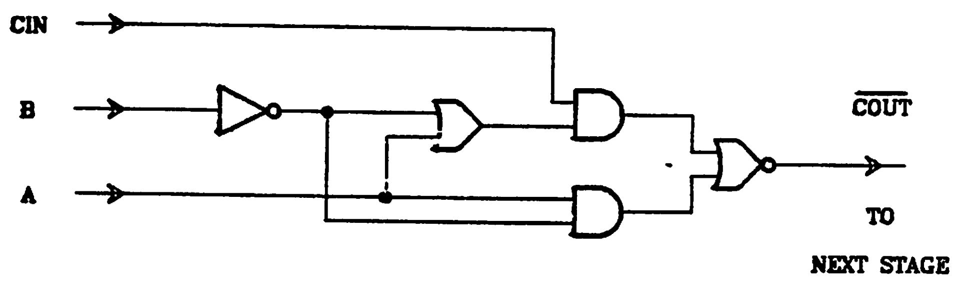
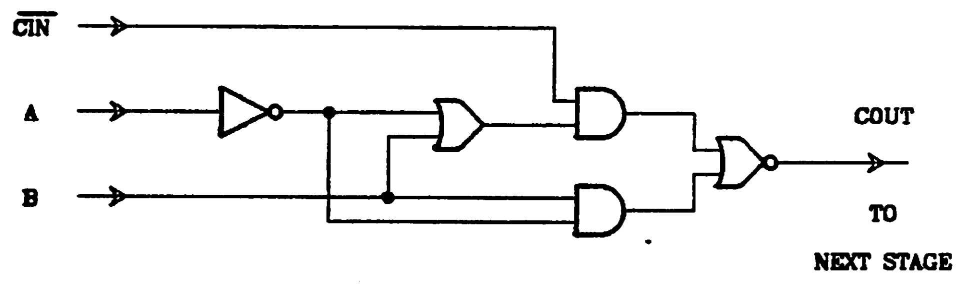


Fig. 4.14 Plot of full-adder cells a) odd bit b) even bit



a)



b)

Fig. 4.15 Comparator a) odd bit b) even bit

The multiplexer selects one signal out of three depending on the state of the control lines. The controls coming from the PLA trough buffers are: "Asel", "Bsel" and "Csel". An additional signal called "CLEARMUX" forces the output of the multiplexer to be low. This is needed when computing the accumulated score for the cell in the first column and in the first row. As there is no accumulated score yet at this point the output of the multiplexer should be zero. The basic configuration of the multiplexer is shown in Fig. 4.16.

4.4.7. PLA operation

The circuit implementing the different conditions appearing during the computation of the accumulated score is realized in a PLA. Its outputs control the multiplexer, the D ADDER, the path length counter and the slope bit "Cslope".

The inputs to the PLA are the results of the magnitude comparators. In addition to the outputs of the comparators it is necessary to know whether the present computation is done for the cell in the first row in the matrix or in the first column of the matrix. Another input to the PLA indicates whether the system is used for isolated word recognition or for connected speech. During the design of the chip, it was decided to leave the option of using slope constraints in the computation of the paths. This is used in some algorithms to constrain the resulting path to meaningful regions. The slope is constraint to remain between 2 and 1/2. This means that the path cannot have two successive horizontal movements nor two successive vertical movements.

To obtain this option we need to devote one bit to indicate if the adjacent accumulated score came from a vertical or horizontal cell or not. This bit is called slope bit and for the two accumulated scores in the A and C registers is called Aslope and Cslope respectively.

The PLA has the following inputs:

1- result of the comparison between C and A ($C<A$). True when high.

2- result of the comparison between C and B ($C<B$). True when high.

3- result of the comparison between A and B ($A<B$). True when high.

4- Aslope indicator. When high, enables the comparison of A . Otherwise A is not considered in the comparison.

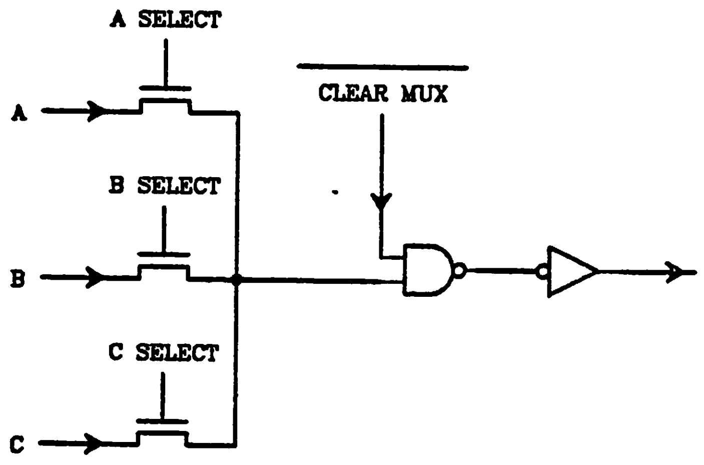


Fig. 4.16a Multiplexer

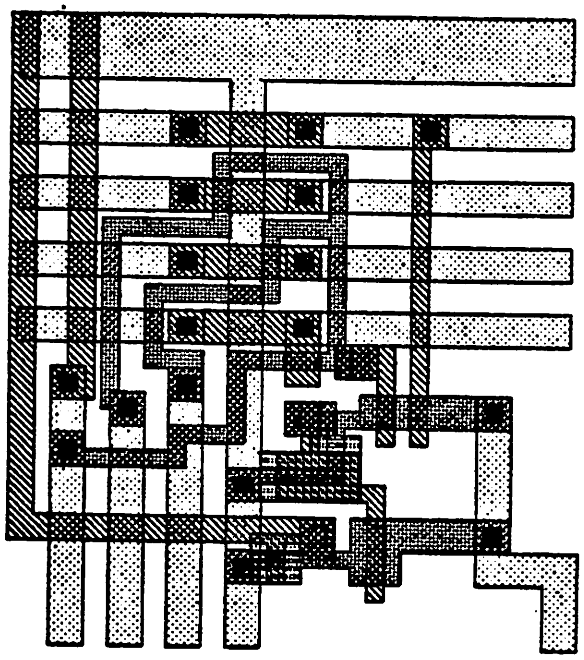


Fig. 4.16b Plot of the layout

5- Cslope indicator. When high, enables the comparison of C . Otherwise C is not considered in the comparison.

6- first row. True when high.

7- first column. True when high.

8- isolated word mode or connected speech mode (IWR/CS)

9- B Select, the inversion of the output $\overline{\text{BSelect}}$.

The size of the PLA is determined by the number of inputs and outputs and by the number of minterms. Reduction of the number of minterms is done using an external inverter to invert the Bselectnot output and re-insert it as input. This means having two PLA in cascade. The speed of operation is lowered - there is still enough time in the slow loop - however the size of the PLA is cut to almost half.

The equations describing the operation of the PLA are the following:

$$\text{Aselect} = \text{Aslope} \,.\, A<B \,.\, (\overline{C<A} + \overline{\text{Cslope}}) \,.\, \overline{\text{fcolumn}};$$

$$\text{Cselect} = \text{Cslope} \,.\, C<B \,.\, (C<A + \overline{\text{Aslope}}) \,.\, (\overline{\text{frow}} + \overline{\text{IWR}});$$

$$\overline{\text{Bselect}} = \text{Aselect} + \text{Cselect} + \text{fcolumn} + (\text{frow} \,.\, \text{IWR});$$

$$\text{Clearmux} = \text{frow} \,.\, (\overline{\text{IWR}} + \text{fcolumn});$$

$$\text{Noselect} = \text{Aselect} + \text{Bselect} + \text{Cselect} + \text{frow} \,.\, \text{fcolumn};$$

$$\text{SetCslope} = \text{Bselect} + \text{frow} \,.\, (\text{fcolumn} + \overline{\text{IWR}});$$

The PLA has six outputs: four of them, "Aselect" ,"$\overline{\text{Bselect}}$ " , "Cselect" and "Clearmux" control the multiplexer. The two other outputs are: "$\overline{\text{Noselect}}$" and "SetCslope". The output "Clearmux" is needed when the computed cell is at the position of first row and first column.

The output "$\overline{\text{Noselect}}$" is needed when the computation takes place in the region

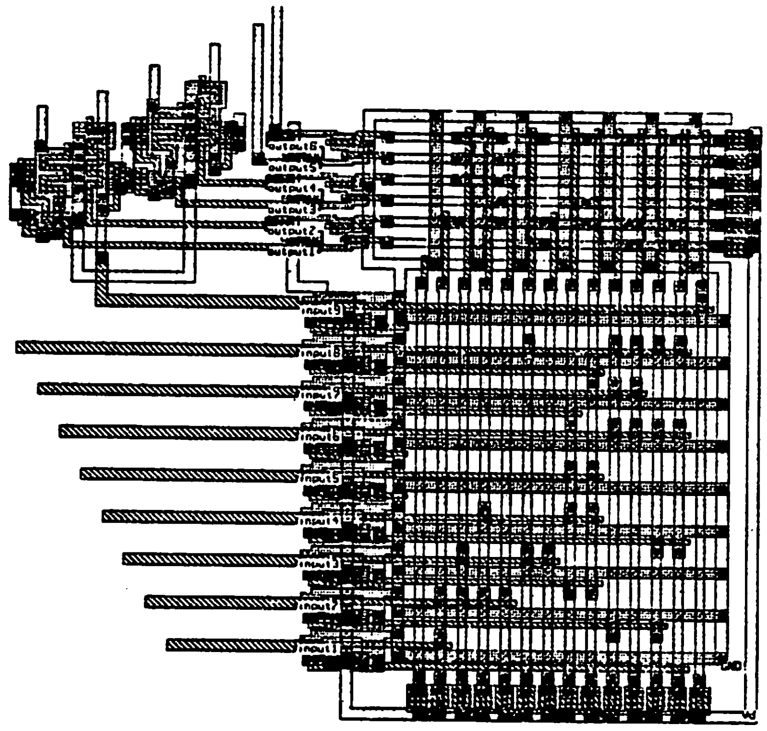


Fig. 4.17 PLA and drivers

where the slope is greater than 2 or smaller than 1/2. In this case, none of the three words is selected. This PLA output forces the D adder into its saturating mode thereby giving a score equivalent to infinity to these locations.
The output "SetCslope" sets the bit enabling the selection of C in the comparison. The option of having a slope constraint can be overrided by the external input " Slope override". This external input is combined with the output of the PLA and always sets the Cslope bit to 1.

The plot of the PLA with its associated drivers is shown in Fig. 4.17.

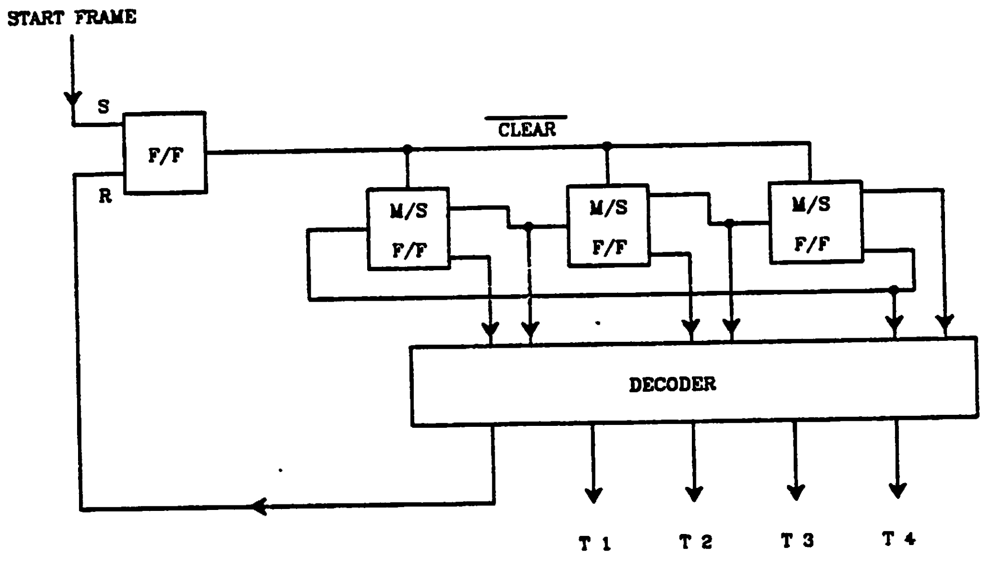


Fig. 4.18 Block diagram of the controller

4.4.8. Controller Operation

A controller is needed to control the operation of the tristate pads connecting the DP memory to the chip, to latch the data in the A and B registers and to control the address counter, R/W and CS controls of the DP memory (not on chip).

The block diagram of the controller is shown in Fig. 4.18. The controller has been realized as a Johnson counter to obtain outputs which are glitch free. The outputs of the counter are decoded using Nor gates to obtain the required time

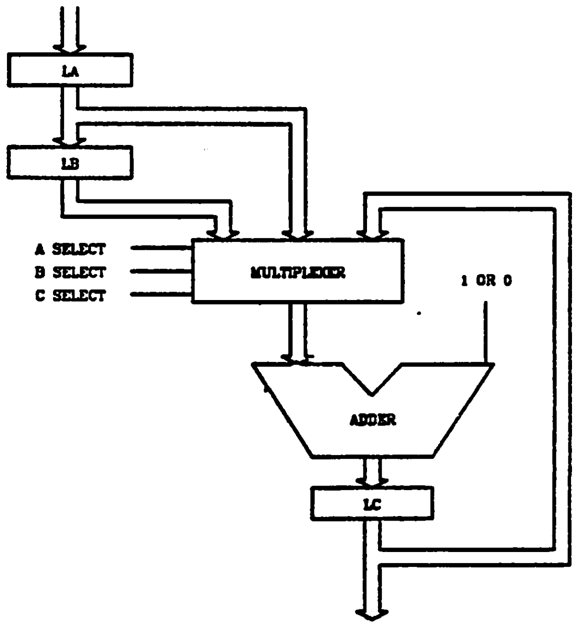


Fig. 4.19 Path length counter

signals. For the 4 required control signals we have to use 3 M/S flip flops. The operation of the controller is started using an external signal. This signal triggers a static latch enabling the counter. The last state of the controller stops its operation by retriggering the latch.

4.4.9. Path length counter

The horizontal projection of the the path is computed in the path length counter shown in Fig. 4.19.
The path length associated with the previous accumulated scores are entered in the registers L_A and L_B. The multiplexer is controlled by the outputs of the PLA "Asel" and "Bsel". The correct value of the path length is selected accordingly. This path length is incremented by one using an 8 bits half-adder and stored in a latch. The content of the latch is then sent back to the DP memory. When neither A nor B are selected the path length associated with that cell is the same as the previous one. The content of the latch is therefore sent unchanged to the DP memory.

A slice of the circuit for the computation of the accumulated score which comprises the D adder, the Cregister, the multiplexer, the three comparators and the two latches A and B is illustrated in Fig. 4.20. To be able to probe the circuit in case of non-operation tests pads have been added at critical points. Two such points are shown in the plot. These are located between the adder and the C register and between the output of this register and the multiplexer. The input to the A register is obtained from the tri-state pad after passing through a supper-buffer (not shown on the plot). The output from register C to the DP memory is connected directly to the amplifiers on the tri-state pad.

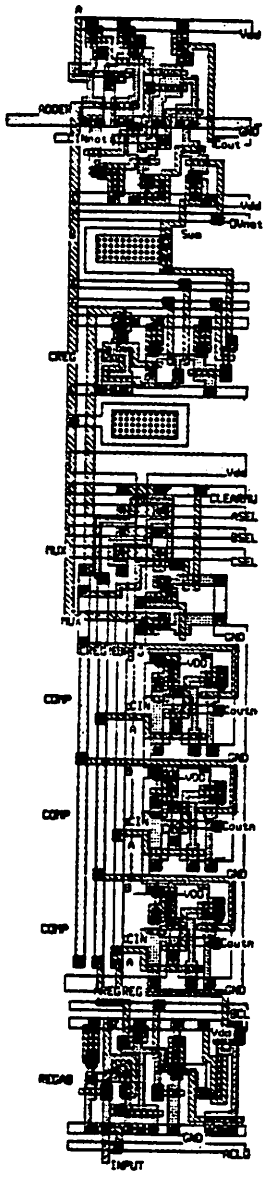


Fig. 4.20 One slice of part of the slow-loop

A picture of the chip is shown in Fig. 4.21

4.4.10. Other Circuits

Other circuits implemented on the chip include the gates for combining the signals activating the saturation of the adders, the gates combining the external signal overriding the slope constraints, inverters clearing the accumulator and the buffers at the input side - chip side - of the tri-state pads.

4.5. Results and Performances

The design of the DTW chip made extensive use of the availability of the CAD tools developed at UC Berkeley. All the cells - except the I/O pads - were of original design. The I/O pads for : input, output, tri-state, Vdd and Ground were taken from the Stanford Cell Library.

4.5.1. Design Considerations

Some particular points of the design were:

(1) To pinpoint possible malfunctions in the chip, test pads were placed at selected locations in the circuits. These test pads were useful when testing the first fabricated chips.

(2) The path of the clock signals was designed to parallel the path of the data in order to eliminate possible clock skew.

(3) The initial design was done using butting contacts for connections between the diffusion and polysilicon layers. Later, the cells and pads were modified using buried contacts.

(4) Careful consideration was given to long lines on the chip. The optimal width of these lines has to be chosen according to the capacitance of the load, the

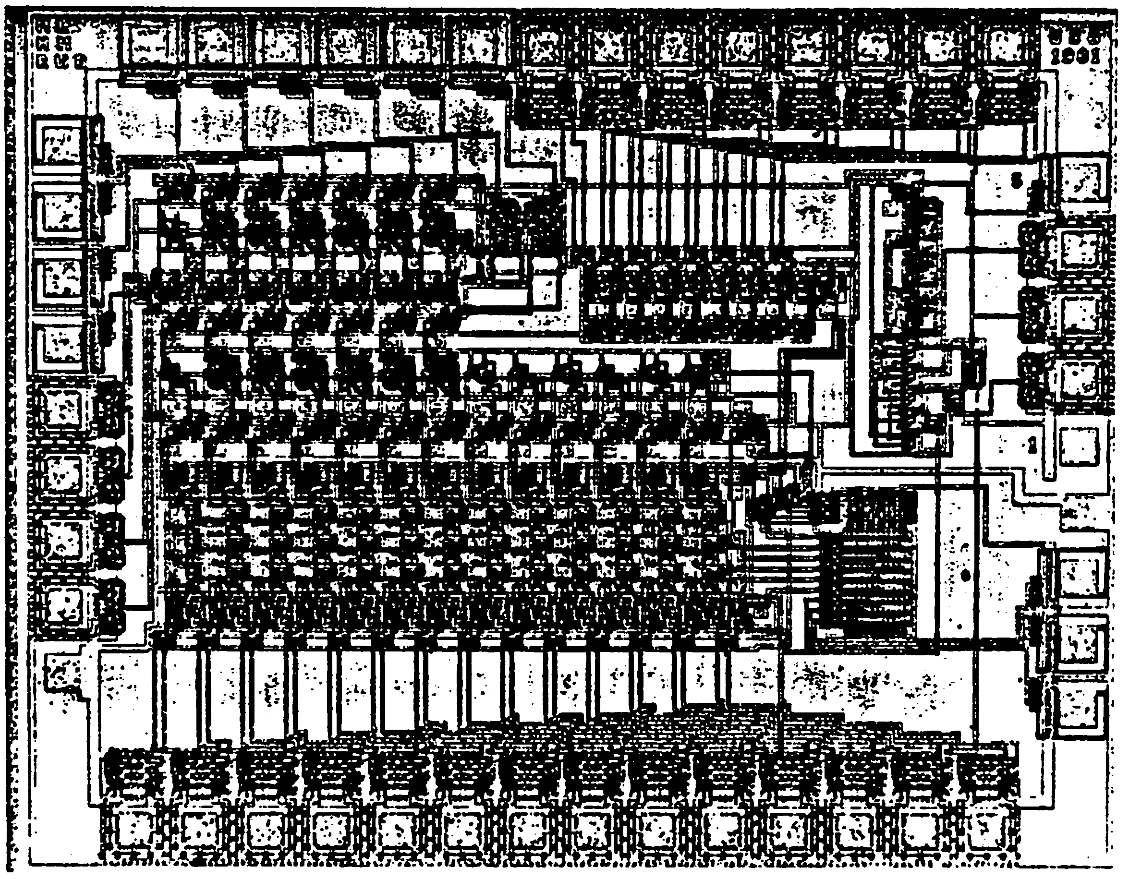


Fig. 4.21 Microphotograph of the chip

resistance and capacitance of the line and the driving current charging the capacitances. The time constant is given by:

$\tau = (R_{load} + R_{line})(C_{line} + C_{load})$

The resulting optimal line width is:

$$W_{opt} = \sqrt{\frac{k_1 C_{load}}{k_2 R_{load}}}$$

where k_1 and k_2 are the constant for resistance per square and capacitance per unit area respectively, and R_{load} is the resistance of the current source driving the line.

At the time of calculations the Spice parameters for circuit simulation were not known as they are dependent on the fabrication run and on the manufacturer. A worst case was assumed which resulted in higher speed of operation than designed for. Fig. 4.22 shows the waveform obtained for the two phases of the fast clock phase generator under normal operation conditions.

4.5.2. Results

In order to compare our system to others the present system has been tested under controlled conditions. The results show that a cooperative speaker can obtain 99.5 % or more correct recognition using a vocabulary of commonly used words. Research is continuing to improve the recognition accuracy. In particular effort is directed in the recognition of the letters of the English alphabet which is considered a difficult task.

The chip was fabricated using a standard NMOS process with a single layer of metal and polysilicon. The minimum feature is 5μ.

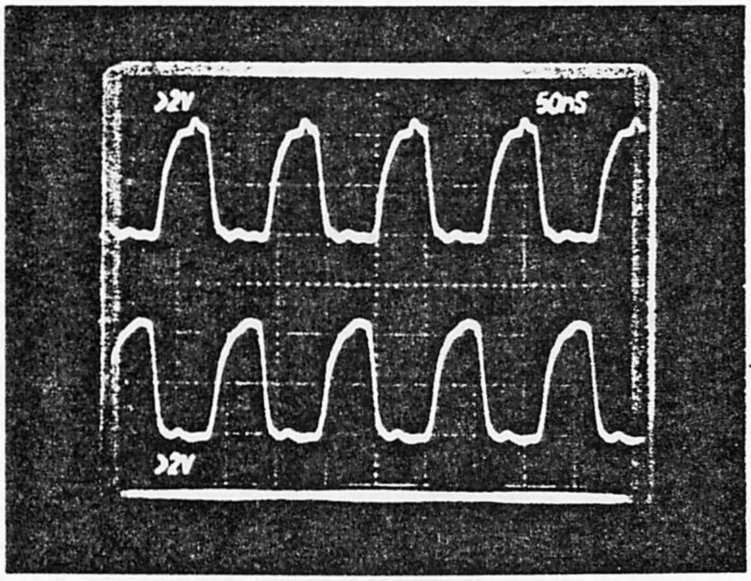


Fig. 4.22 Output waveform of the clock generator

The DTW chip was designed using simplified design rules rules. All the cells and pads on the chip were simulated using time and logic simulators. The chip measures 3.6 mm by 4.2 mm. Current consumption at 5 Volts is around 86 ma. The current distribution among the different cells is the following:

7 bit adder - 450 μa.

13 bit adder - 0.85 ma

PLA - .86 ma

TriState Pad - 3 ma

Output Pad 1.2 ma

Clock generators - 1.5 ma

70% of the current consumption is due to the output and tri-state pads.

Testing the chip was done done using a tester developed at UCB. This tester uses a VAX/780 as host computer. The chip was tested at the maximum frequency of, the tester which is about 4 MHz .

After connecting the DTW chip in circuit and with the experience gained, different improvements were suggested as well as additions to the chip and modifications to the basic design. The changes for the new version of the DTW chip are presented in chapter 5.

CHAPTER 5

Improvements to the Dynamic Time Warp Circuit

5.1. Improvements and Modifications of the Chip

Additional circuits are being added to the present DTW chip to allow reduction of the complete recognition system to a small number of integrated circuits. New developments in continuous speech recognition algorithms also require modifications of the existing circuits by adding sub-circuits and increasing clock rates. The new circuits were designed by Tobias Noll on sabbatical leave from Siemens central research laboratories.

Degradation occurs in continuous speech recognition when processing the speech frames with the downsampling technique used in isolated word recognition[38]. To maintain high recognition accuracy it is necessary to retain the original sampling period of 10 msec. or less thereby decreasing the number of words that can be recognized in real time. For this reasons, we need to increase the rate of data processing in the system to maintain the original vocabulary size.

The modifications can be divided in two groups: those needed to increase the processing speed on the chip and those increasing the number of functions realized on chip.

5.2. Increasing the Processing Speed

5.2.1. Clock Rate Limitations

The processing rate of the system is limited by the characteristic features of the circuits external to the chip. These features are the Template memory cycle time and the bus access time. This limits the clock rate to about 5 MHz.

The new chip is designed to recognized a vocabulary of 1000 words in real time. Assuming an average word of 25 frames of 20 msec., this means processing 25000 frames in 20 msec. while still maintaining a clock rate of 5 MHz. Therefore the 16 features of each frame must be processed in four cycles. The circuits limiting the rate of data processing are those in the "slow loop". The "slow loop" computes a score during four clock cycles.

It is relatively simple to increase the speed of operation of this loop to about 400 ns or even less. At that clock rate, it is possible to process 50,000 frames in 20 msec. or 2000 words having an average length of 25 frames.

5.2.2. Increasing the Speed of the Circuit

To increase the processing rate, the adders and PLA on chip are redesigned. The speed of the adders is increased by increasing the power of the load transistors in the carry path and reducing the size of the transistors in the sum path, thereby reducing the capacitances. The speed of the PLA's, used as ROM's, is also increased by increasing the power of the load transistors.

Another way to increase speed is to use a multiplier to square the difference obtained from the subtractor instead of using a PLA. By using carry save technique, the result of each cell is passed to the next row without waiting for the carry to propagate along the row. It is therefore possible to insert pipeline registers between the rows to increase the rate of computation.

5.2.3. Slow-Loop Consideration

By operating in parallel on four features at a time, we have reduced the slow-loop computation time to four fast clock periods. The delays occuring in this loop are due to a 15 bit adder, a 15 bit comparator, PLA and multiplexer propagation times. The sequence of operations needed in the slow-loop must also be modified to squeeze the write and read cycle times during this interval.

Efficient use is made of the time available by writing to the DP memory during the computation time of the slow-loop. Therefore the sequence of operations will begin by reading the stored accumulated score from memory to register A so that the computation in the loop begins immediately. The next operation consists of decreasing the address by one and writing the contents of register C to memory. The address is then increased by two for the next readout operation.

The sequence of operations is the following:

1- Write → Read ; Changing mode of memory
   A → B ; Transfer content of A to B
   DP memory → A ; Stored accumulated score readout to A

2- Address = Address -1 ; Address to store present accumulated score

3- Read → Write ; Changing mode of memory
   C → DP memory ; Accumulated score is written in correct location

4- Address = Address + 2 ; Address for next computation

This sequence of operations requires an adder to modify the address. Further increase in speed is possible by using two adders. The two adders are increased simultaneously but their address is offset by one. These addresses are then stored in registers. In this way the address for writing and reading are stable and available from the multiplexer.

The redesigned adder is fast enough so that there is no need to use two adders to the circuit.

5.3. Additional Circuits on the Chip

The following circuits have been added to the DTW IC:

(1) a 4x16 bit memory to store the features of the incoming frame

(2) the circuit computing the squared difference between the corresponding four features of the incoming and reference frames. Each channel consists of a subtractor, pipeline register and look-up table. The output of the four channels are added, pipeline fashion, using three additional adders.

(3) the circuit indicating the boundary between templates and also the end of the last template in the template memory

(4) a register storing the initial conditions appearing in isolated or in connected word recognition mode.

(5) the address generator circuit for both the template memory and DP memory.

(6) a finite state machine controlling the memory on chip and generating signals for interfacing the chip with the microprocessor.

(7) a timing circuit generating the time signals for the different circuits on the chip

5.4. Description of the New Circuit

The new circuit is redesigned to use 4μ technology. The test pads have been removed and the circuit uses dynamic registers. These modifications result in

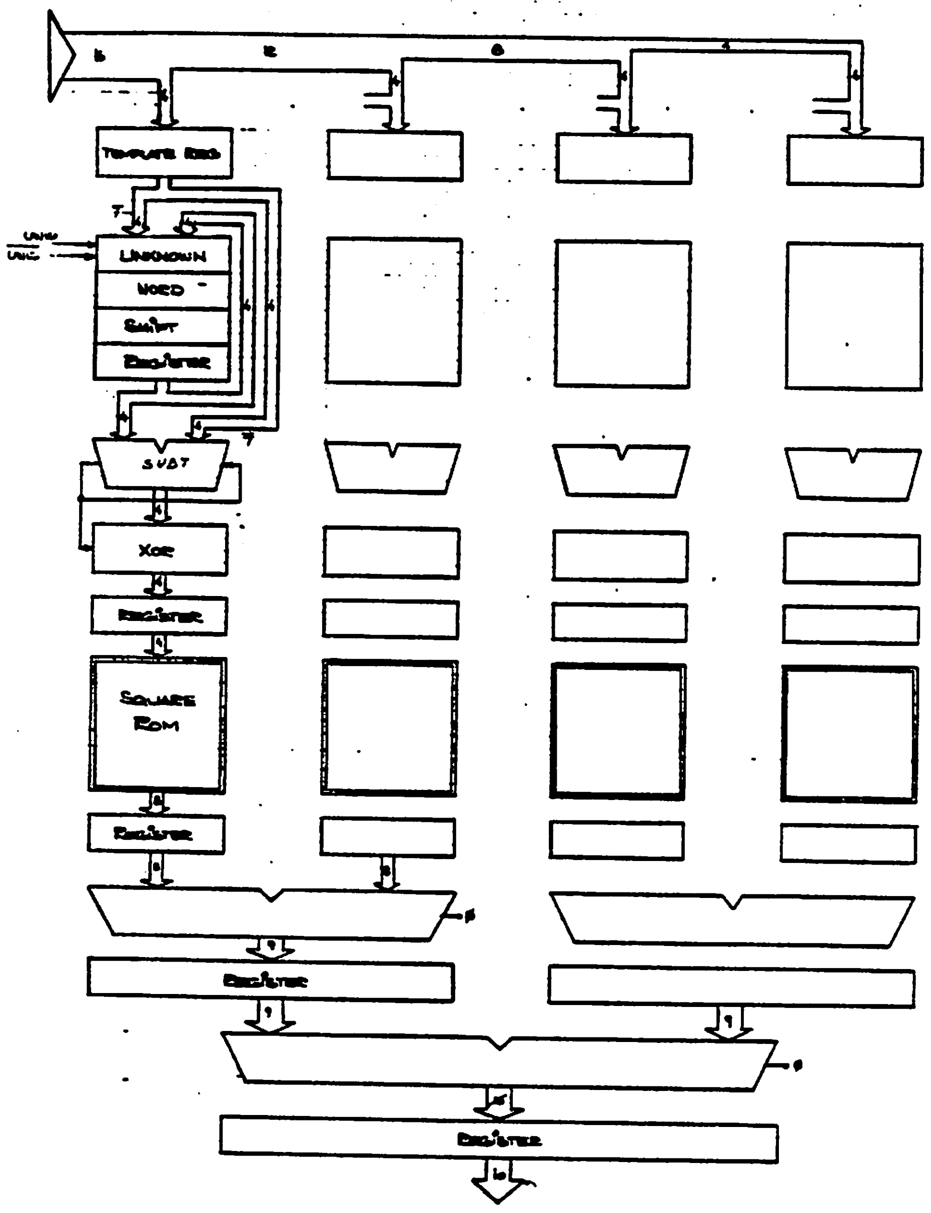


Fig. 5.1 Computation of Euclidean distance

keeping the size of the active circuit to 4.32 mm side. The block diagram of the new chip is shown in Fig. 5.1 and Fig. 5.2. Fig. 5.1 shows the section realizing the unknown frame memory and computing the Euclidean distance, while Fig. 5.2 shows the functional block diagram of the DTW algorithm and memory address circuits.

5.4.1. Euclidean distance computation

Refering to Fig. 5.1, the data is taken from the external bus and processed by four channels in parallel . The data consists of a 16 bit word presetting the conditions for the algorithm and of four 16 bit words containing each four features. The preset word is stored in the preset register (this register is in Fig. 5.2) and the four words representing the features of the incoming frame, are loaded in the shift registers forming the unknown frame memory.

The data is then read from the template memory. Each channel computes the difference between the corresponding features. This difference is squared using the ROM look-up table. The results are added together to produce a partial distance.

Refering to Fig. 5.2, the partial distances are added during four clock cycles to obtain the Euclidean distance between the frames. This distance is then processed to obtain the accumulated score in the same way as described in chapter 4.

5.4.2. Slow Clock Unit

The clocking signals needed in the slow loop circuits are generated by a synchronous counter and decoders in the Slow Clock Unit. This unit also generates the two lower bits for the address of the template memory, timing signals

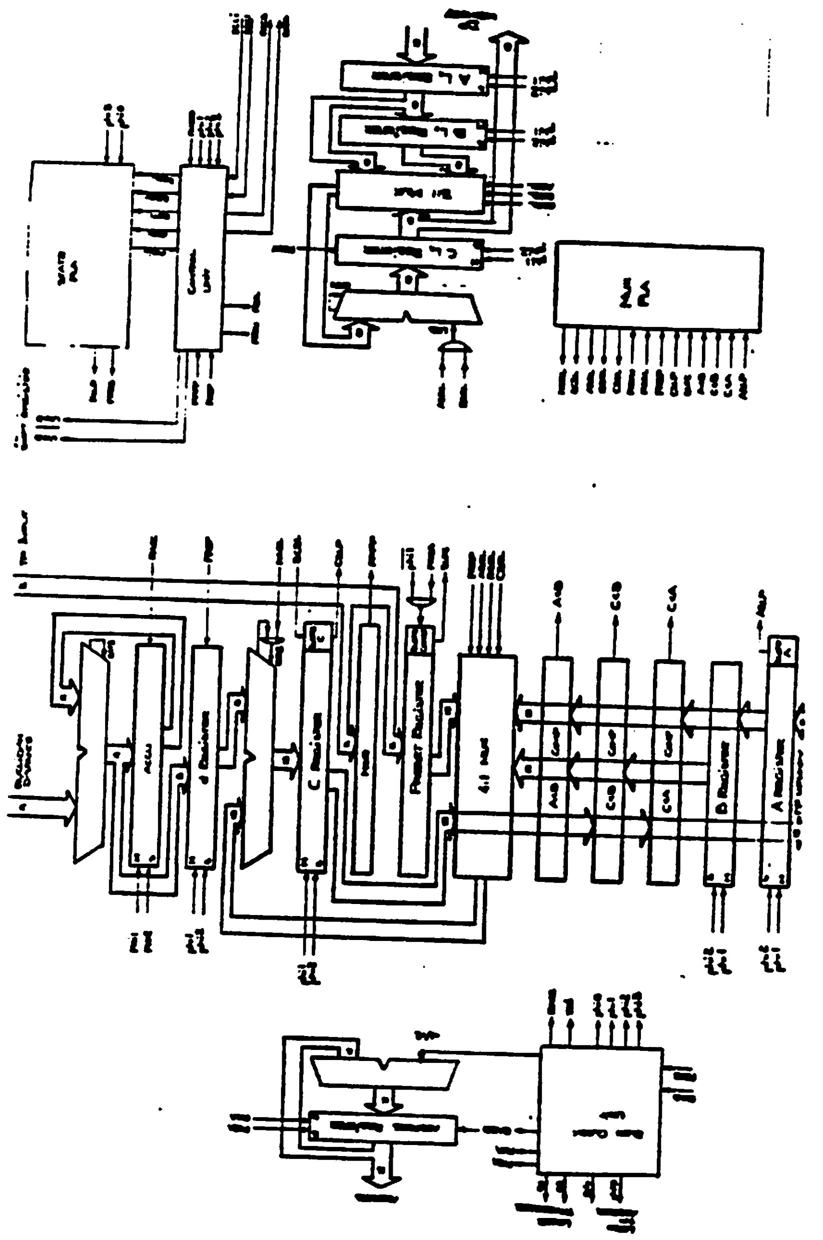


Fig. 5.2 Functional diagram of the DTW chip

for latching externally this address, and control signals for the DP memory.

5.4.3. Memory Address Generation

The address for both the template and DP memory is generated using one circuit. This circuit generates a 16 bit address at the beginning of every slow clock period (one slow clock = four fast clock cycles). These 16 bits are combined with the two lower bits of the previous unit to form an 18 bit address for the template memory. Meanwhile, the 16 bit address is modified to either read from or write to the DP memory according to the state of the slow loop. When no recognition is taking place, the DP memory is continually read to refresh its content while the write signal is inhibited.

5.4.4. Control Unit

A finite state machine (FSM), generates the various signals needed to control the unknown memory, the slow clock unit, and the interface with the microcomputer in the system. This circuit is realized using a PLA a register and a control unit delaying the signals according to the pipelined flow of data. The operation of this unit is controlled by signals from the microprocessor. The FSM also generates signals controlling the state of the DP memory.

CHAPTER 6

Summary and Conclusions

The advent of custom designed LSI and VLSI circuits opens new opportunities for the realization of algorithms on chip. Due to advances in technology and in CAD tools it was possible to realize a small size, inexpensive but accurate, speech recognition system. This first version was used as an experimental tool.

The dynamic time warp algorithm was implemented first in integrated form as this resulted in the largest reduction in computation rate. The results obtained with our speech recognition system demonstrate the advantage of using special circuits to solve the problem of high computation rate.

When correctly trained for a cooperative speaker, our system is able to recognize in real time, isolated words from a 500 words vocabulary with an accuracy of above 99 %. The recognition accuracy for continuous speech recognition is not as high. Research is still carried on the algorithms for continuous speech recognition.

Our system is only a step towards the more ambitious goal of speaker independent continuous speech recognition. To increase the performances of our system, some of the improvements needed are the following:

(1) Additional processing of the speech signal yields information about the speaker independent transient consonants as described in [67,68]. This and additional time domain information should increase recognition accuracy if it could be effectively added to the results obtained from the spectral analysis.

(2) Clustering techniques could be used for special vocabularies to greatly reduce the amount of storage needed in the system without impairing the recognition accuracy. For small, special vocabularies this could mean a complete recognition system one one chip. a complete recognition system one one chip.

(3) Additional tools are needed to speed up the design and testing of LSI/VLSI integrated circuits.

APPENDIX A

Filters Frequency Response

This appendix presents the frequency response of the seven lowest bandpass filters of the filter bank using the approximated coefficients of table 3.2 . For comparison , the transfer function obtained with infinite precision coefficients is also indicated.

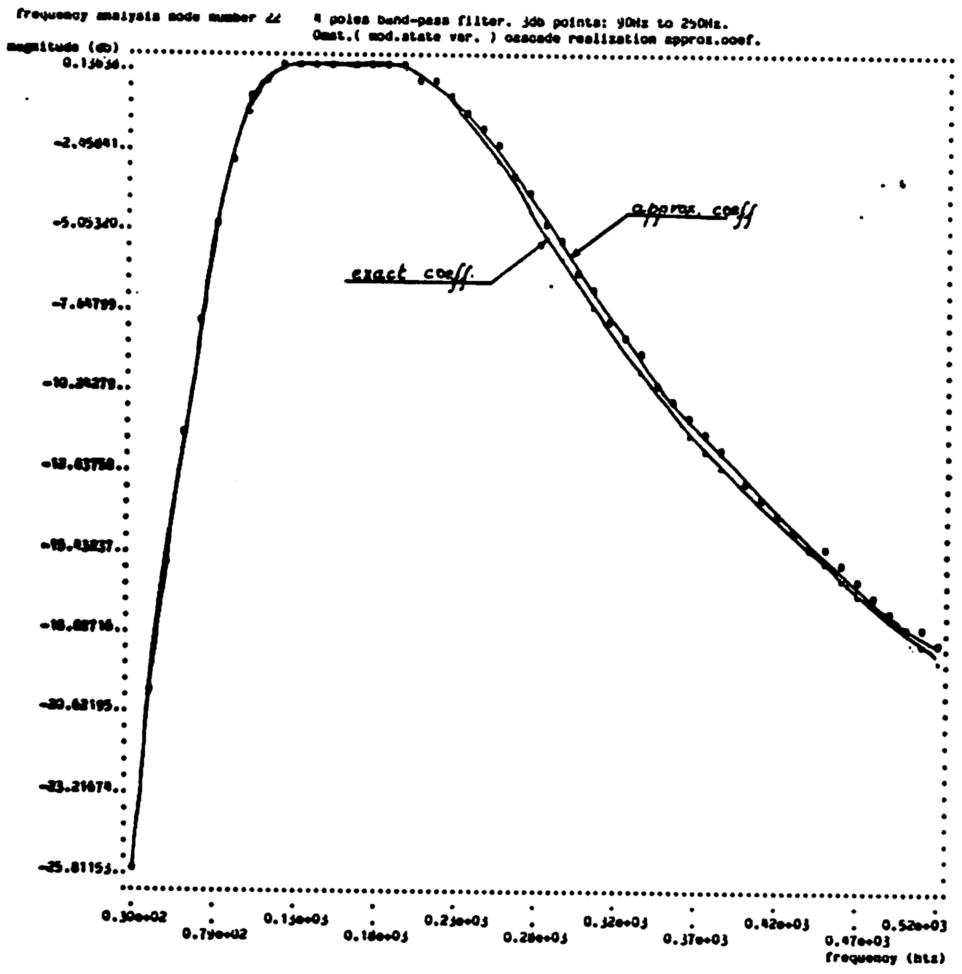


Fig. A1 First filter f_{-3db} = 90 Hz , 251 Hz

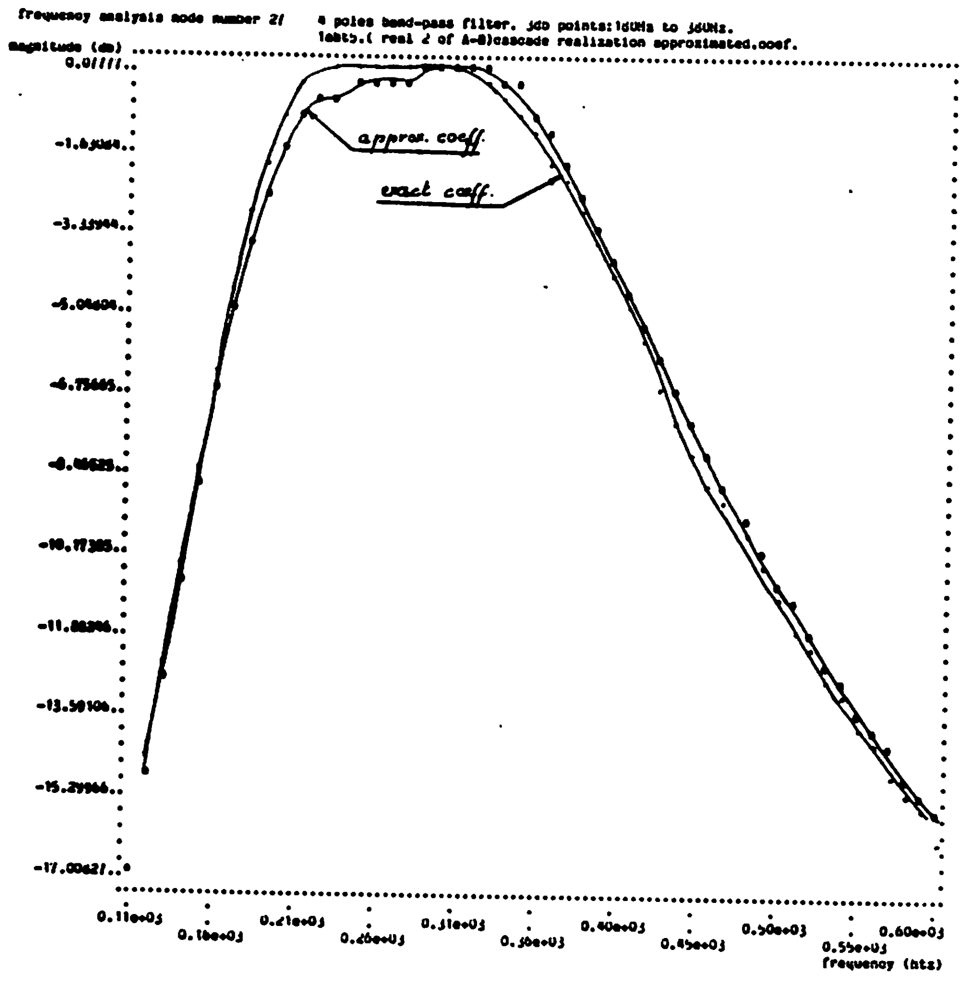


Fig. A2 Second filter f_{-3db} = 187 Hz , 385 Hz

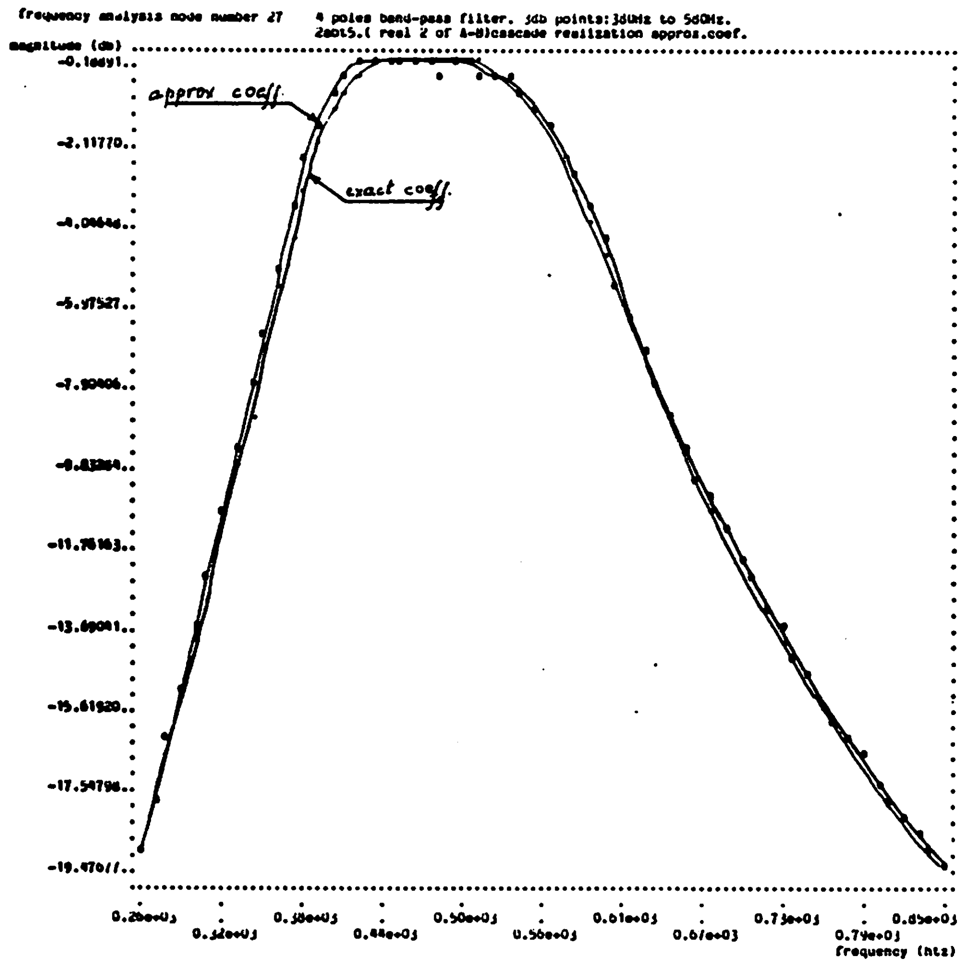


Fig. A3 Third filter f_{-3db} = 375 Hz , 582 Hz

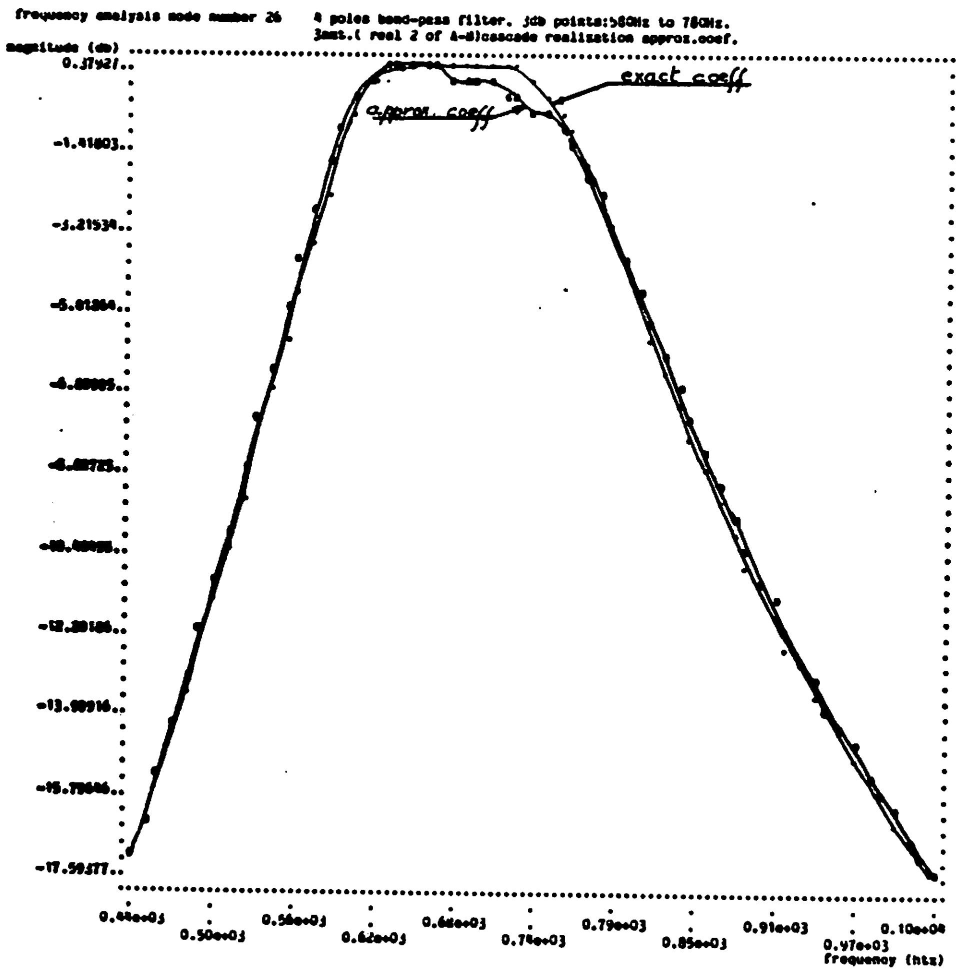


Fig. A4 Fourth filter f_{-3db} = 572 Hz , 780 Hz

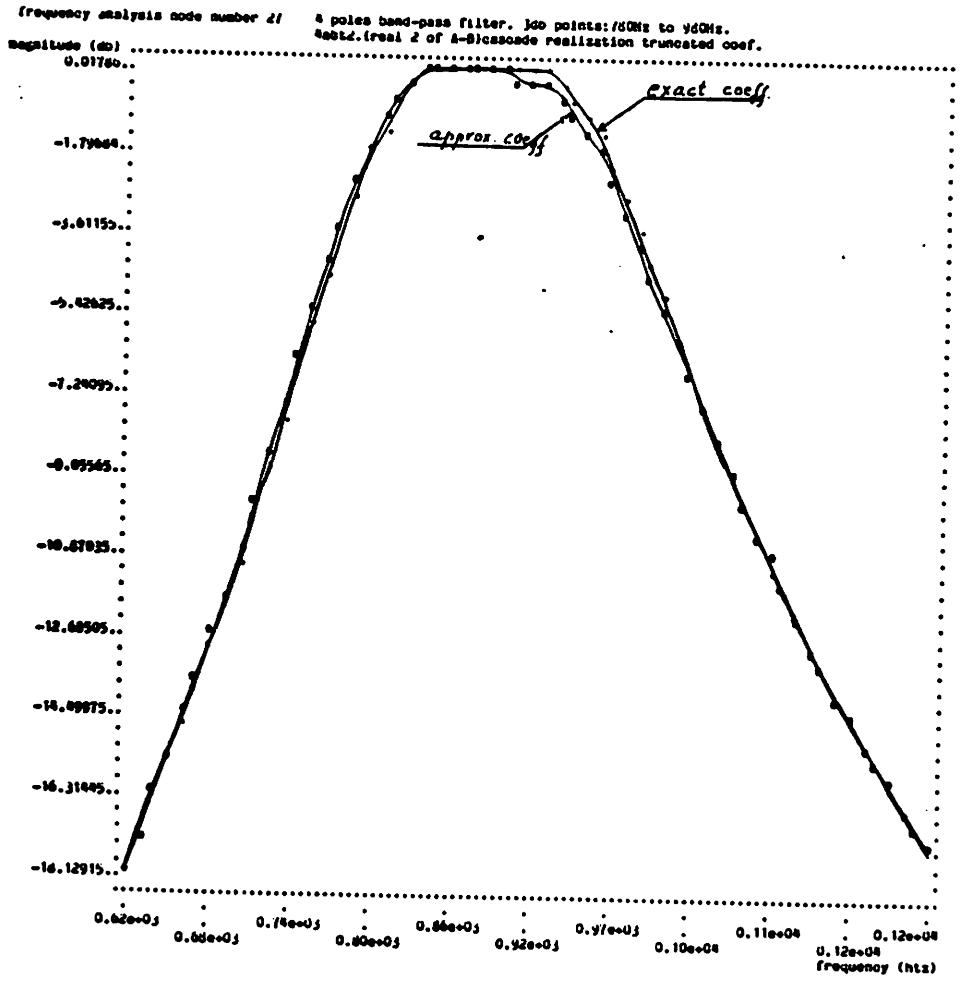


Fig. A5 Fifth filter f_{-3db} = 776 Hz , 978 Hz

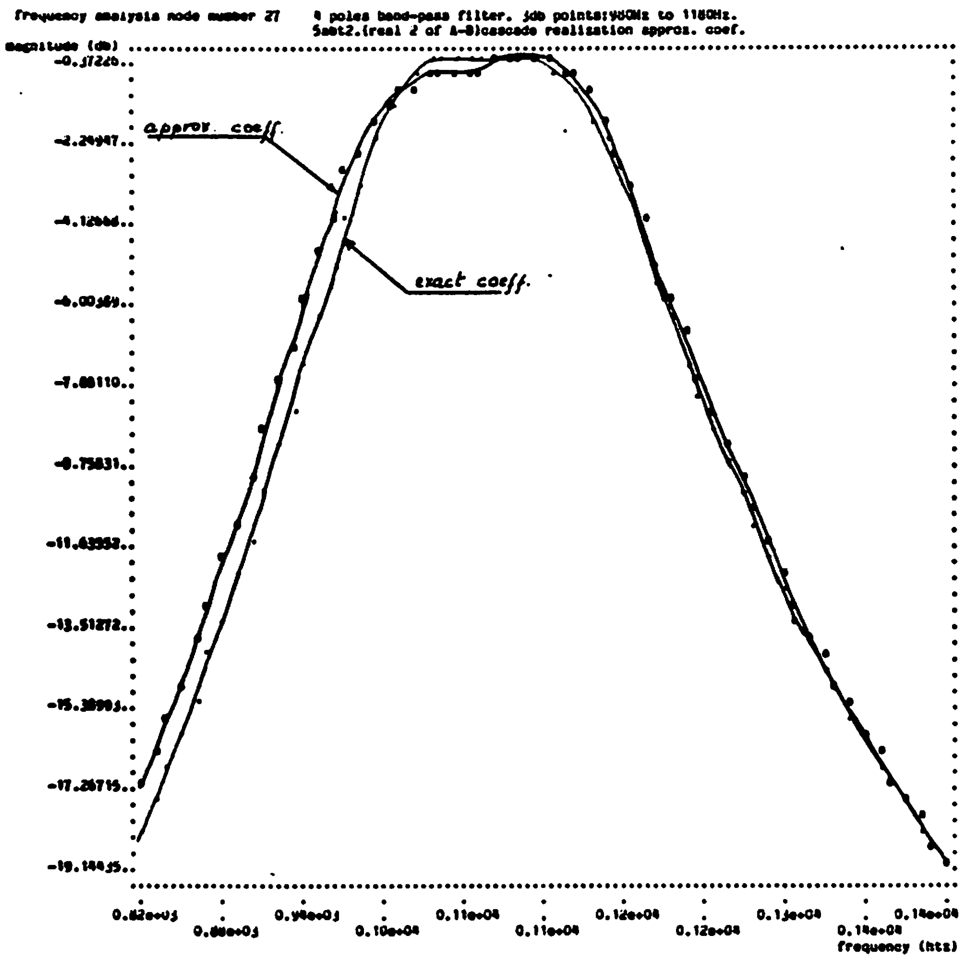


Fig. A6 Sixth filter f_{-3db} = 968 Hz , 1180 Hz

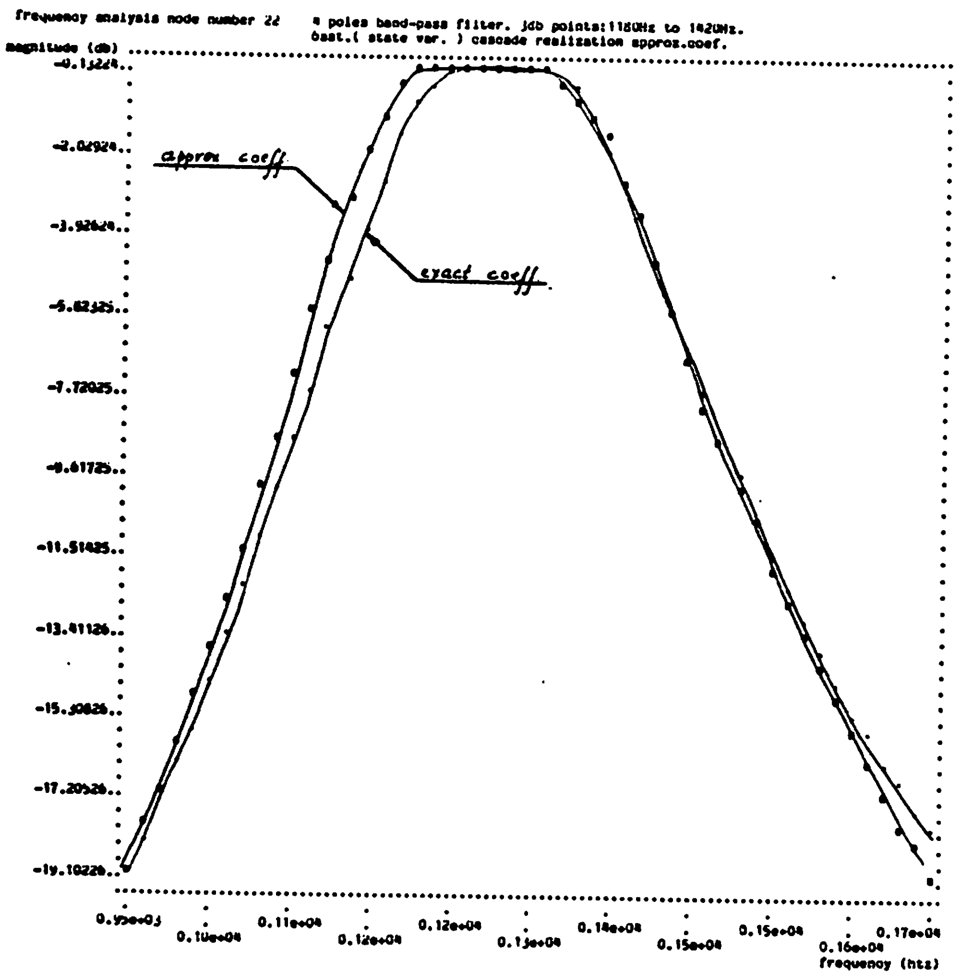


Fig. A7 Seventh filter f_{-3db} = 1147 Hz , 1408 Hz

Appendix B

Coefficient Reduction in Second Order Digital Filter Sections

1. Derivation of the Coefficients of the Second-Order Sections

We use the following procedure to obtain the exact coefficients of the two individual second order sections of the Butterworth bandpass filter

1- The bandpass filter is derived from the lowpass prototype.

The lowpass prototype for the fourth order Butterworth bandpass filter is the second order section given by:

$$H(s) = \frac{1}{(s + \frac{1+j}{\sqrt{2}})\,(s + \frac{1-j}{\sqrt{2}})} \tag{B1}$$

2- The bandpass filter is obtained from the lowpass prototype using the lowpass to bandpass transformation:

$$s = \frac{s^2 + \omega_0^2}{s\,\omega_b}$$

where:

$\omega_0 = \sqrt{\omega_u \omega_l}$ is the center frequency of the filter and

$\omega_b = \omega_u - \omega_l$ is the bandwidth of the filter.

ω_u and ω_l are the upper $-3db$ and lower $-3db$ frequencies respectively of the bandpass filter.

The transfer function of the bandpass filter is then:

$$H(s) = \frac{s\omega_b}{s^2 + s\omega_b(\frac{1+j}{\sqrt{2}}) + \omega_0^2} \cdot \frac{s\omega_b}{s^2 + s\omega_b(\frac{1-j}{\sqrt{2}}) + \omega_0^2} \tag{B2}$$

The roots of the first second order polynomial are:

$$s_{1,2} = -\frac{\omega_b(1+j)}{2\sqrt{2}} \pm j\omega_0\sqrt{1+\frac{j}{4Q^2}}$$

Using the usual definition of Q as:

$$Q = \frac{\omega_0}{\omega_b}$$

and defining:

$$K = \sqrt{1+\sqrt{1+\frac{1}{16Q^4}}}$$

then, after rearranging the real and imaginary parts and some manipulations, the roots can be written as:

$$s_1 = -\frac{\omega_b}{2\sqrt{2}}(1-\frac{1}{2QK}) + \frac{j\omega_0}{\sqrt{2}}(K-\frac{1}{2Q}) \tag{B3.a}$$

$$s_2 = -\frac{\omega_b}{2\sqrt{2}}(1+\frac{1}{2QK}) - \frac{j\omega_0}{\sqrt{2}}(K+\frac{1}{2Q}) \tag{B3.b}$$

The roots s_3 and s_4 of the other second-order polynomial in the denominator of (B2) are the complex conjugate of s_2 and s_1 respectively.

Associating s_1 and s_4 in a second order polynomial and s_2 and s_3 in another one, the transfer function of the filter can be written as:

$$H(s) = \frac{s\,\omega_b}{s^2 + \frac{\omega_b}{\sqrt{2}}(1 - \frac{1}{2QK})s + \left[\frac{\omega_b^2}{8}(1 - \frac{1}{2QK})^2 + \frac{\omega_0^2}{2}(K - \frac{1}{2Q})^2\right]} \cdot \frac{s\,\omega_b}{s^2 + \frac{\omega_b}{\sqrt{2}}(1 + \frac{1}{2QK})s + \left[\frac{\omega_b^2}{8}(1 + \frac{1}{2QK})^2 + \frac{\omega_0^2}{2}(K + \frac{1}{2Q})^2\right]} \qquad \text{(B4)}$$

In these equations, ω_0 and ω_b represent the center frequency and bandwidth respectively of the complete fourth-order bandpass filter.

3- Pre-warping the frequencies.

The frequencies used in the design of the analog filter must be pre-warped to compensate for the frequency warping effect due to the use of the bilinear transformation.

If we denote the prewarped frequencies by a tilde (~) above the symbol, the prewarped analog frequencies are:

$$\tilde{\omega}_0 = \frac{2}{T}\tan\frac{\omega_0 T}{2}$$

$$\tilde{\omega}_b = \frac{2}{T}(\tan\frac{\omega_u T}{2} - \tan\frac{\omega_l T}{2})$$

where T represents the period of the sampling frequency.

Due to these new frequencies, the Q and K will also take new values.

To facilitate the notation, we shall drop the tilde above all the symbols where they appear. It should be remembered however that from here on we are dealing with prewarped frequencies which appear also in the constants K and Q.

4- Transforming the analog filter to a digital one using the bilinear transformation:

$$s = \frac{2}{T}\frac{1-z^{-1}}{1+z^{-1}}$$

We denote:

$$A_1 = \frac{\omega_b}{\sqrt{2}}(1 - \frac{1}{2QK})$$

$$B_1 = \frac{\omega_b^2}{8}(1 - \frac{1}{2QK})^2 + \frac{\omega_0^2}{2}(K - \frac{1}{2Q})^2$$

$$A_2 = \frac{\omega_b}{\sqrt{2}}(1 + \frac{1}{2QK})$$

$$B_2 = \frac{\omega_b^2}{8}(1 + \frac{1}{2QK})^2 + \frac{\omega_0^2}{2}(K + \frac{1}{2Q})^2$$

Using this notation, (B4) is written as:

$$H(s) = \frac{s\omega_b}{s^2 + A_1 s + B_1} \cdot \frac{s\omega_b}{s^2 + A_2 s + B_2} \qquad \text{(B5)}$$

Using the bilinear transformation in (B5) and ordering the denominator in ascending powers of z^{-1}, the first second-order section of (B5) becomes:

$$H_1(z) = \frac{\frac{T}{2}\omega_b(1 - z^{-2})}{\left|1 + \frac{T}{2}A_1 + (\frac{T}{2})^2 B_1\right| + \left|-2 + 2(\frac{T}{2})^2 B_1\right| z^{-1} + \left|1 - \frac{T}{2}A_1 + (\frac{T}{2})^2 B_1\right| z^{-2}}$$

The second second-order section of (B5) is:

$$H_2(z) = \frac{\frac{T}{2}\omega_b(1 - z^{-2})}{\left|1 + \frac{T}{2}A_2 + (\frac{T}{2})^2 B_2\right| + \left|-2 + 2(\frac{T}{2})^2 B_2\right| z^{-1} + \left|1 - \frac{T}{2}A_2 + (\frac{T}{2})^2 B_2\right| z^{-2}}$$

The transfer function of the bandpass filter can be written as:

$$H(z) = \frac{G_1(1-z^{-2})}{1+a_{11}z^{-1}+a_{21}z^{-2}} \cdot \frac{G_2(1-z^{-2})}{1+a_{12}z^{-1}+a_{22}z^{-2}} \tag{B6}$$

where the coefficients G_1, a_{11}, a_{21}, G_2, a_{12}, a_{22} are given by:

$$G_1 = \frac{\frac{T}{2}\omega_b}{1+\frac{T}{2}A_1+(\frac{T}{2})^2B_1} \qquad G_2 = \frac{\frac{T}{2}\omega_b}{1+\frac{T}{2}A_2+(\frac{T}{2})^2B_2}$$

$$a_{11} = \frac{-2+2(\frac{T}{2})^2B_1}{1+\frac{T}{2}A_1+(\frac{T}{2})^2B_1} \qquad a_{12} = \frac{-2+2(\frac{T}{2})^2B_2}{1+\frac{T}{2}A_2+(\frac{T}{2})^2B_2}$$

$$a_{21} = \frac{1-\frac{T}{2}A_1+(\frac{T}{2})^2B_1}{1+\frac{T}{2}A_1+(\frac{T}{2})^2B_1} \qquad a_{22} = \frac{1-\frac{T}{2}A_2+(\frac{T}{2})^2B_2}{1+\frac{T}{2}A_2+(\frac{T}{2})^2B_2}$$

2. Finding the Positions of the Roots in the z Plane

The position of the roots of the denominator of (B6) in the z plane indicates which structure to use for realizing the second-order sections.[1]

The distance - r - from the origin to the root in the z plane and its angle - ϑ - can be obtained in terms of the coefficients just derived by noting that the denominator of a second order section can also be written as:

$$z^2 - z\,2r\cos\vartheta + r^2 \tag{B7}$$

By comparing with the denominator in (B6), we obtain the following relations:

$$r_1 = \sqrt{a_{21}} \tag{B8.a}$$

[1]Agarwal and Burrus, *New Recursive Digital Filter Structures Having Very Low Sensitivity and Roundoff Noise*, IEEE Trans.on Circuits and Systems,Vol. CAS-22, No.12,Dec. 1975,pp 921-927.

$$r_2 = \sqrt{a_{22}} \tag{B8.b}$$

$$\vartheta_1 = \cos^{-1}\left(-\frac{a_{11}}{2\sqrt{a_{21}}}\right) \qquad a_{11} \le 0 \tag{B8.c}$$

$$\vartheta_1 = \pi - \cos^{-1}\left(\frac{a_{11}}{2\sqrt{a_{21}}}\right) \qquad a_{11} \ge 0 \tag{B8.d}$$

$$\vartheta_2 = \cos^{-1}\left(-\frac{a_{12}}{2\sqrt{a_{22}}}\right) \qquad a_{12} \le 0 \tag{B8.e}$$

$$\vartheta_2 = \pi - \cos^{-1}\left(\frac{a_{12}}{2\sqrt{a_{22}}}\right) \qquad a_{12} \ge 0 \tag{B8.f}$$

3. Filter Realization Using Discrete Values Parameters.

When realizing digital filters only a finite number of bits are available to represent the coefficients. The discrete values thus obtained will differ from the desired ones that have an infinite precision. This will change the center frequency and the bandwidth of the individual filters and therefore also the composite filter.

The following is a procedure to obtain the discrete coefficients that realizes the desired frequency and bandwidth of the composite filter. In the process of obtaining these coefficients, we reduce the number of *1*'s in the binary representation of the coefficients and thus reduce the computations needed in the realization of the composite filter. The final value of the coefficient is a compromise between two conflicting requirements: exact realization of the filter specifications and minimization of the number of computations.

This procedure chooses coefficients having deviations that compensates one another so that the original -3db frequencies of the composite filter are retained.

The first step is to find how changes in the coefficients of the individual sections affect the center frequency and bandwidth of the composite filter.

1- Relations between the individual sections and the composite filter

The center frequency of the composite filter, denoted by ϑ_{oc}, is given by:

$$\vartheta_{oc} = \sqrt{\vartheta_{o1} \cdot \vartheta_{o2}} \qquad \text{(B9)}$$

where:

ϑ_{o1} and ϑ_{o2} are the center frequencies of the first and second section respectively.

The bandwidth of the composite filter can be found from the following relation which exist for maximally flat filters[2] - also called "staggered tuned filters" :

$$\frac{1}{Q_{oc}^2} = \frac{1}{Q_{o1}} \cdot \frac{1}{Q_{o2}} + \frac{(\vartheta_{o2} - \vartheta_{o1})^2}{\vartheta_{o1} \cdot \vartheta_{o2}}$$

where:

Q_{oc} , Q_{o1} and Q_{o2} are the Q's of the composite filter and of the first and second sections respectively.

The bandwidth of the composite filter denoted by B_c is then given by:

$$B_c^2 = \frac{\vartheta_{o1}}{Q_{o1}} \cdot \frac{\vartheta_{o2}}{Q_{o2}} + (\vartheta_{o2} - \vartheta_{o1})^2$$

which can be written as:

$$B_c = \left\{ (\vartheta_{u1} - \vartheta_{l1}) \cdot (\vartheta_{u1} - \vartheta_{l2}) + (\vartheta_{o2} - \vartheta_{o1})^2 \right\}^{\frac{1}{2}} \qquad \text{(B9.b)}$$

[2]Valley and Wallman, *Vacuum Tube Amplifiers*, M.I.T., Radiation Laboratory Series Vol. 18, McGRAW-HILL, 1948. Page 185.

where:

ϑ_{u1}, ϑ_{l1}, ϑ_{u2} and ϑ_{l2}, are the upper and lower -3db frequencies of the first and second sections respectively.

The first order deviations in the center frequency and the bandwidth of the composite filter are then:

$$\Delta \vartheta_{oc} = \sum_{i=1}^{4} \frac{\partial \vartheta_{oc}}{\partial x_i} \Delta x_i \tag{B10.a}$$

$$\Delta B_c = \sum_{i=1}^{4} \frac{\partial B_c}{\partial x_i} \Delta x_i \tag{B10.b}$$

where:

x_i i = 1 , 4 are the coefficients of the two second order sections.

To find the first order deviation in the composite filter it is necessary to express the center frequencies and the -3db frequencies of the individual sections in terms of their coefficients.

2- Center and - 3db frequencies of individual sections

For simplicity, we assume that the numerator of the second order sections remain constant for small changes in the frequency. The center frequency of the second order section is that one which minimizes the absolute value of the denominator of its transfer function.

The denominator of the second order section denoted by $D(\vartheta, r)$, is given by (B7) where $z = e^{j\vartheta}$

By taking the absolute value of $D(\vartheta, r)$ and finding the angle ϑ_o for which this absolute value is minimized we obtain the center frequency given by:

$$\cos \vartheta_o = \frac{1+r^2}{2r}\cos \vartheta$$

The center frequency of the first section is thus given by:

$$\cos \vartheta_{o1} = \frac{1+r_1^2}{2r_1}\cos \vartheta_1 \qquad \text{(B11.a)}$$

where ϑ_1 and r_1 are the coordinates of the pole with positive imaginary part in the z plane and are related to the coefficients of the filter as described in the previous section.

The center frequency of the second section is given by the same relation when changing r_1 to r_2 and $\cos \vartheta_1$ to $\cos \vartheta_2$.

The upper and lower -3db frequencies of the first second order section, denoted by ϑ_{u1} and ϑ_{l1} respectively are found from their definition. Thus:

$$\frac{\dfrac{1}{|D(e^{j\vartheta_{u1}})|^2}}{\dfrac{1}{|D(e^{j\vartheta_{o1}})|^2}} = \frac{1}{2} \qquad \text{(B12)}$$

where $D(e^{j\vartheta_{u1}})$ and $D(e^{j\vartheta_{o1}})$ are the values of the denominator at the upper -3db frequency and at the resonant frequency respectively.

$\left|D(e^{j\vartheta_{o1}})\right|$ is obtained by substituting the value of ϑ_{o1} in the denominator of the second order section and taking the absolute value.

The result is:

$$\left|D(e^{j\vartheta_{o1}})\right| = (1 - r_1^2)\sin \vartheta_1 \qquad \text{(B13)}$$

Substituting (B13) in (B12), we obtain:

$$\left|D(e^{j\vartheta_{u1}})\right|^2 = 2\,(\,1 - r_1^2\,)^2\,(\,1 - \cos^2\vartheta_1\,) \tag{B14}$$

Expanding $\left|D(e^{j\vartheta_{u1}})\right|^2$ we obtain an equation of the second degree in $\cos\vartheta_{u1}$ which yields the two -3db frequencies:

$$\cos\vartheta_{u1} = \frac{(\,1 + r_1^2\,)}{2r_1}\cos\vartheta_1 - \frac{(\,1 - r_1^2\,)}{2r_1}\sin\vartheta_1 \tag{B15.a}$$

$$\cos\vartheta_{l1} = \frac{(\,1 + r_1^2\,)}{2r_1}\cos\vartheta_1 + \frac{(\,1 - r_1^2)}{2r_1}\sin\vartheta_1 \tag{B15.b}$$

It is now possible to write the equations relating the center frequency and bandwidth of the composite filters and the coordinates of the poles of the second order sections in the z plane.

(B9.a) can be written as:

$$\vartheta_{oc} = \left\{(\,\cos^{-1}\frac{1 + r_1^2}{2r_1}\cos\vartheta_1\,)\,(\,\cos^{-1}\frac{1 + r_2^2}{2r_2}\cos\vartheta_2)\right\}^{\frac{1}{2}} \tag{B16.a}$$

(B9.b) is now:

$$B_c = \left\{\left[\cos^{-1}(\frac{1+r_1^2}{2r_1}\cos\vartheta_1 - \frac{1-r_1^2}{2r_1}\sin\vartheta_1) - \cos^{-1}(\frac{1+r_1^2}{2r_1}\cos\vartheta_1 + \frac{1-r_1^2}{2r_1}\sin\vartheta_1)\right]\cdot\right.$$

$$\cdot\left[\cos^{-1}(\frac{1+r_2^2}{2r_2}\cos\vartheta_2 - \frac{1-r_2^2}{2r_2}\sin\vartheta_2) - \cos^{-1}(\frac{1+r_2^2}{2r_2}\cos\vartheta_2 + \frac{1-r_2^2}{2r_2}\sin\vartheta_2)\right] +$$

$$\left. + \left[\cos^{-1}(\,\frac{1 + r_2^2}{2r_2}\cos\vartheta_2\,) - \cos^{-1}(\,\frac{1 + r_1^2}{2r_1}\cos\vartheta_1\,)\right]^2\right\}^{\frac{1}{2}} \tag{B16.b}$$

Using these equations and the relations between the coordinates of the poles and the coefficients of the filters, equations (B10) can be found.

3- First order deviations in center frequency and bandwidth.

The first order deviations in the center frequency and bandwidth depend on the configuration chosen for realizing the second order sections in (B6).
We choose the configuration which gives the lowest peak sensitivity of the denominator of the second order sections to the coefficients of the filters. This enables us to realize the second order sections with coarse parameters using the lowest possible number of bits in their binary values and still to obtain good agreement with the theoretical center frequency and bandwidth. As the sensitivity of the denominator depends on the position of the poles, the choice of the configuration is dictated by the coordinates of the poles of the second order sections in the z plane.[3]
The coefficients of the filters are realized differently in the different configurations. For most of the filters the coefficients in (B6) are realized in the following way:

$$a_{21} = 1 - b_1 \quad \text{(B17.a)}$$

$$a_{22} = 1 - b_2 \quad \text{(B17.b)}$$

$$a_{11} = -2 + a_1 \quad \text{(B17.c)}$$

$$a_{12} = -2 + a_2 \quad \text{(B17.d)}$$

where a_1, a_2, b_1, b_2, are now the coefficients which enter in the calculations of the deviations.
Using equations (B9) in (B10.a) and (B10.b) and using the new coefficients, the deviations in the center frequency and the bandwidth are:

$$\Delta \vartheta_{oc} = \frac{1}{2\sqrt{\vartheta_{o1}\vartheta_{o2}}} \left\{ \left[\frac{\partial \vartheta_{o1}}{\partial a_1} \Delta a_1 + \frac{\partial \vartheta_{o1}}{\partial b_1} \Delta b_1 \right] \vartheta_{o2} + \right.$$

[3]see 1

$$+ \vartheta_{o1}\left[\frac{\partial \vartheta_{o2}}{\partial a_2}\Delta a_2 + \frac{\partial \vartheta_{o2}}{\partial b_2}\Delta b_2\right]\Bigg\} \tag{B18.a}$$

$$\Delta B_c = \frac{1}{2B_c}\Bigg\{\left[\frac{\partial(\vartheta_{u1}-\vartheta_{l1})}{\partial a_1}\Delta a_1 + \frac{\partial(\vartheta_{u1}-\vartheta_{l1})}{\partial b_1}\Delta b_1\right](\vartheta_{u2}-\vartheta_{l2}) +$$

$$+ (\vartheta_{u1}-\vartheta_{l1})\left[\frac{\partial(\vartheta_{u2}-\vartheta_{l2})}{\partial a_2}\Delta a_2 + \frac{\partial(\vartheta_{u2}-\vartheta_{l2})}{\partial b_2}\Delta b_2\right] +$$

$$+2(\vartheta_{o2}-\vartheta_{o1})\left[-\frac{\partial \vartheta_{o1}}{\partial a_1}\Delta a_1 - \frac{\partial \vartheta_{o1}}{\partial b_1}\Delta b_1 + \frac{\partial \vartheta_{o2}}{d a_2}\Delta a_2 + \frac{\partial \vartheta_{o2}}{d b_2}\Delta b_2\right]\Bigg\}. \tag{B18.b}$$

By substituting equations (B17) into (B8) and using (B16) when taking the derivatives, we obtain the required relation.

Substitution of (B17) into (B8) yields:

$$r_1 = \sqrt{1-b_1} \tag{B19.a}$$

$$r_2 = \sqrt{1-b_2} \tag{B19.b}$$

$$\vartheta_1 = \cos^{-1}\frac{2-a_1}{2\sqrt{1-b_1}} \tag{B19.c}$$

$$\vartheta_2 = \cos^{-1}\frac{2-a_2}{2\sqrt{1-b_2}} \tag{B19.d}$$

From (B11) and (B15) and using (B19), we obtain the derivatives:

$$\frac{\partial \vartheta_{o1}}{\partial a_1} = \frac{1}{\sin\vartheta_{o1}}\cdot\frac{2-b_1}{4(1-b_1)} \tag{B20.a}$$

$$\frac{\partial \vartheta_{o1}}{\partial b_1} = -\frac{1}{\sin\vartheta_{o1}}\cdot\frac{2-a_1}{4(1-b_1)^2} \tag{B20.b}$$

$$\frac{\partial(\vartheta_{u1}-\vartheta_{l1})}{\partial a_1} = \frac{(2-b_1)}{4(1-b_1)}\left[\frac{1}{\sin\vartheta_{u1}} - \frac{1}{\sin\vartheta_{l1}}\right] + \frac{b_1\cos\vartheta_1}{4(1-b_1)\sin\vartheta_1}\left[\frac{1}{\sin\vartheta_{u1}} + \frac{1}{\sin\vartheta_{l1}}\right] \qquad \text{(B20.c)}$$

$$\frac{\partial(\vartheta_{u1}-\vartheta_{l1})}{\partial b_1} = \frac{1}{4(1-b_1)^{\frac{3}{2}}}\left\{2\cos\vartheta_1\left[\frac{1}{\sin\vartheta_{l1}} - \frac{1}{\sin\vartheta_{u1}}\right] + \left(2\sin\vartheta_1 - \frac{b_1}{\sin\vartheta_1}\right)\left[\frac{1}{\sin\vartheta_{u1}} + \frac{1}{\sin\vartheta_{l1}}\right]\right\} \qquad \text{(B20.d)}$$

The derivatives for the second section are the same equations when substituting the subscript 2 for the subscript 1.

For the other configurations used in the realization of the filters, equations (B19.a) and (B19.b) remain the same. Equations (B19.c) and (B19.d) are slightly different.In the "direct form" realization, when a > 1 they have the following representation:

$$\vartheta_1 = \cos^{-1}\frac{a_1}{2\sqrt{1-b_1}}$$

$$\vartheta_2 = \cos^{-1}\frac{a_2}{2\sqrt{1-b_2}}$$

for (B19.c) and (B19.d) respectively.

The derivatives in (B20) are changed accordingly. Equations (B16) and (B18) remain the same.

4- Obtaining the coefficients.

We want the actual filter to have the same -3db frequencies as the theoretical one or alternatively the same center frequency and bandwidth. That is:

$\Delta \vartheta_{oc} = \Delta B_c = 0$

As we have four unknowns which are the four deviations of the coefficients, and only two equations - for $\Delta\vartheta_{oc}$ and for ΔB_c - we choose two unknowns and calculate the other two so as to obtain the desired $\Delta\vartheta_{oc}$ and ΔB_c with the least total number of operations needed to realize the coefficients.

The number of operations needed in the realization of the filter depends on the method used to multiply the data by the coefficients. By using a parallel-serial multiplier and canonical coding, every *1* in the binary value of the coefficients requires one " add and shift " operation in the calculation of the multiplication. Therefore we are interested in reducing the number of *1*'s in the coefficients.

The results obtained are not the correct ones because the relations given by equations (B18) are not exact. The reason is that:

1- the first order difference holds well only for zero deviation of the center frequency and bandwidth. When minimizing the amount of computation by using coefficients with a low number of bits, we are sometimes far from zero deviation

2- the influence of the numerator of the filter does not appear in the relations. This influence is important for filters having poles close to the zeros of the numerator.

Therefore we choose the first two coefficients - one of each section - that are the closest to a combination of powers of 2, to obtain small deviations.Using these deviations from the theoretical coefficients, we obtain equations (B18) which have for unknowns the other two deviations. The deviations obtained by solving (B18) enable us to find the remaining two coefficients.

We would like to choose the first two coefficients with a minimum number of *1*'s. This is not always possible as the increased deviation in these coefficients increases the sensitivity to the other two. This in turn would require more precision when approximating these coefficients to obtain the desired center frequency and bandwidth.

The next step in this procedure is using DINAP- the digital filter simulation program - to simulate the digital filter using these four coefficients. The result of this simulation enables us to introduce a compensating factor $\Delta\vartheta_c$ and ΔB_c in (A18) to obtain the desired center frequency and bandwidth.

Using these corrected equations, the new deviations are now obtained. This yields the last two coefficients. These coefficients are approximated to the desired precision depending upon the requirements of the filters.

The filter realized using the obtained coefficients has -3db frequencies close to the desired ones with substantial savings in the computations needed. However its frequency response curve is distorted. Most of the time there is no great advantage of having a maximally flat filter. Its advantages are its simple mathematical expression and easily recognizable curve. Its disadvantages are that the phase shift it introduces is greater than the Bessel filters and its bandwidth is smaller than the Tchebycheff ones. For our application, the phase shift of the Butterworth filters seems to have little influence in the accuracy of the recognition algorithm. Therefore we expect that the added phase shift due to the distorted response has negligible influence.

5- Example

We want to realize a fourth-order Butterworth filter for which: $f_1 =$ 1172 Hz and $f_2 =$ 1402 Hz.

where:

f_1 and f_2 are the lower and upper -3db frequencies respectively.

The coefficients of the filter, obtained from (B6) are:

$a_1 = 0.341665$

$b_1 = 0.0665588$

$a_2 = 0.425042$

$b_2 = 0.0742053$

A simple way to obtain discrete coefficients from the theoretical ones is to approximate each one to the nearest power of two. In this case, we would need three 1's for each of the a_1 and a_2 and two for each of b_1 and b_2 or a total of 10 to obtain the desired specifications of the filter.

To minimize the number of 1's it is possible to try different coefficients and to test the results using DINAP. The disadvantage of this method is that there is no indication how to choose the coefficients so that the center frequency and the bandwidth will be close to the desired ones. This is important in the realization of a bank of filters. Without being able to control the location of the center frequency and of the bandwidth (or of the two - 3db frequencies) a gap or an overlap might appear between two filters. Also this method does not ensure the minimum number of bits for the coefficients and hence the minimum number of operations in the realization of the digital filter.

The first step to obtain the desired coefficients is to calculate the coefficients appearing in equations (B18).

From the given coefficients of the filters we obtain:

$\cos\vartheta_1 = 0.8582208$ $\quad\quad$ $\cos\vartheta_2 = 0.8184302$

$\sin\vartheta_1 = 0.5132806$ $\quad\quad$ $\sin\vartheta_2 = 0.5746060$

$\vartheta_{o1} = 0.5380108\ rad$	$\vartheta_{o2} = 0.6110633\ rad$
$\sin\vartheta_{o1} = 0.5124286$	$\sin\vartheta_{o2} = 0.5737387$
$\vartheta_{u1} = 0.5715757\ rad$	$\vartheta_{u2} = 0.6486814\ rad$
$\sin\vartheta_{u1} = 0.5409580$	$\sin\vartheta_{u2} = 0.6041362$
$\vartheta_{l1} = 0.5024403\ rad$	$\vartheta_{l2} = 0.5713056\ rad$
$\sin\vartheta_{l1} = 0.4815657$	$\sin\vartheta_{l2} = 0.5407307$

Using these values we obtain the following equations for the first order deviations of the bandwidth and the center frequency:

$$-0.147727\ \Delta a_1 + 0.219570\ \Delta b_1 + 0.132367\ \Delta a_2 - 0.0414564\ \Delta b_2 = 0 \qquad \text{(B21.a)}$$

$$0.617500\ \Delta a_1 - 0.567403\ \Delta b_1 + 0.487654\ \Delta a_2 - 0.430781\ \Delta b_2 = 0 \qquad \text{(B21.b)}$$

where we have assumed that $\Delta\vartheta_c = \Delta B_c = 0$.

We choose $b_1 = 0.0625$ and $a_2 = 0.04375$ as these values are the closest to a combination of powers of 2. From these values we obtain the deviations Δb_1 and Δa_2 and insert them into equations (B21). From the resulting equations, we obtain Δa_1 and Δb_2. Adding these deviations to the theoretical values a_1 and b_2 we obtain a new set of coefficients.

The relations (B21) are only approximate. (B21.a) relating the deviation in the bandwidth of the composite filter is a worse approximation than (B21.b). Using DINAP with these coefficients gives us a deviation from the expected bandwidth. To correct (B21.a) we add the opposite value $-2B_c\Delta B_c$ to the right side of the equation.

Repeating the process of computation of Δa_1, Δb_2, a_1, b_2 and using DINAP, we compensate alternatively the bandwidth and the frequency deviation. Using the compensated equations, we find the last two coefficients. The discrete coefficients are then approximated to the ideal coefficients. The accuracy of

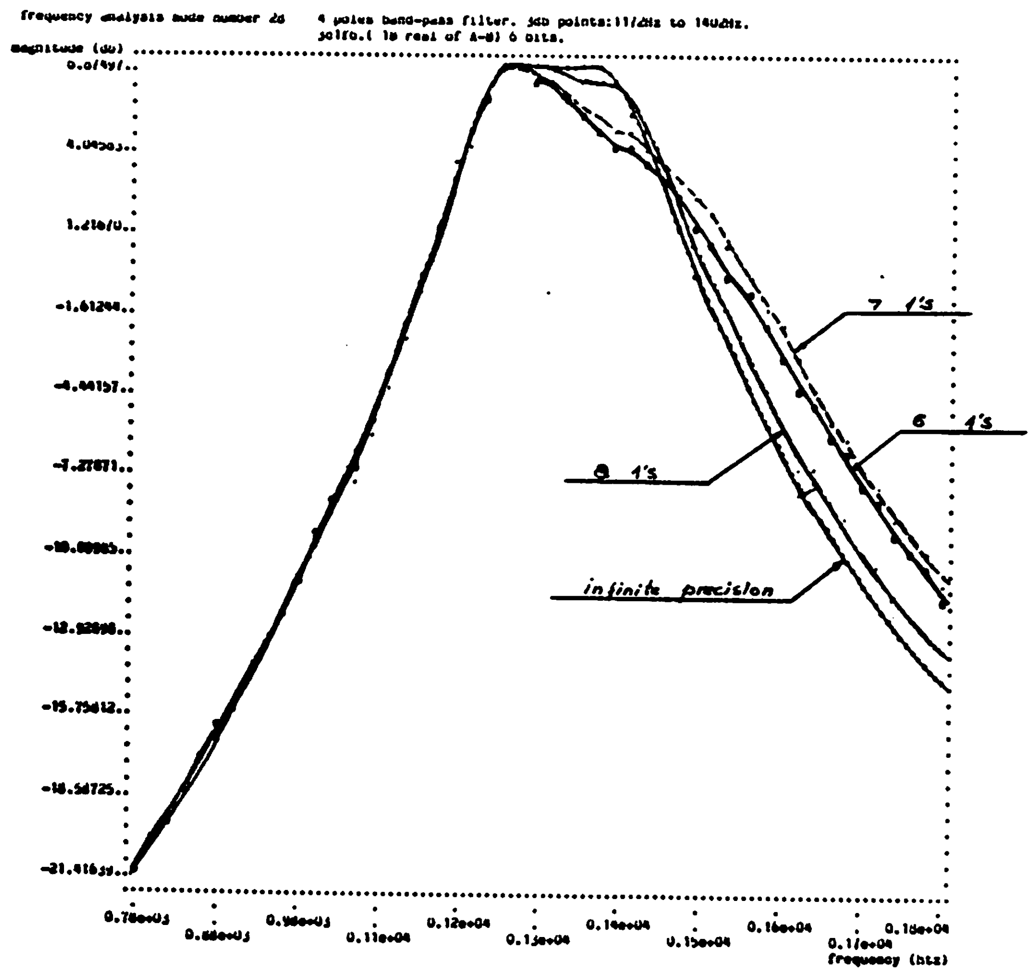


Fig. B.1 Frequency response for different coefficients

this approximation depends on the amount of deviation from center frequency and bandwidth that can be tolerated.

6- Results:

Figure B.1 shows the frequency response of the composite filter for various coefficients. For the first two coefficients chosen as stated earlier, (a total of three *1*'s), we need another five *1*'s to obtain reasonable good agreement with the theoretical frequency response.

When choosing $a_2 = 0.5$ instead of $a_2 = 0.5 - 0.0625$ the high frequency side of the frequency response curve is degraded. The number of *1*'s however can be reduced to 7 and even to 6 depending on the deviation tolerated in the bandwidth of the composite filter.

The number of "add and shift" operations has been reduced from 10 to 6 for this particular filter. The new transfer function is shown in Fig. B.2 along with the original one. In the realization of 16 filters, a substantial reduction in the number of operations is obtained when using this procedure.

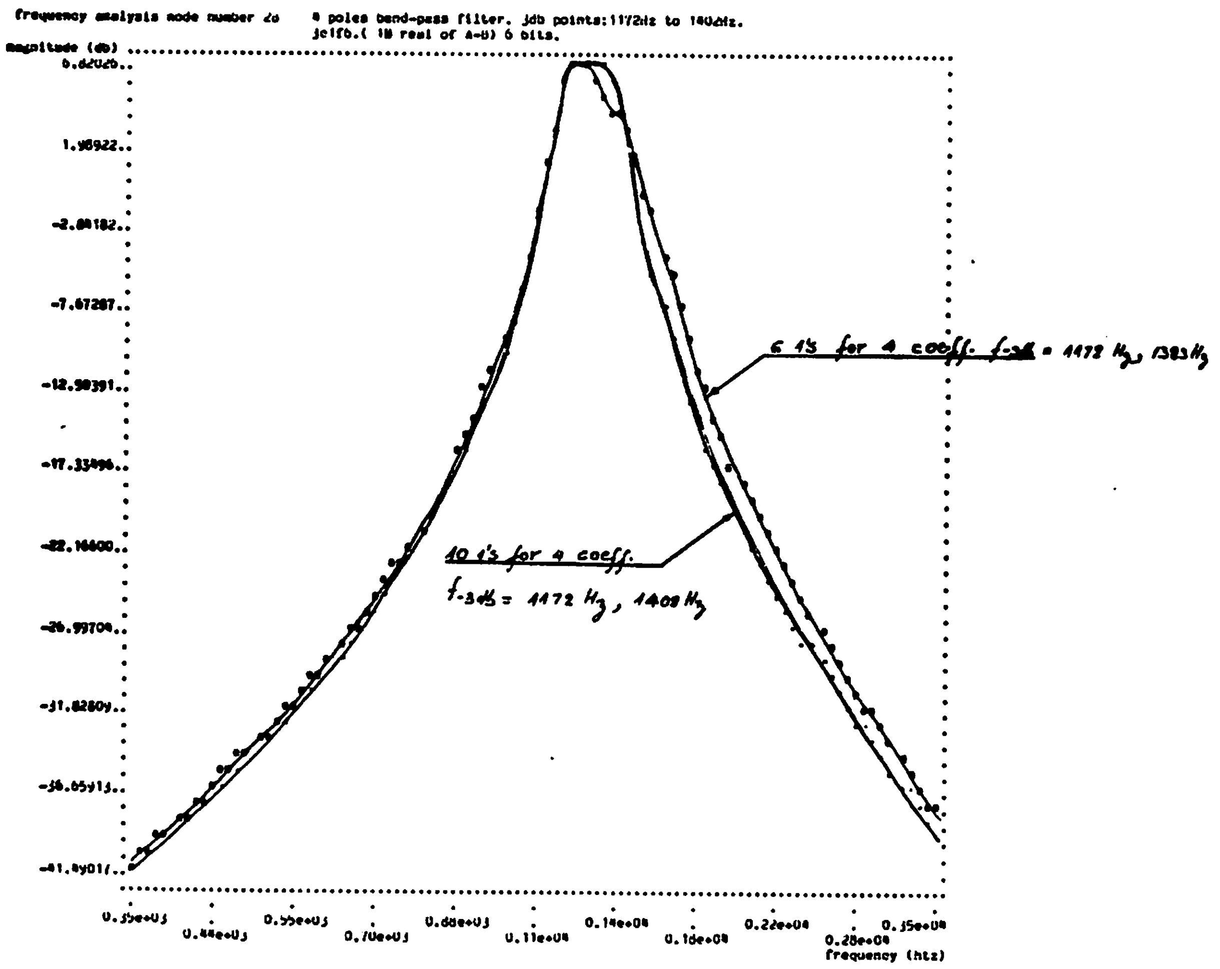


Fig. B.2 Frequency response of the filter with 10 '1's and 6 '1's

REFERENCES

[1] N.Rex Dixon and Thomas B. Martin Eds.,"Automatic Speech and Speaker Recognition," IEEE Press, New York, 1979 .

[2] H.Sakoe and S.Chiba, "A Dynamic Programming Approach to Continuous Speech Recognition," *Proc. 7th Int. Congr.Acoustics*, 1971, Paper 20, p.C13.

[3] G.M. White and R.B. Neely,"Speech Recognition Experiments with Linear Prediction, Bandpass Filtering and Dynamic Programming," *IEEE Trans. Acoust.,Speech and Signal Processing*,Vol.ASSP-24 (2),April 1976.

[4] R.W. Brodersen, "Signal Processing Using MOS-VLSI Technology" from "VLSI Electronics: Microstructure Science", Vol. 2,pp.189-227 Academic Press Inc.,1981.

[5] D.R Reddy and A.Newell,"Knowledge and its representation in a speech understanding system," in *Knowledge and Cognition*,L.W. Gregg, Ed. Washington,DC.1974,pp. 253-285.

[6] D.B. Fry,"*The Physics of Speech*",Cambridge University Press, 1979.

[7] Wayne A. Lee, "What Causes Speech Recognizers to Make Mistakes ?", *IEEE Transactions*,pp. 2030-2033, 1982.

[8] A. Newell, "A tutorial on speech understanding systems," in *Speech Recognition:Invited Papers of the IEEE Symp.*,D.R.Reddy ed. New York: Academic Press, 1975.

[9] S.R.Hyde, "Automatic Speech Recognition," from *Human Communication: A Unified View*, E.E. David, Jr. and P.B. Denes, Eds., pp. 399-438, McGraw Hill, New York, 1972.

[10] D. Raj Reddy, "Speech Recognition by Machine: A Review," *Proc. IEEE*,vol. 64,pp. 501-531, Apr. 1976.

[11] R.K. Potter, G.A. Kopp, and H.C. Green, "Visible Speech," Van Nostrand, New York, 1947.

[12] K.H.Davis,R.Biddulph and S.Balashek, Automatic Recognition of Spoken Digits,*J. Acoust. Soc. Am.*,**24**(6):637-642 (1952).

[13] J. Wiren and H.L. Stubs, Electronic Binary Selection System for Phoneme Classification,*J..Acoust.Soc.Am.*,vol. 28,pp.1082-1091 (1956).

[14] J.W. Forgie and C.D. Forgie, Results Obtained from a Vowel Recognition Computer Program,*J. Acoust. Soc. Am.*, vol. 31,pp.1480-1489 (1959).

[15] J. Suzuki and K. Nakata, Recognition of Japanese Vowels-Preliminary to the Recognition of Speech, *J. Radio Res. Lab. Tokyo*, vol. 8 (37),pp.193-212 (1961).

[16] T. Sakai and S. Doshita, The Phonetic Typewriter, Information Processing 1962,*Proc. IFIP Congr.*, Munich,August–September 1962.

[17] L.C. Pols, "Real-time recognition of spoken words,"*IEEE Trans.Comput.*, vol.C-20,pp.972-978,Sept.1971.

[18] L.C.Pols,H.R.C. Tromp and R. Plomp," Frequency Analysis of Dutch Vowels from 50 Male Speakers,"*J.Acoustical Society of America*, Vol. 53 (4),pp.1093- 1101,1973.

[19] Dennis H. Klatt, "Review of the ARPA Speech Understanding Project",*J. Acoust.Soc.Am.*, vol. 62,pp.1345-1366,Dec. 1977.

[20] J.K. Baker, "The DRAGON system-An overview,"*Ieee Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, pp. 24-29, Feb. 1975.

[21] G.R. Doddington and T.B. Schalk, "Speech recognition:turning theory to practice",*IEEE Spectrum*,pp.26-32,September 1981.

[22] Frederick Jelinek,"Self-Organized Continuous Speech Recognition", IBM T.J. Watson Research Center, November 1982.

[23] E.P. Neuburg, "Philosophies of Speech Recognition,"in *Speech Recognition* (D.Raj Reddy,ed.), Academic Press,New York,1975.

[24] Hisao Ishizuka et al., "A Microprocessor For Speech Recognition",pp.503-506, ICASSP 83, Boston.

[25] L. Lewis, Z. Amitai and H.F. Silverman, "The APS-II Processor for Speech Recognition",pp.483-485,ICASSP 83, Boston.

[26] R.W. Schafer and L.R. Rabiner, "Digital Representation of Speech Signals," *Proc. of the IEEE*, Vol. 63,No. 4, pp. 662-677,April 1975.

[27] L.T.Linlet al., "A Monolithic Audio Spectrum Analyzer for Speech Recognition System,"*IEEE INt. Solid-State Circuits Conference*, FAM 19.5,1982.

[28] L.R. Rabiner and R.W. Schafer, "*Digital Processing of Speech Signals*", Prentice-Hall,Inc.,1978.

[29] M. R. Schroeder, "Models of Hearing,"*IEEE Proc.*,Vol.63,No.9, September 1975.

[30] J.L Flanagan and S.W. Christensen, "Computer Studies on Parametric Coding of Speech Spectra", *J.Acoust.Soc.Am.* vol. 68(2),pp. 420-430, Aug. 1980.

[31] B. Scharf,"*Critical Bands*," in *Foundations of Modern Auditory Theory*, edited by J.V. Tobias (Academic,New York), Vol. 1,Chap. 5.

[32] L.L. Beranek,"The design of speech communication systems,"*Proc. IRE Vol.* 35,pp. 880-890,April 1947.

[33] H. Murveit,"Influence of the Selectivity of Bandpass filters on Recognition Accuracy," unpublished report.

[34] H. F. Silverman and N.R.Dixon, "A Parametrically Controlled Spectral Analysis System for Speech", *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. ASSP-22, no.5,pp. 362-381,Oct. 1974.

[35] L.R.Rabiner and M.R.Sambur, "An Algorithm for Determining the Endoints of Isolated Utterances," *Bell System Technical Journal*, Vol.54, no. 2, pp. 297-315, Feb 1975.

[36] Eric Davies,"Endpoint Detection of Speech for Real Time Isolated-Word Recognition,"*M.S. Thesis*,University of California, Berkeley,June 1983.

[37] L. F. Lamel, "Methods of endpoint detection for isolated word recognition",*M.S. thesis*, Massachusetts Inst. Tech., Feb 1980.

[38] Hy Murveit, "A Custom I.C. Based Speech Recognition System ", Ph.D. Thesis U.C.Berkeley, July 1983.

[39] R.Bellman and S.Dreyfus, *Applied Dynamic Programing*, New Jersy, Princeton University Press, 1962.

[40] A.H.Gray,Jr. and J.D. Markel, "Distance measures for speech processing," *IEEE Trans. Acoust.,Speech, Signal Processing*, vol. ASSP-24,pp. 380-391, Oct. 1976.

[41] Cory S. Myers, "A Comparative Study of Several Dynamic Time Warping Algorithms for Speech Recognition". *M.S. thesis*, M.I.T. Feb. 1980.

[42] L.R.Rabiner,A.E.Rosenberg and S.E.Levinson, "Considerations in dynamic time warping algorithms for isolated word recognition," *IEEE Trans. Acoust.,Speech,Signal Processing*,vol. ASSP-26,pp. 575-582,Dec.1978.

[43] John S. Bridle, Michael D. Brown, and Richard M. Chamberlain,"An Algorithm for Connected Word Recognition,"*Proc. Int. Conf. Acoustics, Speech and Signal Processing*,vol.2,pp.899-902, May 1982.

[44] Robert Kavaler,"*Applications of Clustering to Speech Recognition/fP ,University of California, Berkeley (June 1983), M.S. Thesis.*

[45] Kenneth L. Carlock and Menahem Lowy, *Speech Processor, A 19 Channel Vocoder*,, Electronics Research Laboratory, U.C. Berkeley (Oct. 1979), Technioal Memorandum.

[46] D.B.Cox and L.T.Lin, "A Realtime Switched Capacitor Filter", *ISSCC Proc.* ,WPM 8.6,pp.94-95,Feb. 1980.

[47] Siliconix incorporated ,"DG 506, 16-Channel CMOS Analog Multiplexer",*analog switch and IC product data book*, January 1982.

[48] Siliconix incorporated , "DF 331, CMOS $ mu $-255 law companding converter set",*Telecommunications Data Book*, July 1978.

[49] B. Fotouhi and D.A. Hodges, "High-Resolution A/D Conversion in MOS/LSI," *IEEE J. Solid-State Circuits*, Vol. SC-14,No.6,pp. 920-926, Dec. 1979.

[50] A.Peled and B.Liu, "A New Hardware Realization of Digital Filters,"*IEEE Trans.Acoust.,Speech, Signal Process.*,vol.ASSP-22,pp.456-462,December 1974.

[51] S.P. Pope and R.W. Brodersen, "Macrocell Design for Concurrent Signal Processing",Third Caltech Conference on Very Large Scale Integration, Pasadena,March 1983.

[52] Hwang, K., "Computer Arithmetic", Wiley, New York, 1979, p. 149.

[53] P.Ruetz,S. Pope and R.W.Brodersen, "A Front End Filter Chip for Speech Recognition System" to be published.

[54] B. Gold and C.M. Rader,"Digital Processing of Signals," McGraw-Hill, 1969, p.129.

[55] Lawrence R. Rabiner and Bernard Gold,"Theory and Application of Digital Signal Processing," Prentice-Hall,Inc., 1975, p.341.

[56] A.H. Gray,Jr., and John D. Markel,"Digital Lattice and Ladder Filter Synthesis,"*IEEE Trans.Audio and Electroacoustics*, vol.AU-21.NO.6,December 1973, pp.491-500.

[57] W.H. Ku, and S.M. Ng,"Floating-Point Sensitivity and Roundoff Noise of Recursive Digital Filters Realized in Ladder Structures,*IEEE*

Trans. Circuits and Systems,vol.CAS-22, No. 12,pp. 927-936,December 1975.

[58] Ernst Avenhaus,"A Proposal to Find Suitable Canonical Structures for the Implementation of Digital Filters with small Coefficient Wordlength", *Nachrichtenctech. Z.*,vol.8, pp.377-382,August 1972,.

[59] Ramesh C. Agarwal and C.Sidney Burrus,"New Recursive Digital Filter Strucutrures Having Very Low Sensitivity and Roundoff Noise",*IEEE Trans. Circuits and Systems*,vol.CAS-22,No. 12, pp.921-927,December 1975.

[60] Bedrich J. Hosticka, "MOS Sampled Data Recursion Filters Using State Variable Techniques," Memorandum No. ERL-M77/65, University of California, Berkeley,August 1977.

[61] Alan K. Yuen,"Design of a Digital Spectrum Analyser,"M.S. Thesis University of California, Berkeley, June 1982.

[62] R.Bolton et al.,"Computer-Aided Design of Recursive Digital Filters with Coefficients Having Restricted Minimal Representation",*IEEE Trans.Acoust.,Speech,Signal Process.*,vol. ASSP-29,pp.1205-1208,December 1981.

[63] DINAP, digital network analysis program, School of Electrical Engineering , circuits and systems group ,Purdue Univerity.

[64] Hiroaki Sakoe and Seibi Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition,"*IEEE Trans. Acoustics, Speech and Signal Procesing*,Vol. ASSP-26,pp.43-49 (Feb. 1978).

[65] D.J Burr,Bryan Ackland, and Neil Weste, "A High Speed Array Computer for Dynamic Time Warping,"*Proceedings IEEE International Conference on Acoustics, Speech and Signal Processing*, pp.471-473 (Spring 1981).

[66] David M. Mintz, "An Implementation of a Speech Recognition System", M.S Thesis,U.C.B. July 1983.

[67] M. Watari,M. Akabene and Y. Sako, "A Speaker Independent Word Recognition Based on Transient Matching",*IEEE International Conference on Acoustics, Speech and Signal Processing*,pp.715-718,ICASSP 83,Boston.

[68] S. Suzuki et al., " Transients for Speaker Independent Recognition System", pp. 1203-1207,ICASSP 83, Boston.