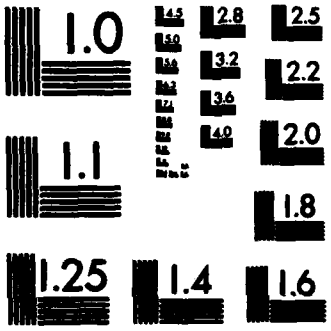
AD-A135 232 REAL-TIME IMPLEMENTATION OF A SPEECH DIGITIZATION ALGORITHM COMBINING TIM..(U) NOTRE DAME UNIV IN DEPT OF ELECTRICAL ENGINEERING J L MELSA ET AL. JUN 83 1/2

UNCLASSIFIED DCA100-82-C-0026 F/G 9/4 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(13)

AD-A135 382

FINAL REPORT - VOLUME 1

REAL-TIME IMPLEMENTATION OF A SPEECH DIGITIZATION ALGORITHM COMBINING TIME-DOMAIN HARMONIC SCALING AND ADAPTIVE RESIDUAL CODING

DTIC
DEC 2 1983
A

Department of
ELECTRICAL ENGINEERING
UNIVERSITY OF NOTRE DAME, NOTRE DAME, INDIANA

DTIC FILE COPY

This document has been approved for public release and sale; its distribution is unlimited.

83 12 [illegible] 028

FINAL REPORT - VOLUME 1

REAL-TIME IMPLEMENTATION OF A SPEECH DIGITIZATION ALGORITHM COMBINING TIME-DOMAIN HARMONIC SCALING AND ADAPTIVE RESIDUAL CODING

This document has been approved for public release and sale; its distribution is unlimited.

Prepared for

Defense Communications Agency
Defense Communications Engineering Center
1860 Wichle Avenue
Reston, Virginia 22090

Contract No. DCA-100-82-C-0026

1 June 1983

DTIC
ELECTE
DEC 2 1983
S D
A

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) REAL-TIME IMPLEMENTATION OF A SPEECH DIGITIZATION ALGORITHM COMBINING TIME-DOMAIN HARMONIC SCALING AND ADAPTIVE RESIDUAL CODING		5. TYPE OF REPORT & PERIOD COVERED Final Feb. 1982 to May 1983 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) James L. Melsa James D. Mills Arvind A. Arora		8. CONTRACT OR GRANT NUMBER(s) DCA-100-82-C-0026
9. PERFORMING ORGANIZATION NAME AND ADDRESS Department of Electrical Engineering University of Notre Dame Notre Dame, IN 46556		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Defense Communications Agency Contract Management Division, Code 260 Washington, DC 20305		12. REPORT DATE June 1983 13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Distribution of this document is unlimited. If may be released to the Clearinghouse, Department of Commerce, for sale to the public.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Speech coding, 16000 bps speech transmission, time-domain harmonic scaling, adaptive residual coding, hardware realization.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the results of a fifteen-month study, supported by DCA Contract 100-82-C-0026, of the real-time implementation of algorithm combining time-domain harmonic scaling (TDHS) and Adaptive Residual Coding (ARC) at a transmission bit rate of 16 kb/s. The modifications of this encoding algorithm as originally presented by Melsa and Pande (1981) to allow real-time implementation are described in detail. A non real-time FORTRAN simulation

(continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

using a sixteen-bit word length was developed and tested to establish feasibility. The hardware implementation of a full-duplex, real-time system has demonstrated that this algorithm is capable of producing toll quality speech digitization.

This report has been divided into two volumes. The first volume discusses the algorithm modifications and FORTRAN simulation. The details of the hardware implementation, schematics for the system and operating instructions are included in Volume 2 of this final report.

ABSTRACT

This report describes the results of a fifteen-month study, supported by DCA Contract 100-82-C-0026, of the real-time implementation of algorithm combining time-domain harmonic scaling (TDHS) and Adaptive Residual Coding (ARC) at a transmission bit rate of 16 kb/s. The modifications of this encoding algorithm as originally presented by Melsa and Pande (1981) to allow real-time implementation are described in detail. A non real-time FORTRAN simulation using a sixteen-bit word length was developed and tested to establish feasibility. The hardware implementation of a full-duplex, real-time system has demonstrated that this algorithm is capable of producing toll quality speech digitization.

This report has been divided into two volumes. The first volume discusses the algorithm modifications and FORTRAN simulation. The details of the hardware implementation, schematics for the system and operating instructions are included in Volume 2 of this final report.

Accession For
NTIS GRA&I ☑
DTIC TAB ☐
Unannounced ☐
Justification
By
Distribution/
Availability Codes
Dist | Avail and/or Special
A-1

PROJECT PERSONNEL

James L. Melsa, principal investigator

James D. Mills, research assistant

TABLE OF CONTENTS

		Page
ABSTRACT		ii
PROJECT PERSONNEL		iii
LIST OF FIGURES		vi
LIST OF ABBREVIATIONS		vii
CHAPTER I:	INTRODUCTION	1
CHAPTER II:	ORIGINAL SYSTEM	3
	2.1 Introduction	3
	2.2 Transmitter	6
	2.3 Channel	7
	2.4 Receiver	8
CHAPTER III:	FINAL SYSTEM	10
	3.1 Introduction	10
	3.2 Transmitter	10
	3.3 Channel	13
	3.4 Receiver	13
CHAPTER IV:	INITIAL ALGORITHM REFINEMENTS	15
	4.1 Introduction	15
	4.2 16 kb/s	15
	4.3 Bit error rate	24
CHAPTER V:	HARDWARE INDUCED MODIFICATIONS	28
	5.1 Introduction	28
	5.2 Serial Pitch Extraction	28
	5.3 8 kHz Sampling Rate	36
	5.4 Hardware Compatible ARC	37
	5.5 Finite Word Length	39
CHAPTER VI:	METHODOLOGY	41
	6.1 Introduction	41
	6.2 Files and Options	41
CHAPTER VII:	CONCLUSIONS	49
APPENDIX A:	FINITE WORD LENGTH SIMULATION SOURCE FILES	51

		Page
APPENDIX B:	UNRESTRICTED WORD LENGTH SOURCE FILES	111
APPENDIX C:	SUPPORTING PROGRAMS	160
APPENDIX D:	FILTER DESIGN	171
APPENDIX E:	FINAL OUTPUT EXAMPLES	172
REFERENCES:	.....	180

LIST OF FIGURES

		Page
Figure 2.1.1	Original TDHS-ARC System	4
Figure 2.1.2	Original System Basic Structure	5
Figure 3.1.1	Final TDHS-ARC System	11
Figure 4.1.1	First Stage of Algorithm Refinements	16
Figure 4.2.1	Determination of Scaling Factors and Thresholds	18
Figure 4.2.2	Hand Picked Scaling Factors and Thresholds	22
Figure 4.2.3	Logarithmically Distributed Scaling Factors and Thresholds	23
Figure 4.2.4	Quantizer Thresholds Scaling Factors, and Expansion Factors for Speech at 6400 Hz and 8000 Hz Sampling	25
Figure 4.3.1	Channel with Temporary BER Assumption	27
Figure 5.2.1	TDHS Algorithms	31
Figure 5.2.2	Center Clipping Algorithms	33
Figure 6.1.1	Final System Algorithm and Output Files ...	42
Figure 6.2.1	Graphical Pitch Extraction	44
Figure 6.2.2	Algorithm Capabilities	47

LIST OF ABBREVIATIONS

DCA	Defense Communications Agency
TDHS	time domain harmonic scaling
ARC	adaptive residual coder
FWL	finite word length
TDHC	time domain harmonic compression
TDHE	time domain harmonic expansion
CVSD	continuously variable slope delta modulator
$s(k)$	input sampled speech signal
$y(k)$	compressed speech signal
$q(k)$	quantizer level sequence
$\hat{q}(k)$	system estimate of the original $q(k)$
$\hat{y}(k)$	system estimate of the original $y(k)$
$\hat{s}(k)$	system estimate of the original $s(k)$
LPF	low pass filter
FIR	finite impulse response
Hz	Hertz
kb/s	kilobits per second
b/s	bits per second
BER	bit error rate
Δ	triangular or change in
NQ	number of quantization level
MSF	maximum scaling factor
SF_q	q^{th} scaling factor
TR_q	q^{th} threshold

$e(k)$	error sequence
$\hat{e}(k)$	system estimate of original $e(k)$
$\sigma(k)$	approximation to the standard deviation of input sequence
dB	decibels
C_k	clipping level
E_k	approximation of the energy

CHAPTER I

INTRODUCTION

This report describes the results of a fifteen-month study, supported by DCA Contract 100-82-C-0026, of the real-time implementation of an algorithm combining time-domain harmonic scaling (TDHS) and Adaptive Residual Coding (ARC) at a transmission bit rate of 16 kb/s. The modifications of this encoding algorithm as originally presented by Melsa and Pande (1981) to allow real-time implementation are described in detail. A non real-time FORTRAN simulation using a sixteen-bit word length was developed and tested to establish feasibility. The hardware implementation of a full-duplex, real-time system has demonstrated that this algorithm is capable of producing toll quality speech digitization.

This report has been divided into two volumes. The first volume discusses the algorithm modifications and FORTRAN simulation. The details of the hardware implementation, schematics for the system and operating instructions are included in Volume 2 of this final report.

The minimum requirements for the TDHS-ARC speech digitization technique are that it should provide toll quality speech under optimum conditions and high quality robust speech in the presence of 60 dB acoustic background noise and in the presence of random channel error rates up to 0.001. The algorithm should be designed for a data rate of 16 kb/s. [Statement-of-work]

The trade-off between transmission quality and hardware complexity was studied with the overall objective of finding algorithm refinements that simplify implementation without degrading the previous performance.

It is important to note that minor sacrifices in system performance may be made to achieve significant hardware simplification. This volume presents the software simulation for a time domain harmonic scaling - adaptive residual coder system. It includes the finite word length simulations as the final form.

A necessary condition for understanding the direction and significance of this work is a thorough familiarity with the original form of the TDHS-ARC 9.6 kb/s system. Therefore, a brief survey of this system comprises the second chapter. Under the heading of "Final System", the third chapter outlines the final algorithm. This chapter will highlight the major modifications without providing the background associated with each variation. The fourth chapter presents the algorithm changes which were primarily due to the system options called for in the statement of work. The changes dictated by hardware considerations are discussed in the fifth chapter. The sixth chapter reports the effects of replacing the error-free channel approximation with a noisy channel. The seventh chapter is concerned with an explanation of the program options and output files, as well as their use. The final chapter forms a summary of the work and the final system.

CHAPTER II

ORIGINAL SYSTEM

2.1 Introduction

Throughout this report references will be made to the "original system". This is a reference to the 9.6 kb/s TDHS-ARC system presented by Pande [Melsa and Pande, 1981]. A block diagram which reflects the major points of interest of the system on this date is illustrated in Fig. 2.1.1, entitled "Original TDHS-ARC System". This is essentially the beginning point of the research herein reported. In order to evaluate the significance of this work, it is necessary to understand this original system. This chapter also highlights the characterstics of the original system which were modified to obtain the final system; these modifications are the subjects of subsequent chapters.

The basic structure of the original system is illustrated in Fig. 2.1.2. This structure was first proposed by Malah in 1980 when he presented the results of a TDHS-CVSD system and suggested the usefulness of combining TDHS with other waveform coders [Malah, 1980]. The structure is a time domain harmonic compression unit concatenated with an adaptive residual encoder in the transmitter, and an adaptive residual decoder concatenated with a time domain harmonic expansion unit in the receiver. The receiver's components perform the inverse function of the transmitter's sections (logically resulting in receiver expressions such as inverse ARC). The quality is limited by noise sources, such as, quantization in the ARC and compression versus expansion differences in TDHS. Again, it is worthwhile to note that this structure is the heart of the system and was maintained throughout the investigations.

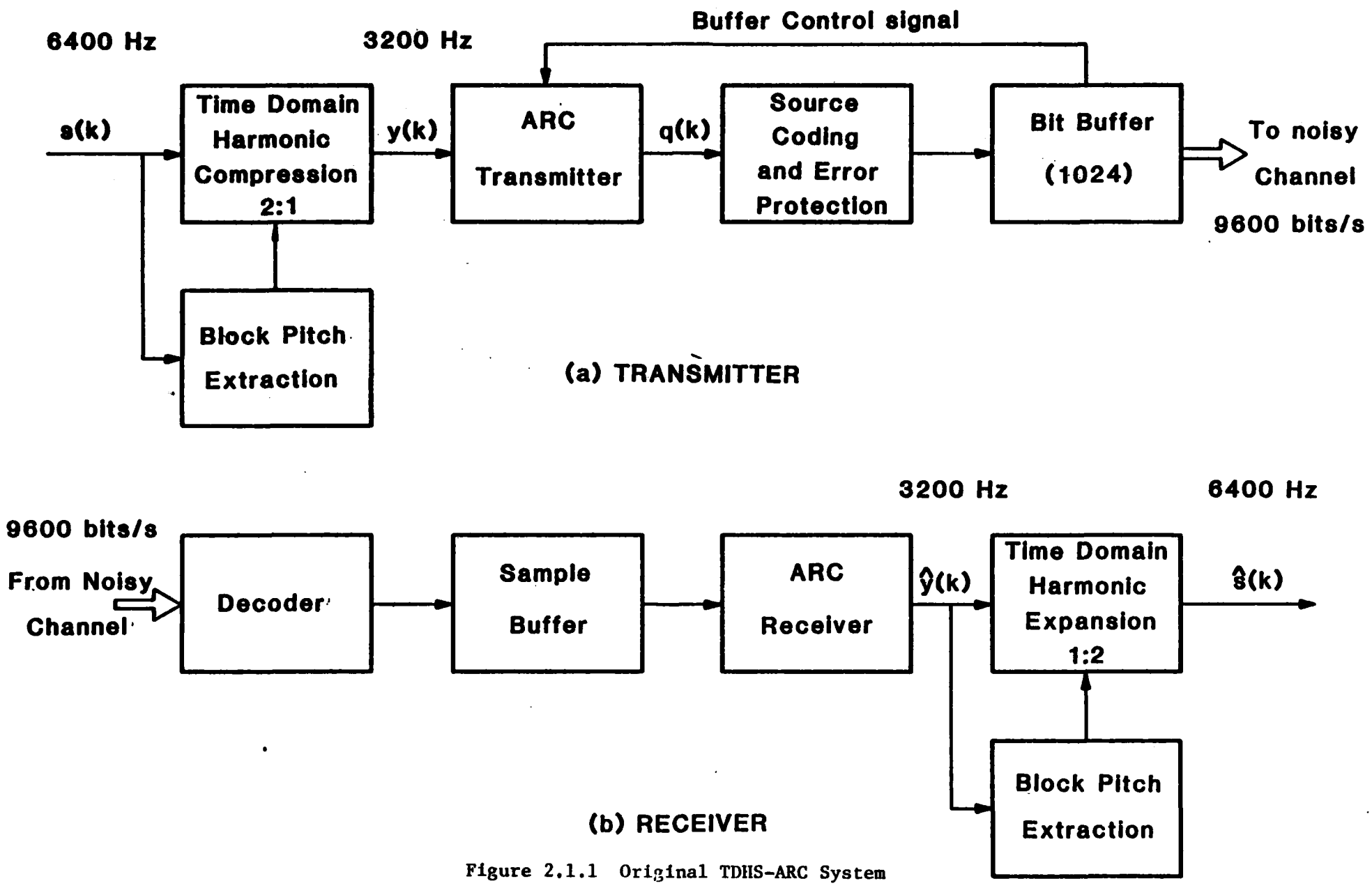


Figure 2.1.1 Original TDHS-ARC System

Figure 2.1.2 Original System Basic Structure

The many notable characteristics of the original system are identifiable as pertaining to the transmitter, channel, or receiver. They have been individually classified as such and are presented in the following sections. Many of the points of interest of the receiver are similar to those of the transmitter. They are, therefore, noted as being congruent and only briefly mentioned.

2.2 Transmitter

The input to the transmitter, in Fig. 2.1.1 is denoted s(k), an analog speech signal sampled at 6400 Hz. This sequence forms the input to both the compression block and the block pitch extraction unit which jointly form the TDHC algorithm in the transmitter. A compression ratio of 2:1 is used and although many acceptable* windows may be used, a triangular window was suggested due to computational simplicity [Melsa and Pande, 1981]. A higher compression ratio can be effective. However, the choice of a higher scaling factor entails the need for a more complex window function for equivalent performance [Malah, 1981]. The new characterization of block pitch extraction was precipitated by a major topic of investigation which is described in Chapter V. A pitch period is determined for a block of speech by locating the maximum of a correlation computation. The block length and range of the computation are 100 and 20 to 100 respectively. The output of the TDHE unit is a compressed 3200 samples/second speech signal.

---

* An acceptable window is defined as any window $h(i:N_p)$ such that

$$h(1:N_p) = 1$$
$$h(N_p:N_p) = 0$$

This is required by continuity considerations.

The compressed signal constitutes the input to the transmitter ARC. The encoder embodies a fourth order adaptive predictor and a 21 level buffer controlled quantizer. The adaptation of the sequentially adaptive linear predictor coefficients is based upon the steepest descent of the energy in the error between the predicted and actual values [Melsa and Pande, 1981]. The buffer control signal shifts the quantizer thresholds to reduce the possibility of buffer underflow or overflow. This is achieved by forcing or restricting the occurrence of a run length codeword. The sequence of quantization levels is the only information passed on to the remainder of the transmitter, the channel, and eventually, the receiver. The receiver must, therefore, require no more than this information.

The coding block accepts the quantization levels and maps them to a variable length codeword sequence that adds the error protection. The variable length code precipitates the need for a buffered output to the channel. The buffer serves a dual role. It allows the transmitter to emit bits at a constant rate and, as illustrated by the buffer control signal, it presents the adaptation information to the variable quantizer. The buffer outputs bits to the channel at a transmission rate of 9600 bits per second.

2.3 Channel

The original system incorporated a noisy channel simulation with an interactive bit error rate. The channel model uses a pseudo-random number generator to determine the location of a bit error in the channel sequence. Given the decision of the pseudo-random number generator, the algorithm changes the bit stream accordingly.

The channel is constrained to a 9600 bits/second rate. Although this is a channel constraint, its influence is accommodated in the receiver and transmitter algorithms by restricting the average bits allowed per quantization level. In this way, it affects the coding and decoding blocks and the fineness of quantization in both ARC algorithms.

Therefore, the importance of the simulated channel lies in its ability to induce errors in the bit stream between the transmitter and receiver. The transmission rate of 9600 bits/second plays a role in the transmitter and receiver algorithms and not in the channel algorithm.

2.4 Receiver

A bit stream at the transmission rate of 9600 bit/second is received from the channel by the decoder. The decoding block maps the bit stream back into a sequence of quantization levels which may or may not deviate from the original quantizer level sequence. The efficiency of mapping $q(k)$ to $\hat{q}(k)$ is dependent only upon the specified error correcting capability, bit error rate, and code. Again, the requirement of a sample buffer is generated by the utilization of a variable length code. The sample buffer contains the quantization levels contaminated by noise induced in the channel.

These levels are accessed by the inverse ARC block. This ARC is also a fourth order adaptive linear predictor combined with a 21 level quantizer. Note that there is no need for a buffer control signal in the receiver and that the receiver need not know the threshold values used by the transmitter. Other than these major disparities, the inverse ARC operates as the exact inverse to the encoder in the transmitter.

The time domain harmonic expansion element in the receiver acts, as stated earlier, much the same as the compression blocks. A pitch extraction routine identical to that in the transmitter, is exploited in the receiver. The major relevant differences are the expansion ratio, which is, of course, 1:2, the exact inverse of the compression ratio, and the window. Here, the choice of windows is forced due to the choice of a triangular window in the transmitter. Once again employed is the notion of an acceptable window. That is, one that maintains the continuity at its fringes, as well as those of the succeeding window. The impelling concept of this algorithm, as a whole, is the production of a high quality sequence of estimates, $\hat{s}(k)$, to the original speech sequence, $s(k)$. This estimate forms the output of the receiver and hence the algorithm. It is considered a 6400 Hz sampled speech signal estimate.

CHAPTER III

FINAL SYSTEM

3.1 Introduction

A block diagram of the final system is illustrated in Fig. 3.1.1. This simulation incorporates the specifications cited in Chapter I. It is immediately apparent that the final algorithm maintained the basic structure discussed in the opening remarks of Chapter II and illustrated in Fig. 2.1.2. The transmitter preserves the TDHC-ARC format and the receiver upholds the inverse ARC-TDHE structure. Given that the major concept of a TDHS-ARC system is maintained, the points of interest of the final algorithm are the modification of the individual components. Although detailed discussions of these changes are deferred for future chapters, they are highlighted in the remainder of this chapter.

3.2 Transmitter

The input to the transmitter is composed of an analog speech signal sampled at 8000 Hz. This signal forms the input to both the compression block and the pitch extraction block. The compression ratio of 2:1, is preserved in this final algorithm. Also preserved is the utilization of a triangular window, the length of which is twice the detected pitch period length.

The pitch extraction block underwent crucial transformations. The input sampled speech is first low-pass filtered by a 9th order linear phase FIR digital filter before pitch detection is performed. Following this operation, a serial pitch extraction routine is executed. The investigation of this routine constituted the majority of the effort

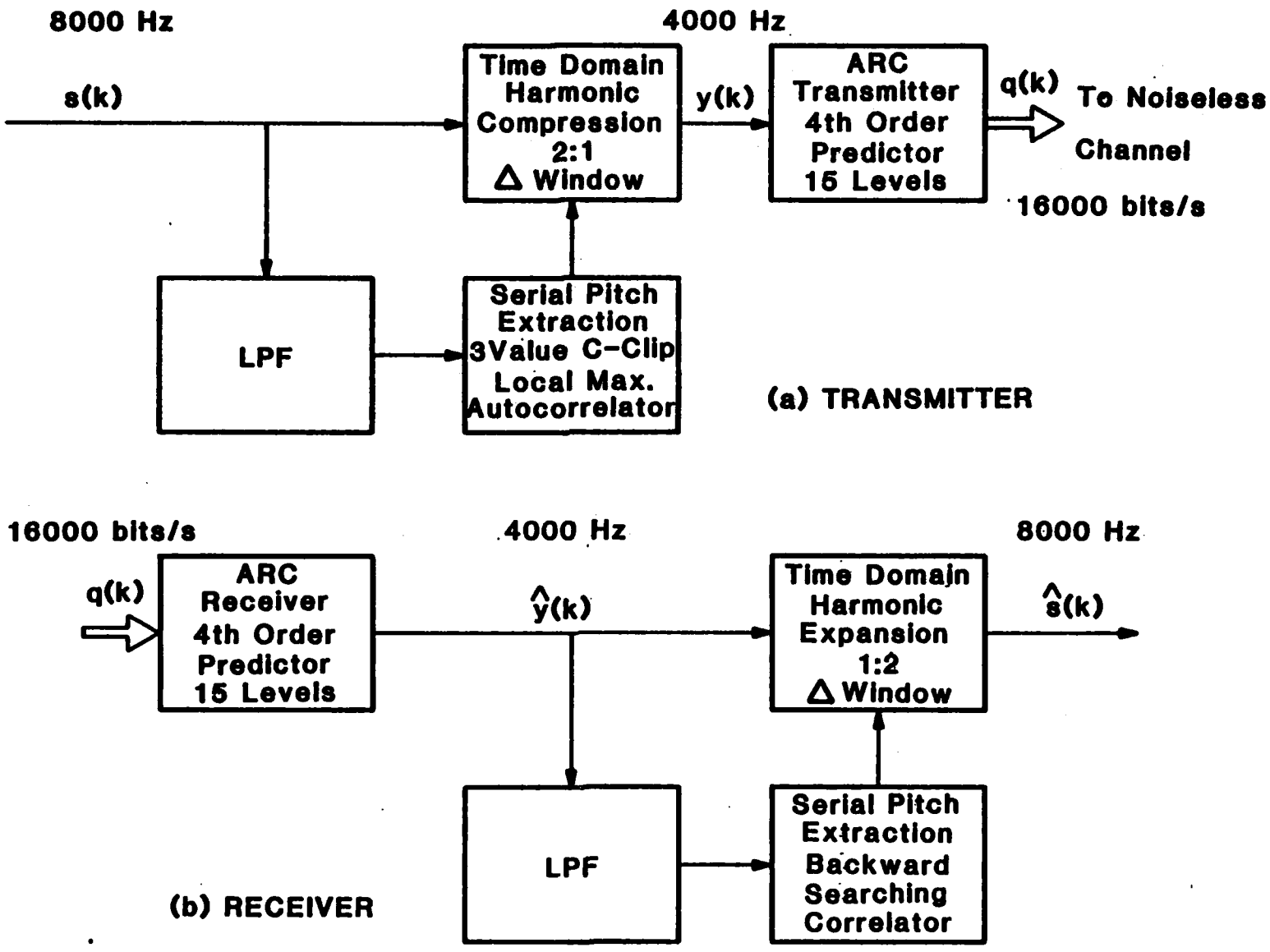


Figure 3.1.1 Final TDHS-ARC System

allotted to the modification of the TDHS algorithm. It is based on hardware considerations presented in Chapter V. Pseudo-random pitch picking and a new processing of the correlation function's data are exercised by the changes. The hardware simplicity associated with the three value center clipped autocorrelation computation was retained.

The adaptive residual encoder found in the transmitter incorporated a fifteen level fixed quantizer and a fourth order linear predictor. The adaptation procedure is modified to eliminate the division operation called for in the original system. This is accomplished by adopting a fixed gain adaptation procedure. Another division operation is eliminated by using the quantizer normalization factor as a multiplicative adaptation to the thresholds and scaling factors as opposed to a dividing adaptation to the error signal. This results in only a mild truncation of the original system's performance. The division operations are eliminated to make the algorithm hardware compatible with the NEC7720 signal processing chip which does not support a division operation. The use of a fixed length code, a sampling rate of 8000 Hz and the restriction to a 16 kilobit/second transmission rate dictated that the maximum number of quantization levels be 15.

Due to algorithm changes presented in Chapter IV, the quantization levels became the information passed from the transmitter to the receiver. The 15 quantization levels require 4 transmission bits/sample resulting in an effective transmission rate of 4000 quantization level samples/second. The quantization levels, therefore formed the output of the transmitter.

3.3 Channel

A temporary approximation to a noiseless channel is instrumental in the reduction of the transmitter and receiver algorithms to their form in Fig. 3.1.1. The performance of the system in a noisy channel is reported in Chapter VI. A crucial consequence of the noiseless channel approximation is that the output of the transmitter is exactly the input to the receiver. In essence, the channel is not present, save the transmission rate characteristic.

The channel transmission rate of 16 kilobits/second is another of the channel's features that dictated significant alterations to the original system. Although these variations did not predominate in the channel model itself, they are critical to the final algorithm. The transmission rate restricted the quantization of the ARC and thus, upper-bounded the signal-to-noise ratio of the ARC system. The effects of transmission rate, then, are sustained in the transmitter and receiver and not in the channel.

3.4 Receiver

The channel output, which exactly coincides with its input and the transmitter output, becomes the input to the receiver. It is considered a 16 kilobit/second signal representing 4 bit codewords of the quantization levels. Since the quantizer levels form the transferred information, no decoding is necessary and inverse ARC is applied directly to the input. Inverse ARC in this system is equivalent to the feedback loop of the ARC in the transmitter. It contains a fourth order linear predictor as in the original system. The quantizer has 15 levels and is

fixed in the final algorithm as opposed to the 21 adaptive levels used in the original. The attractive characteristic of the inverse ARC (not requiring the threshold values) is maintained in the final algorithm.

The compressed signal $\hat{y}(k)$ is a 4000 Hz sequence. It is the estimate of the compressed speech signal in the transmitter, $y(k)$, and is the driving sequence of the expansion and pitch extraction units of the receiver. The expansion block of the receiver behaves exactly as that of the original system. This block perpetuates the 1:2 expansion ratio and the triangular window suggested by the original. Again, research activities concentrated on the pitch extraction algorithm. The advent of serial pitch extraction and its inclusion and success in the transmitter ordained its use in the receiver.

In previously investigated algorithms, the pitch extraction algorithm of the receiver was identical to that of the transmitter. An important innovation of this final algorithm is that there exists a major difference in the receiver's algorithm as compared to the transmitter's. Noting that the receiver has a reduced amount of pitch information available to it, the receiver algorithm is modified to more judiciously use this information. The receiver correlates backward as well as forward one searching blocklength for pitch information. The transmitter algorithm only looks forward.

The product of the receiver is a sequence, $\hat{s}(k)$, which is the approximation of the input speech sequence, $s(k)$. The rate is equal to the input rate of 8000 Hz. The similarity $\hat{s}(k)$ and $s(k)$ is used as an indication of the performance of the system. Therefore, side-by-side taped comparisons are used as a major indication of performance.

CHAPTER IV

INITIAL ALGORITHM REFINEMENTS

4.1 Introduction

The initial algorithm refinements are described in this chapter. These refinements are those that are principally required by the mandates of Chapter I. The hardware-induced modifications are deferred to Chapter V. Chapter I imposes two considerable requirements. It requests a transmission rate of 16000 bits/second and a bit error rate of 0.1%. These conditions immediately translate to significant algorithm adjustments. Although both stipulations are channel characteristics, it is necessary to modify the transmitter and receiver algorithms since the channel is considered unalterable.

The system, as it appeared following these initial changes, is illustrated in Fig. 4.1.1. Hardware induced refinements that arose after these initial changes, such as the switch to 8 kHz sampled input speech, are considered in Chapter V. The discussion of their effects on the refinements presented in this chapter are also deferred to Chapter V. The following discussion is subdivided into two sections. The first describes the refinements due to the new transmission rate. The second relates the added alterations caused by the BER condition.

4.2 16 kb/s

The 16 kb/s transmission rate and the use of a 6400 Hz sampled speech signal affected the ARC and inverse ARC considerably. The high transmission rate and the low sampling rate allowed the use of a fixed length (non-run length) code. A bit rate of 16 kb/s and compressed

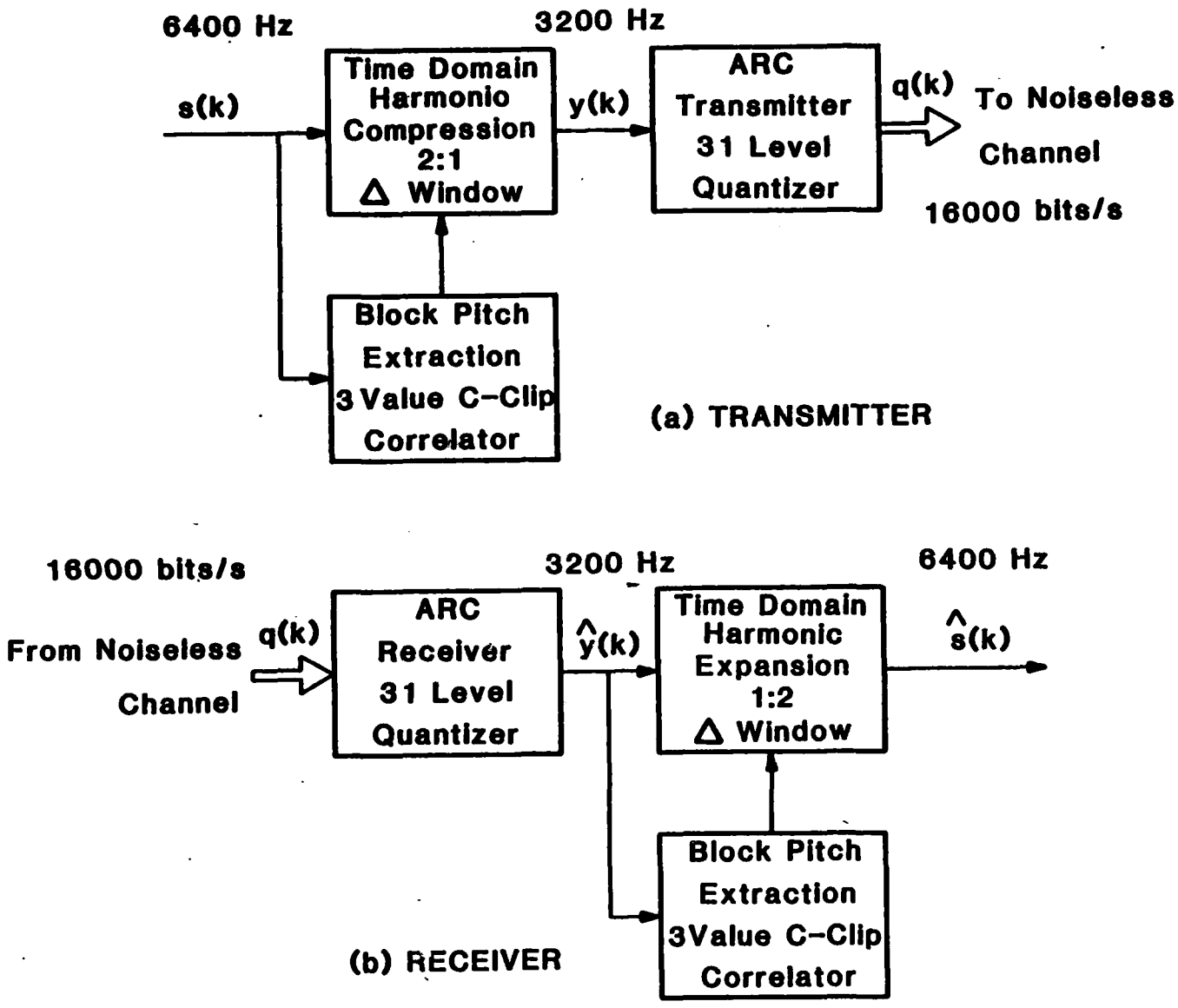


Figure 4.1.1 First Stage of Algorithm Refinements

speech rate of 3200 Hz entitles the algorithm to an average of 5 bits per sample. A five bit fixed length code allows for a 32 level quantizer. Since the quantizer is symmetric about its center, an odd number of quantizer levels are used. Therefore, the sampling frequency and transmission rate dictate a 31 level quantizer if a fixed code is used. Since an eleven level quantizer produced good results [Pande, 1981] in the original algorithm a 31 level quantizer is initially accepted (and experimentally supported in subsequent tests).

Earlier investigations involved fewer levels (11, for instance). In these investigations, the thresholds, scaling factors, and expansion factors are judiciously hand-picked. (The thresholds and scaling factors are the input and output values of the quantizer and the expansion factors are the scalars used to update the standard deviation computation.) The advent of 31 levels brings with it the need for a more systematic method of choosing these values. Because the magnitudes of speech samples are inherently exponentially distributed, a method which incorporated a logarithmic scaling of the quantizer levels was pursued. The final system allows for the choice of a base and a maximum scaling factor. The method is illustrated in Fig. 4.2.1. In this figure, the following abbreviations and definitions are employed

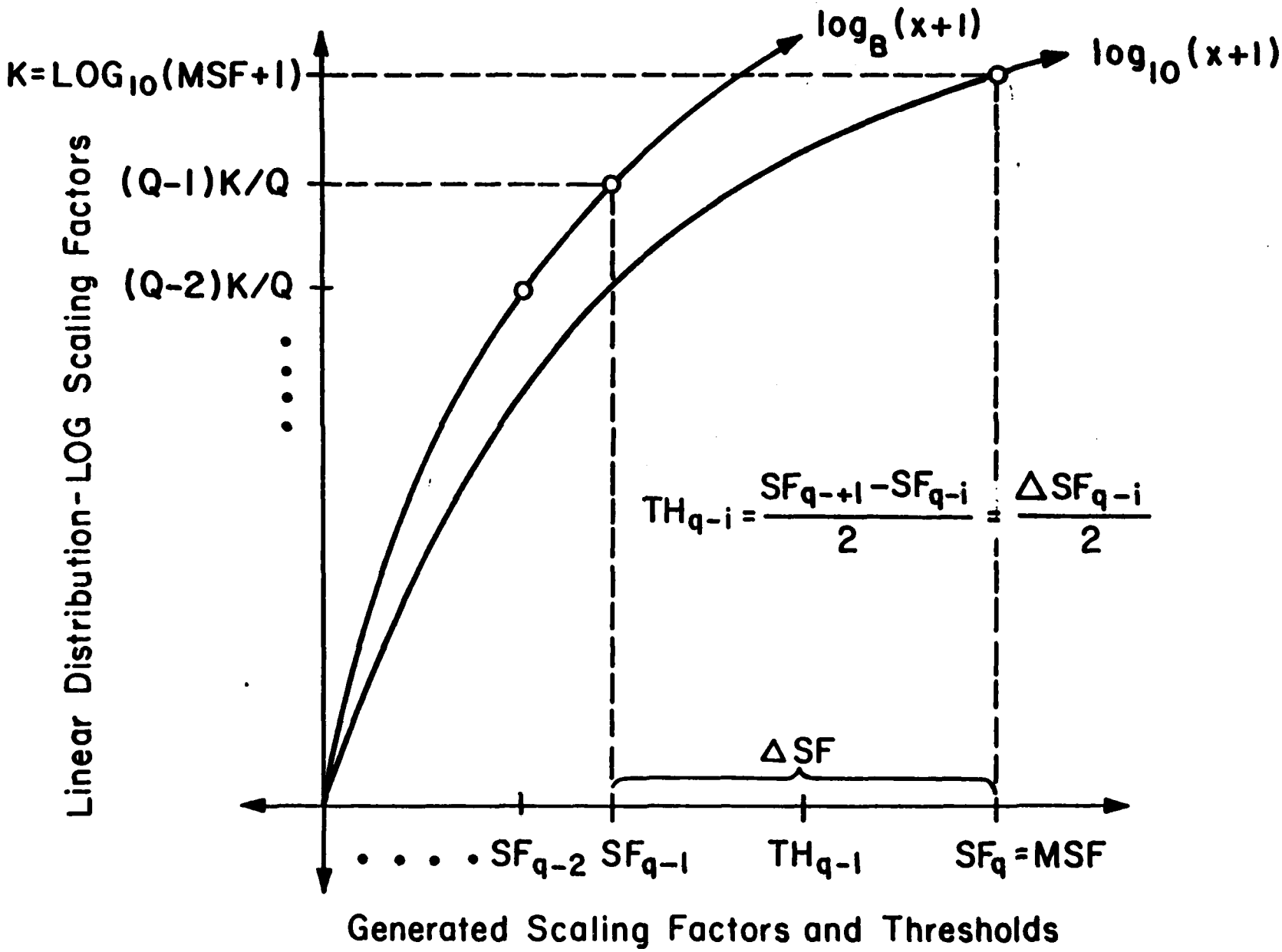
NQ = number of quantization levels,

MSF = maximum scaling factor,

SF_q = q^{th} scaling factor, and

TH_q = q^{th} threshold.

The quantizer is symmetric about zero because speech is nearly as



DETERMINATION OF SCALING FACTORS AND THRESHOLDS

negative-going as it is positive-going. Therefore, the zero-level is pre-chosen and there are only Q independent levels, where

$$Q = \frac{NQ-1}{2} .$$

The maximum scaling factor is mapped to the value K on the ordinate by the shifted logarithmic function

$$Y = \log_{10}(X+1).$$

(X being the independent variable.) The ordinate is then equally divided into Q sections from 0 to K where

$$K = \log_{10}(MSF+1) .$$

The ordinate values are, therefore,

$$\begin{gathered} 0 \\ (1/Q)K \\ (2/Q)K \\ \vdots \\ ((Q-2)/Q)K \\ ((Q-1)/Q)K \\ K \end{gathered}$$

It is desired to map these ordinate values back to the abscissa to obtain the logarithmically scaled values. To add flexibility to this technique, the base of the logarithm used to map the values from the ordinate to the abscissa, termed the distributing base, was made

variable. The change in base from 10 to the chosen B allows the lowest Q quantizer levels to be expanded away from or contracted toward the origin. It was noted that if the distributing base was chosen to be too large, the second largest ordinate value would be mapped to a value that exceeds the requested maximum scaling factor.

An upper bound on the distributing base is determined. The bound is resolved using the constraint that the second largest ordinate value cannot be mapped back to a value that exceeds the requested maximum scaling factor. The bound is

$$\text{Base} \leq 10 \exp \left\{ \frac{NQ}{NQ-1} \left[\log(MSF+1)/\log(MSF)\right] \right\}$$

A distributing base, B, is then chosen and the function

$$Y = \log_B (X+1)$$

is used to map the Q-1 unspecified scaling factors back to the abscissa. Here, Y would be the independent variable and in the usual presentation, the function is written

$$X = (B^Y-1)$$

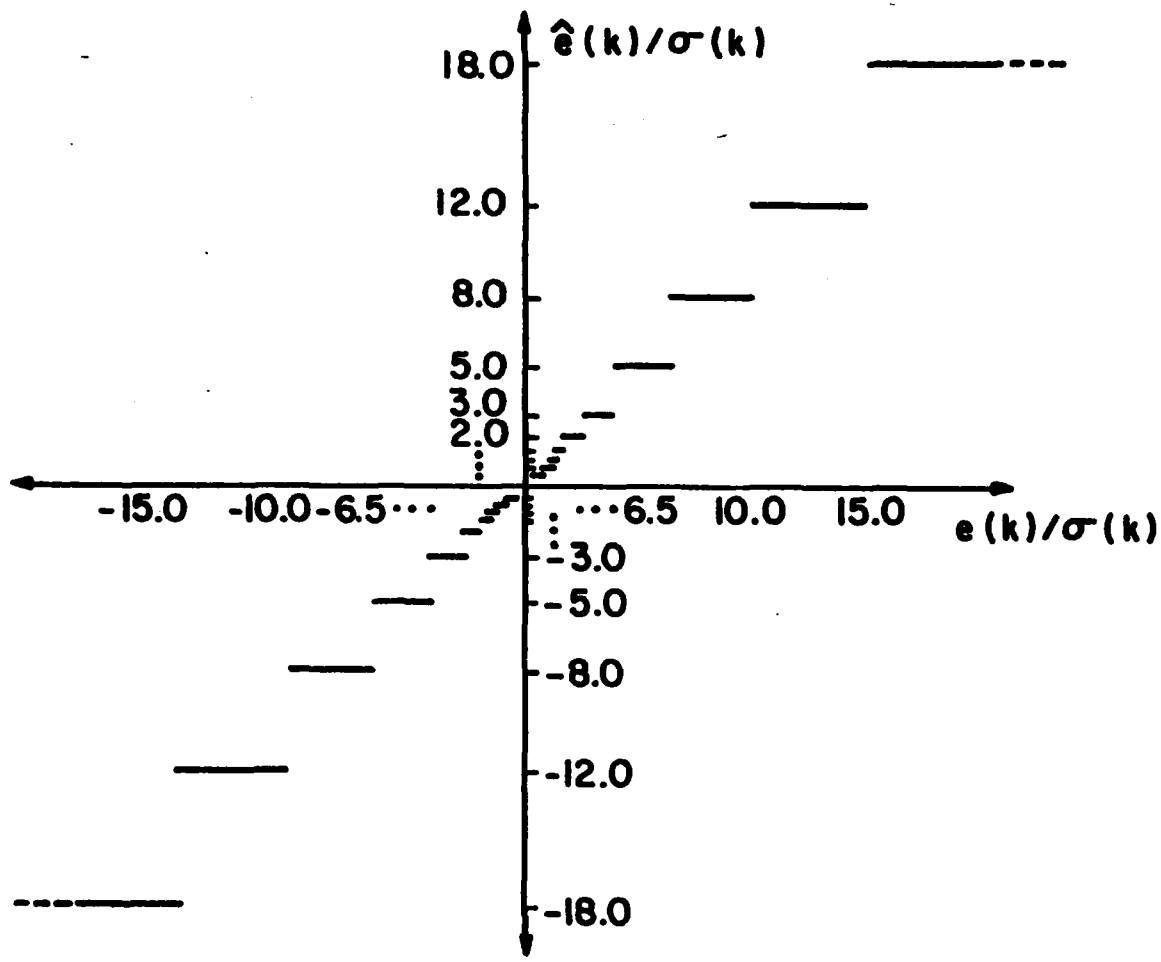
A shifted logarithmic scale is exploited to force the second function to map the scaling factors logarithmically <u>from the zero level</u> to the second largest scaling factor. The largest ordinate value, K, is mapped back to the abscissa using the original base (=10) so that the specified maximum scaling factor is maintained.

The effectiveness of this technique is justified by a comparison between the methodically chosen factors and the hand-picked factors. A minimum mean square error is used to compare the two sets of data. Other comparisons, such as percent error were considered and discarded. Their undesirable weighting of small errors in the lower scaling factors led to their disqualification. Using the mean square error criterion, a local minumum mean square error was found to be 0.4 and occurred at B = 9.0.

As in preceeding work [Melsa and Pande, 1981], the thresholds were chosen to be equidistant from successive scaling factors.

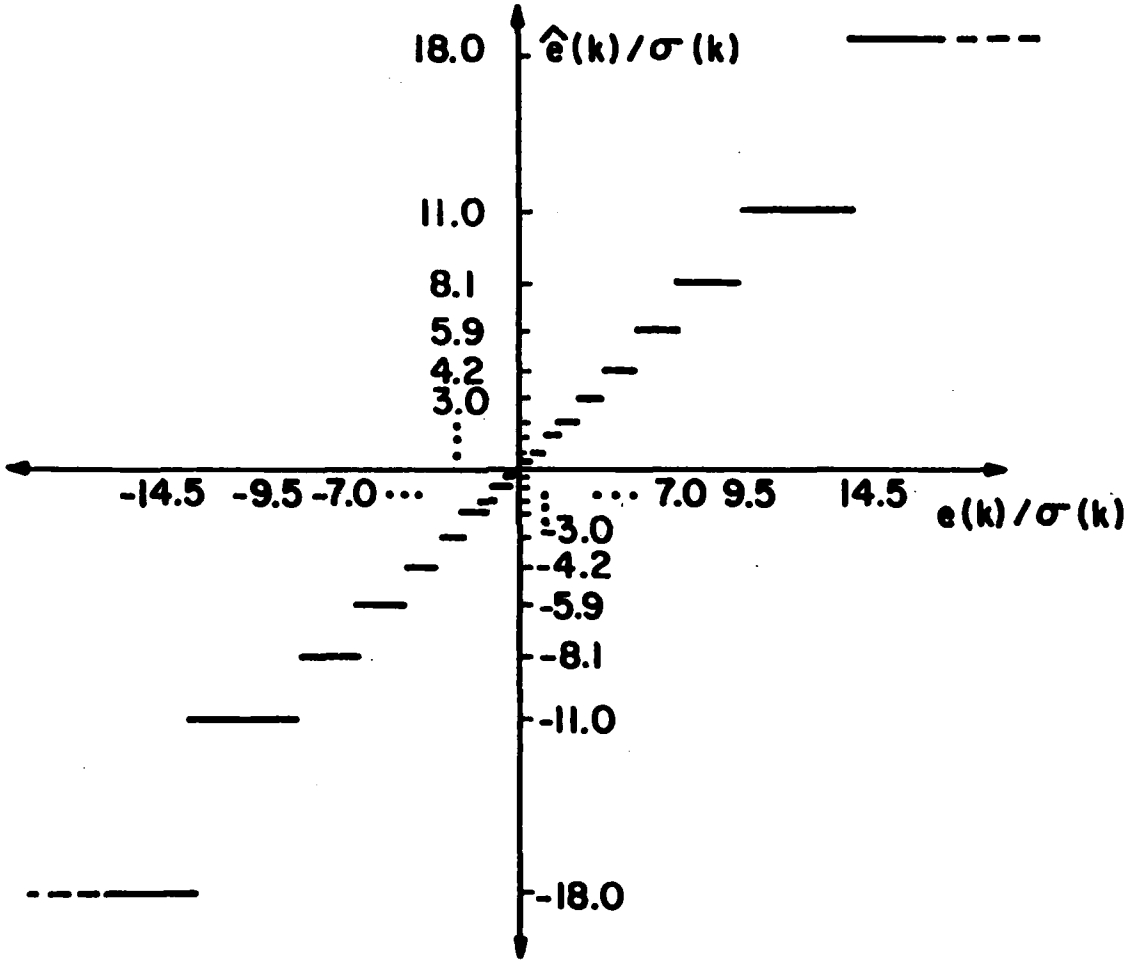
The hand-picked data set used in the comparison is depicted in Fig. 4.2.2 and the resulting logarithmically distributed scaling factors are illustrated in Fig. 4.2.3. Both of these figures incorporate the usual input-output design of past publications. The scaling factors and thresholds agree nearly exactly except for one level in the middle of each side. This abberation is momentarily ignored.

The expansion factors are the parameters that remain unassigned. The expansion factors are used to force the normalized error to the middle of the quantizer. The approximation to the standard deviation should, therefore, remain unchanged if the center of either half of the quantizer is engaged. For this reason and due to the availability of a large number of factors, the middle three factors are set to unity. The remaining upper and lower values are distributed linearly between the largest and smallest expansion factors, respectively. An Aitken double sweep multivariate search [Luenberger, 1965] determined the local maximum of the SNR; pinpointing the maximizing maximum and minimum expansion values.



HAND PICKED SCALING FACTORS AND THRESHOLDS

Figure 4.2.2



LOGARITHMICALLY DISTRIBUTED SCALING FACTORS AND THRESHOLDS

Figure 4.2.3

A table listing of the quantizer thresholds, scaling factors, and expansion factors for speech at a 6400 Hz and a 8000 Hz sampling rate is presented in Fig. 4.2.4. The large number of expansion factors allowed a tighter tracking of the normalized error sequence's movements. This reduced the occupancy frequency in the outer levels of the quantizer. The maximum scaling factor was appropriately reduced to improve the use of the quantizer levels.

4.3 Bit Error Rate

The low channel bit error rate of 0.1% precipitated a noiseless channel approximation. The simulation results, therefore, established a tight upper bound to the system performance in the presence of a 0.1% BER channel. The assumption of no channel errors facilitated the simulation considerably.

An outcome of this assumption is the inclusion of the coding and decoding blocks in the channel model. Since no errors are generated, the estimated quantizer levels $\hat{q}(k)$, of the receiver, are exact replicas of the original quantizer levels, $q(k)$. There is no reason, therefore, to model the coding and decoding blocks. They are deemed nonessential and removed from the transmitter and receiver. These blocks are, of course, essential to a hardware system and for this reason, they are considered as part of the channel.

This concept is illustrated in Fig. 4.3.1. There, the bold lines denote the simulation flow of information. The dotted lines illustrate the imagined information path through the noiseless channel.

i	Ti (6.4)	Ti (8)	SF_i (6.4)	SF_i (8)	$EXPN_i$	$EXPN_i$
1	.1	.16	0.0	0.00	.5	.50
2,3	.3	.54	.2	.33	.6	.75
4,5	.5	1.1	.4	.76	.7	1.00
6,7	.7	1.7	.6	1.34	.8	1.00
8,9	1.0	2.6	.8	2.11	.8	1.00
10,11	1.3	3.8	1.1	3.13	.9	2.00
12,13	1.7	6.2	1.5	4.48	1.0	3.00
14,15	2.1		1.9	8.00	1.0	5.00
16,17	2.7		2.4		1.0	
18,19	3.3		2.9		1.6	
20,21	4.0		3.6		2.3	
22,23	4.8		4.4		2.9	
24,25	5.8		5.2		3.5	
26,27	6.9		6.3		4.1	
28,29	9.2		7.5		4.8	
30,31			11.0		6.0	

Figure 4.2.4 Quantizer Thresholds, Scaling Factors, and Expansion Factors for Speech at 6400 Hz and 8000 Hz Sampling.

The removal of the buffers in the transmitter and receiver is assumed when this modification is made. The buffers are removed due to a combination of the fixed length code condition and the temporary assumption of no channel errors. The removal of the buffer takes with it the buffer control signal. Since there is no error protection or correction coding needed and further since a fixed length code is used, transmission need not be buffered. The original system incorporated a variable threshold quantizer that adapted according to a buffer control signal. This is now unnecessary.

The modifications noted in this chapter are justified by a side-by-side comparison of processed and unprocessed speech sentences and the predictor's computed performance. At this point, the SNR of the predictor was recorded in excess of 22 dB for sentence 1 and over 27 dB for sentence 6. A side-by-side listening session established the speech estimate as being a very good representation of the original speech.

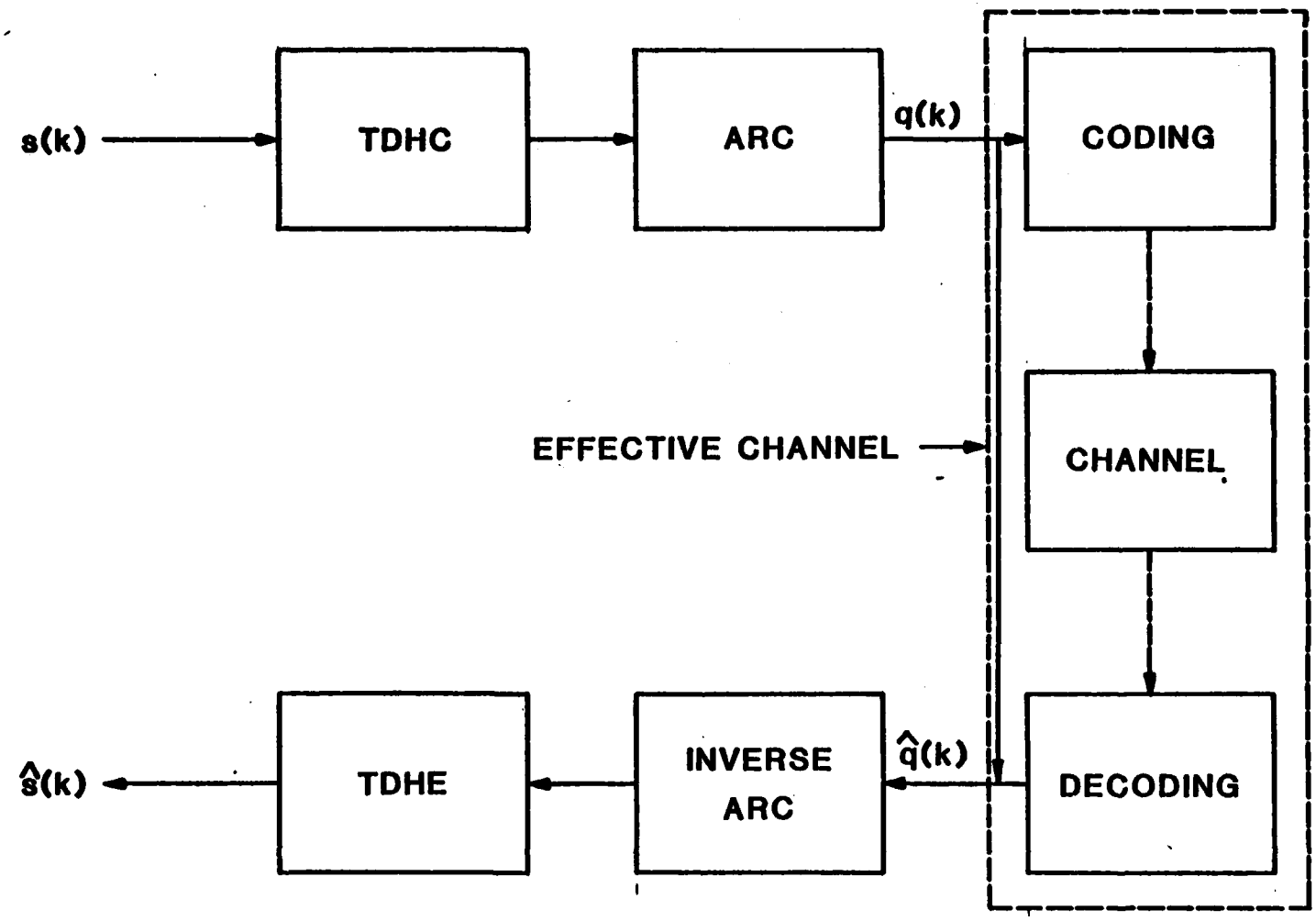


Figure 4.3.1 Channel with Temporary BER Assumption

CHAPTER V

HARDWARE INDUCED MODIFICATIONS

5.1 Introduction

The refinements that are principally induced by hardware considerations are presented in this chapter. It is subdivided into four sections. They are 1) serial pitch picking, 2) 8 kHz sampling rate, 3) hardware compatible ARC, and 4) finite word length. The section titles refer to the hardware desired condition which instigated the investigation into that portion of the algorithm.

5.2 Serial Pitch Picking

The original algorithm chose pitch by first buffering a block of speech samples. The first and last third of the samples in the block are then scanned for the maximum speech sample magnitude and the clipping level is set to a specified percentage of this maximum value.

$$C_K = \gamma \cdot \max\{ S(n) : n \in [\ 0, \frac{L}{3}\ ] \quad \text{or} \quad n \in [\ \frac{2}{3} L, L\ ]\}$$

where L is the block length in samples, C_K is the clipping level for the kth determination of pitch, and s(n), $n \in [0,L]$ are the speech samples in the buffer. Another buffer of the same length, L, is filled by three value center clipping the buffered speech samples. A correlation computation is performed on the clipped speech and a shifted version of it. The correlation is computed over the specified search range for each shift between its maximum and minimum shifts. The shift value associated with the maximum correlation computation is accepted as the pitch period. A window is formed using this value and compression or

expansion is performed. This technique of pitch detection is especially attractive due to its simple hardware implementation.

After the transmitter compresses the speech, the speech buffer is shifted by two pitch periods and refilled with new samples. By shifting out two pitch periods of processed speech and shifting in two pitch periods of new speech, the next clipping level, C_{K+1}, can be different than the previous clipping level, C_K. This is caused by the term "max{•}" in the clipping level computation, which can result in different values for the two speech sample buffers. Since the entire buffer of speech is clipped using a single clipping level, the unprocessed samples of the previous speech buffer must be reclipped with the new clipping level. This means that the entire previously clipped speech is useless and must be discarded. Furthermore, clipping cannot take place until the speech buffer is completely filled and a new clipping level is determined.

The problem of waiting for the speech buffer to fill before starting the clipping process and the problem of clipping speech more than once before processing the associated speech samples are eliminated by serial pitch picking [Arora, 1983]. The solution to this time wasting problem is termed serial pitch picking. The serial technique uses a scaled approximation to the speech signal energy as a sample by sample updated clipping level. The clipping level, C_k, is

$$C_k = \beta \cdot E_k$$

where

$$E_k = \alpha \cdot E_{k-1} + (1-\alpha)|S_k|.$$

β is the scaling constant and α is a constant related to the length of

the transient response. The response of this computation is described by

$$H(Z) = (1-\alpha)\frac{Z}{Z-\alpha}$$

which describes a low pass filter whose time constant, in samples, is a function of α. The time constant, n, in samples is

$$n \cong \frac{1}{1-\alpha} .$$

As the new speech enters the buffer, the energy is updated, the clipping level is determined, and the speech is three value center clipped using this value. The clipped speech sample is shifted into the clipped speech buffer and the routine is ready to accept the next speech sample. If the previously detected pitch period is large, a large number of samples must be accepted and clipped to set up the buffers for the next pitch detection. The large number of new samples will take a proportionately large hardware time period to clip. This will not present a problem since the time between pitch detections is also proportional to the pitch period length. (As the number of samples involved in the compression increases, the time required to compress these samples increases and the frequency of pitch detection decreases.)

The unprocessed speech of the previous speech buffer was clipped at the correct clipping levels. The clipped speech associated with the unprocessed speech is, therefore, retained and the redundancy of the clipping in the block pitch picking is eliminated. The clipped speech is generated as the samples enter the speech buffer. Therefore, the

problem of waiting for the speech buffer to fill found in block pitch picking is also eliminated.

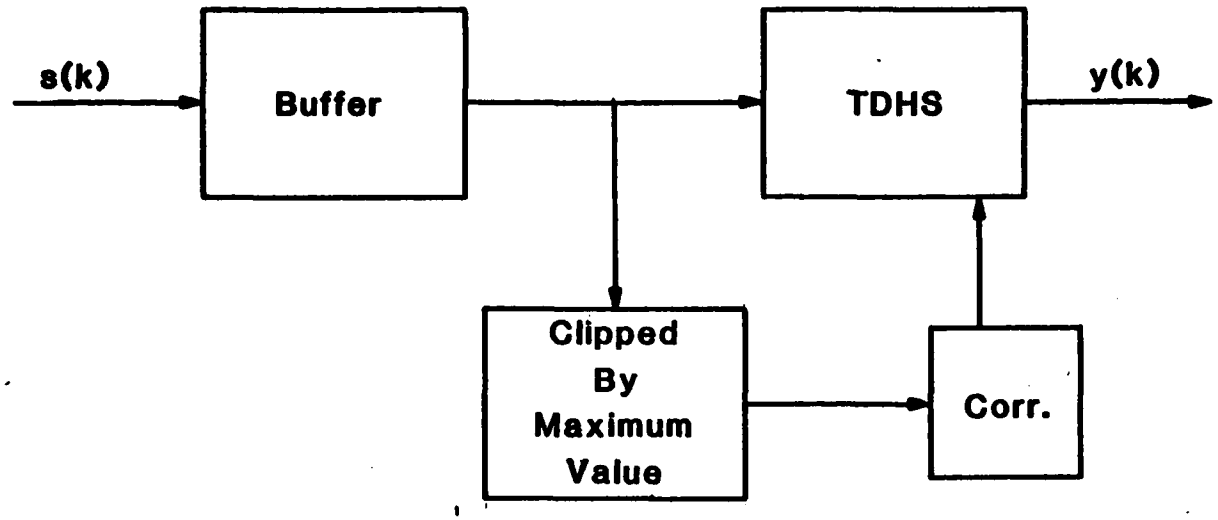
The computational simplicity of the three value center clipped speech correlator is sustained. Therefore, serial pitch picking retains the simple hardware implementation of the block pitch picking and yet eliminates the problems associated with it. The comparison of the two techniques is illustrated in Fig. 5.2.1. The difference in the algorithm for the receiver is that at most one maximum pitch period of samples could be used whereas the transmitter's maximum is twice this value.

Numerous tests were performed and the suggested parameter values are

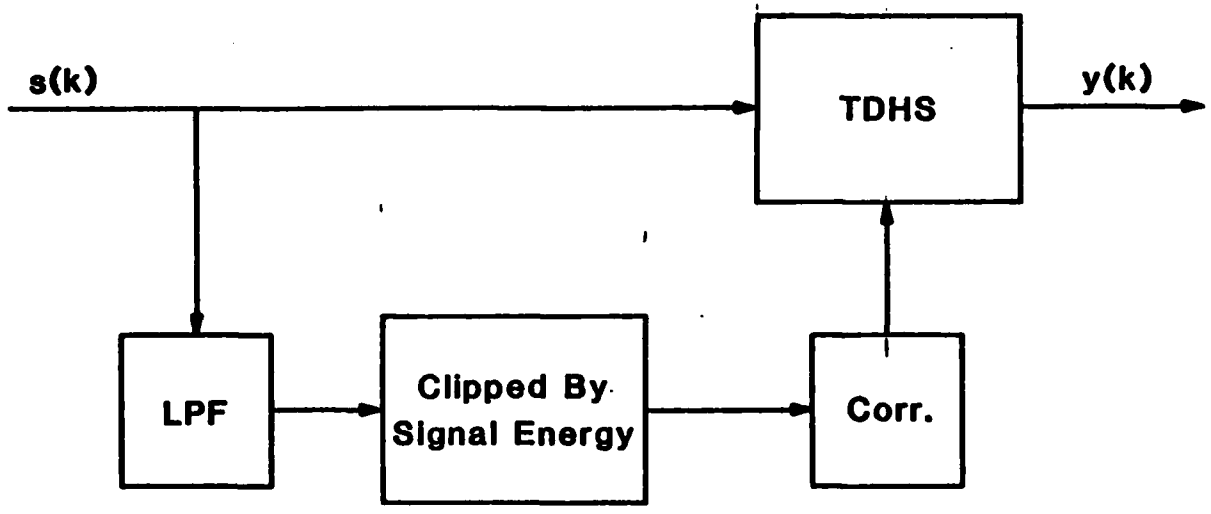
$$\beta = 1.5$$
$$\alpha = 0.992$$

This α value allowed the proper tracking of the signal energy required by the technique and the above β value insures the success of the clipping routine by properly scaling the resultant signal energy. The fundamental concept of three value center clipping is to retain only the first couple of positive and negative going pulses of each pitch period [Schafer, 1976]. Figure 5.2.2(a) shows how block pitch extraction succeeds in clipping in this manner. Figure 5.2.2(b) demonstrates serial clipping's similar performance and Fig. 5.2.2(c) is the independent result of the two methods.

Two problems developed in the serial clipping implementation. Half pitch was sporadically picked in the voiced segments and a buzzing sound appeared in the output speech during unvoiced segments. This buzzing sound is attributed to the repeated minimum pitch periods picked in



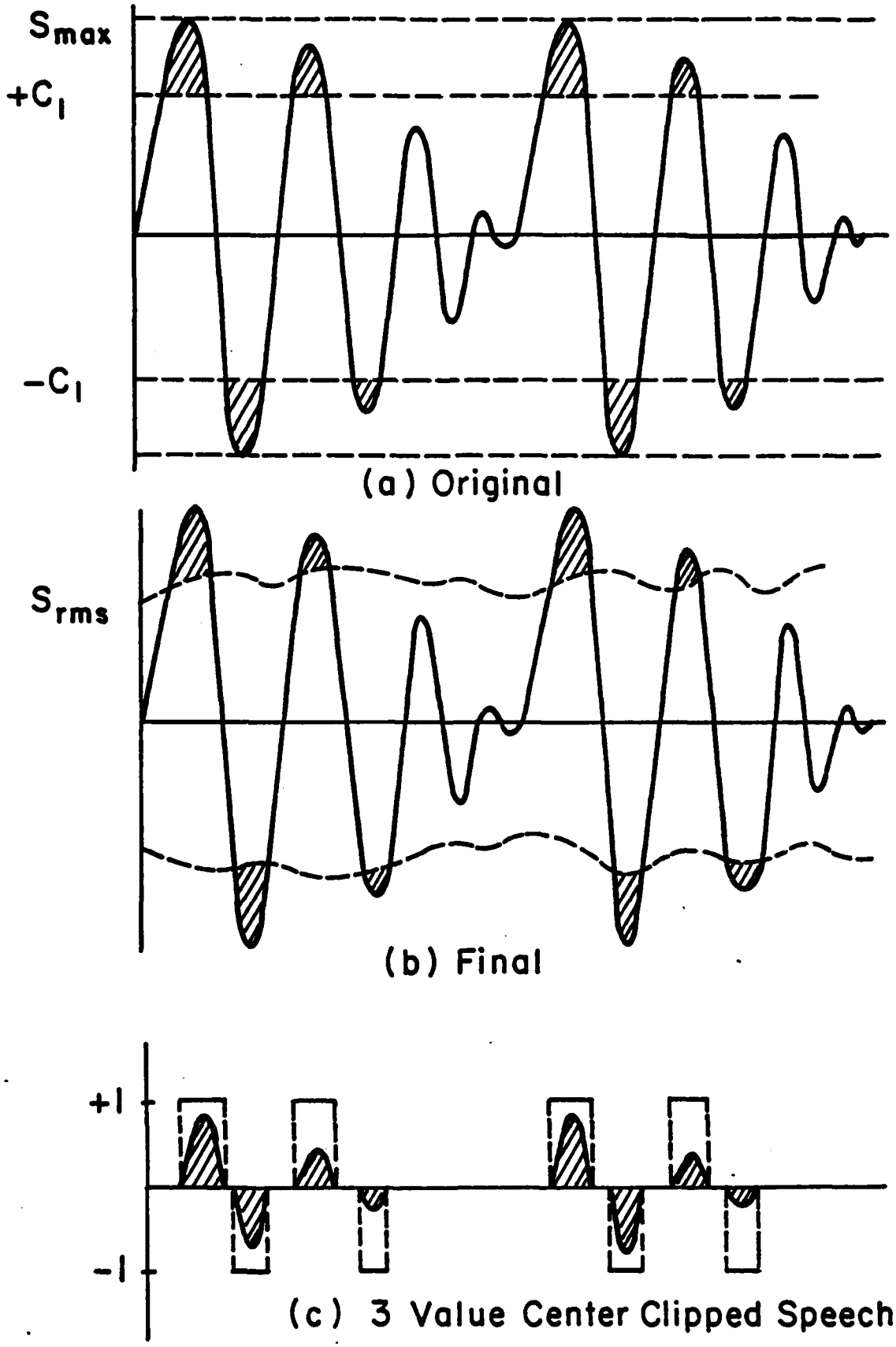
(a) Original



(b) Hardware Compatible

TDHS ALGORITHMS

Figure 5.2.1



CENTER CLIPPING ALGORITHMS

Figure 5.2.2

unvoiced segments by the serial pitch extraction technique. Both problems are charged to the clipped speech sequence, which alone, affects the correlation computation.

The half-pitch problem is solved by implementing a digital low pass filter and by modifying the processing of the correlation computations. The actual correlation computation, itself, remained unaltered. However, the processing of the elements of the correlation vector was changed. The routine now seeks the largest of the correlation vector's one sample local maxima. A one sample local maximum is defined as an element whose magnitude is greater than or equal to its immediate neighbors. The largest local maxima indicates the pitch period as did the vector's maximum element in the original algorithm. The incorporation of this strategy eliminated the possibility of choosing either the maximum or minimum pitch when they represent the rising or falling side, respectively, of a correlation sequence's local maximum. This technique aided in picking true pitch in the voiced regions.

The half-pitch problem is further mastered by installing a low pass filter between the speech sequence and the clipper. It is a 9th order linear phase FIR filter (see Appendix D). The filter is designed to smooth out the anomalies of the individual voiced segments, thereby forcing the pitch periods to appear more similar. This aids the correlator by coercing the clipped speech of each pitch period to be more alike. The result of these two methods is the elimination of half-pitch picked in voiced segments.

The problem of repeated minimum pitch period picked in the unvoiced segments is corrected by implementing a controlled random pitch

generator. Whenever the minimum pitch period is picked, a pseudo-random pitch is chosen and employed as the correct pitch. The random pitch is restricted to the lower half of the search range so that a large randomly picked pitch period in an unvoiced region will not adversely affect the beginning of a voiced segment. Since the pitch in the voiced regions is considerably larger than the minimum value, the random pitch picking should not be engaged and no effect to the voiced regions should occur. If a minimum pitch period is picked in a voiced region, it is an incorrect pitch and randomly choosing another pitch period will not harm and may improve the system performance. This variation to the pitch picking algorithm eliminates repeated minimum pitch estimations in the unvoiced regions. The problem is, therefore, solved.

For the first time, the receiver pitch extraction algorithm differed from that of the transmitter. It is noted that a forward searching correlator in the receiver has less pitch information available to it than did its counterpart in the transmitter. The larger probability of incorrectly choosing pitch at the boundaries of voiced regions results in serious distortions of the output speech. Therefore, whenever pitch is present in the compressed signal estimate $\hat{y}(k)$, it is critical that the information is extracted and used. The correlator has been modified to include a larger searching range. The beginning of this search range is located in previously expanded compressed speech. The buffer sizes in the receiver must, therefore, be larger since this previously processed speech is normally discarded. This variation led to improved performance of the previously unsatisfactory receiver serial pitch extraction algorithm. Further testing into the possibilities of reducing the search

range from 20 to 100 to a digital hardware compatible range such as 24 to 88 was performed. Multiple testing on the boundaries of the searching range of the correlator proved that no perceptable difference in speech quality accompanied the reduction of the searching range as far as 24 to 88. This searching range is, therefore, adopted in the final algorithm.

5.3 8 kHz Sampling Rate

Throughout the term of this research, studies paralleling those described for a system operating on a 6400 Hz sampled speech signal were performed on an alternate system designated for a 8000 Hz sampling rate. The 8 kHz sampling rate is a standard industry rate and repeatedly presented itself as a possible alternative to the 6400 Hz system. Eventually, the superior availability of reliable 8 kHz filters proved to be the convincing argument for the shift to an 8 kHz system.

The modifications necessary included a reduction in the number of quantization levels to $2^4 - 1 = 15$, instead of the 31 levels previously used. The 31 levels of the 6.4 kHz system produced such fine quantization that a fixed (first order) linear predictor was considered. The quality of the fixed predictor system at 6.4 kHz is identical to that of the adaptive predictor at 6.4 kHz. The reduction in available quantization levels and the goal to produce equivalent quality speech at an 8 kHz sampling rate, dictated the use of an adaptive predictor. The thresholds, scaling factors, and expansion factors are all selected using the technique that generated the 31 level values presented in Chapter IV.

Other parameters that require adjustment are the alpha used in the signal energy, the block length and searching range of the correlator, and the LPF coefficients. The alpha value reported in the previous section is the corrected value for the 8 kHz system. The searching range limits are changed to 40 to 110, which reflects nearly an 8000/6400 = 5/4 scaling factor. The 8 kHz filter with 1000 Hz cutoff freqency presented in the Appendix is the filter used in the 8 kHz sampling rate system.

5.4 Hardware Compatible ARC

The main processing integrated circuits of the proposed hardware implementation are the INTEL 8085 and NEC 7720. Although each in itself is a very versatile chip, neither supports division capabilities. This hardware consideration leads to significant modifications in the adaptive residual coder. Specifically, the adaptation of predictor coefficients and the normalization of the input to the quantizer use a division operation which must be eliminated.

The original system adapted the coefficients according to the steepest descent of $e^2(k)$ [Melsa and Cohn, 1975].

$$a_i(k+1) = \delta b_i + (1-\delta)\left[a_i(k) + \frac{g\hat{y}(k-i)e(k)}{\langle|\hat{y}(k)|\rangle^2}\right]$$

where $\langle|\hat{y}(k)|\rangle$ is a biassed exponential time average of $|\hat{y}(k)|$ [Melsa and Pande, 1981].

The division by $\langle|\hat{y}(k)|\rangle^2$ must be modified. This is accomplished by adopting a new adaptation procedure. The final algorithm uses a

fixed gain (percentage) adaptation.

$$a_i(k) = b_i + a_i^*(k)$$

$$a_i^*(k+1) = \beta(a_i^*(k) + \mathrm{sgn}\{\hat{e}(k),\hat{y}(k-i)\}\ \max\{g|a_i^*(k)|,a_{bias}\})$$

$$a_i(k+1) = b_i + a_i^*(k+1)$$

where the recommended values are

$$b_1 = 1.79$$

$$b_2 = -1.40$$

$$b_3 = 0.57$$

$$b_4 = -0.15$$

$$\beta = 0.95$$

$$a_{bias} = 0.001$$

$$g = 0.15$$

The b_i are determined using the technique suggested by R.A. McDonald (1966). The term, $\max\{\cdot\}$, computes the biased percent change desired. The a_{bias} term helps the coefficient move from the zero level. Without such a term, a coefficient which attained a zero value would remain at zero. The term, $\mathrm{sgn}\{\cdot\}$, is a first order estimate of the direction in which the coefficient should move to improve the predictor performance.

The original quantizer accepted the error sequence from the predictor and normalized these values. The normalized values were then compared to the thresholds and mapped to appropriate quantization levels,

$$e(k)_{NORM} = e(k)/\sigma(k)$$

$$e(k)_{NORM}\ \alpha\ T(i) = SF(i)$$

The division by the approximation to the standard deviation, $\sigma(k)$, must be eliminated. This division is removed by comparing the error sequence, e(k), to adaptive thresholds,

$$e(k) \alpha T(i,k) = SF(i,k)$$

$$T(i,k) = \sigma(k) \times T(i)$$

$$SF(i,k) = \sigma(k) \times SF(i)$$

These modifications resulted in no noticeable difference in output speech quality.

5.5 Finite Word Length

With any real time system a restriction on the word length is imposed. The integrated circuitry proposed for the construction of a real time system uses 16 bit words. Therefore, a software simulation which restricts all wordlengths to a maximum of 16 bits is desired. It should be noted that it is possible to increase the word accuracy to some integer multiple of 16, such as 32. This requires a much more complicated hardware algorithm since each software computation would in fact require more than one hardware computation. Keeping track of the order and location of multiple word computations further complicates such a scheme. Therefore, a 16-bit finite word length simulation is highly desired.

In fact, a 16-bit finite wordlength simulation resulted in very good quality output speech as compared to the output of the simulation having no such restriction. In all computations and representations the most significant bit is left unoccupied to give a factor of 2 margin of

error. This is done to virtually eliminate the possibility of severely distorting overflows. Reiterating, the performance of the finite word-length simulation is comparable to that of the final unrestricted word length simulation. A 16-bit wordlength simulation is, for this reason, adopted.

The speech quality as perceived by side-by-side comparisons of a taped output version of the speech and input was qualified as very good. The predictor SNR is in excess of 23 dB for sentence 1.

CHAPTER VI

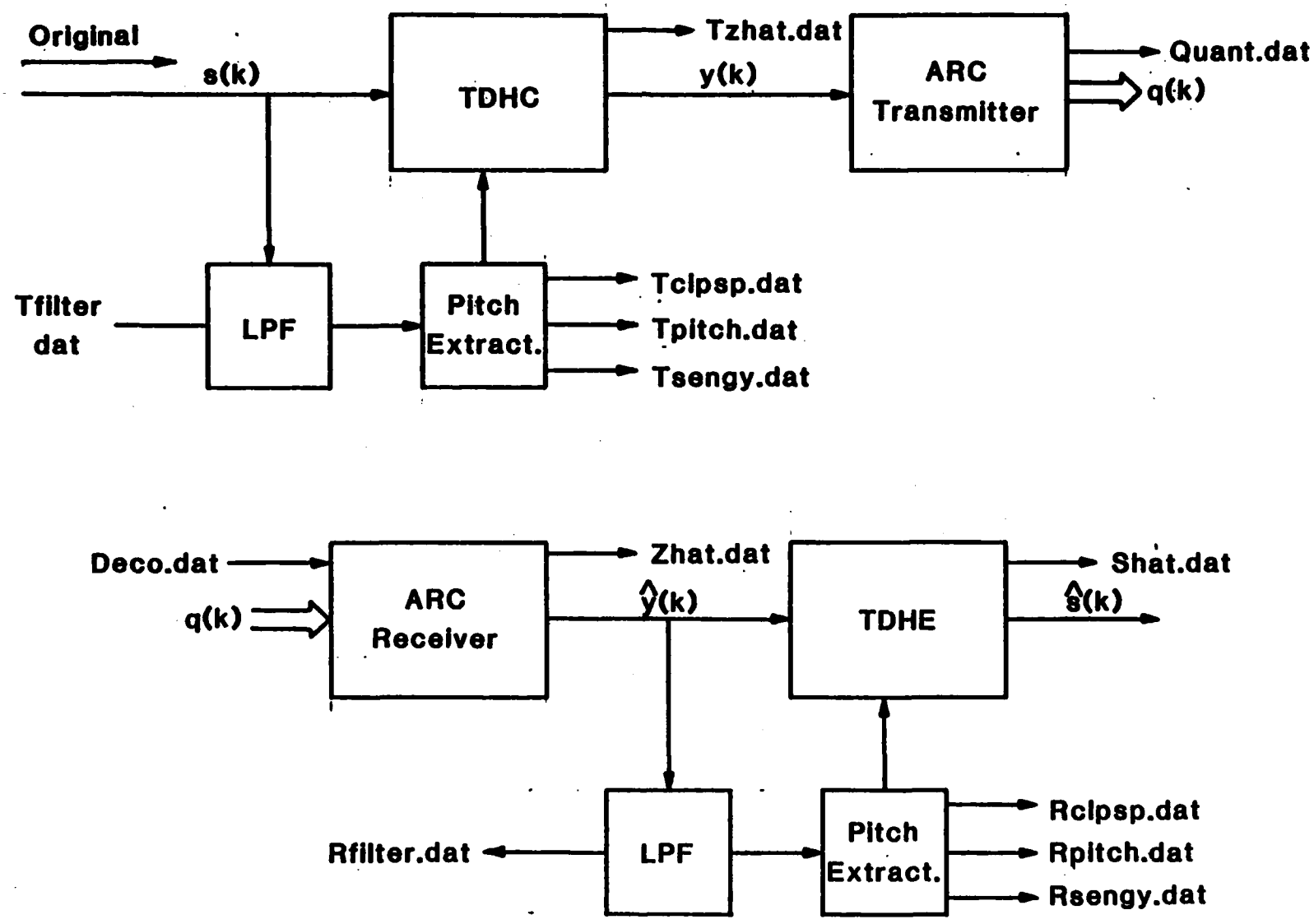
METHODOLOGY

6.1 Introduction

This chapter introduces the software capabilities and describes the associated output files of the simulation in the context of their usefulness through past research activities and for the benefit of future researchers using this software. The output files are presented first to simplify the task of explaining the program options which employ these files. They are illustrated as output files in Fig. 6.1.1. The file names will precede their individual discussion and a (T)(R) at the beginning of a file name indicates that an option exists to output this file from the transmitter and/or the receiver. If it is generated in the transmitter, the first letter of the name is a T, and if the receiver is the source, the file name begins with an R.

6.2 Files and Options

The changes incorporated in the pitch extraction algorithms resulted in the generation of a number of files which aided in the evaluation of the technique under consideration. Those associated with this topic are (T)(R) PITCH.dat, (T)(R) CLPSP.dat, (T)(R) SENGY.dat and (T)(R) FILTER.dat. (T)(R) PITCH.dat proved to be a vital file in the investigation of pitch extraction. It contains the values of pitch picked in the respective algorithm, transmitter or receiver. A simple listing of these files contains a large quantity of information. With this listing, it is possible to make a quick determination of whether the pitch is being extracted properly or not. Given a sampling rate of 8000 Hz,



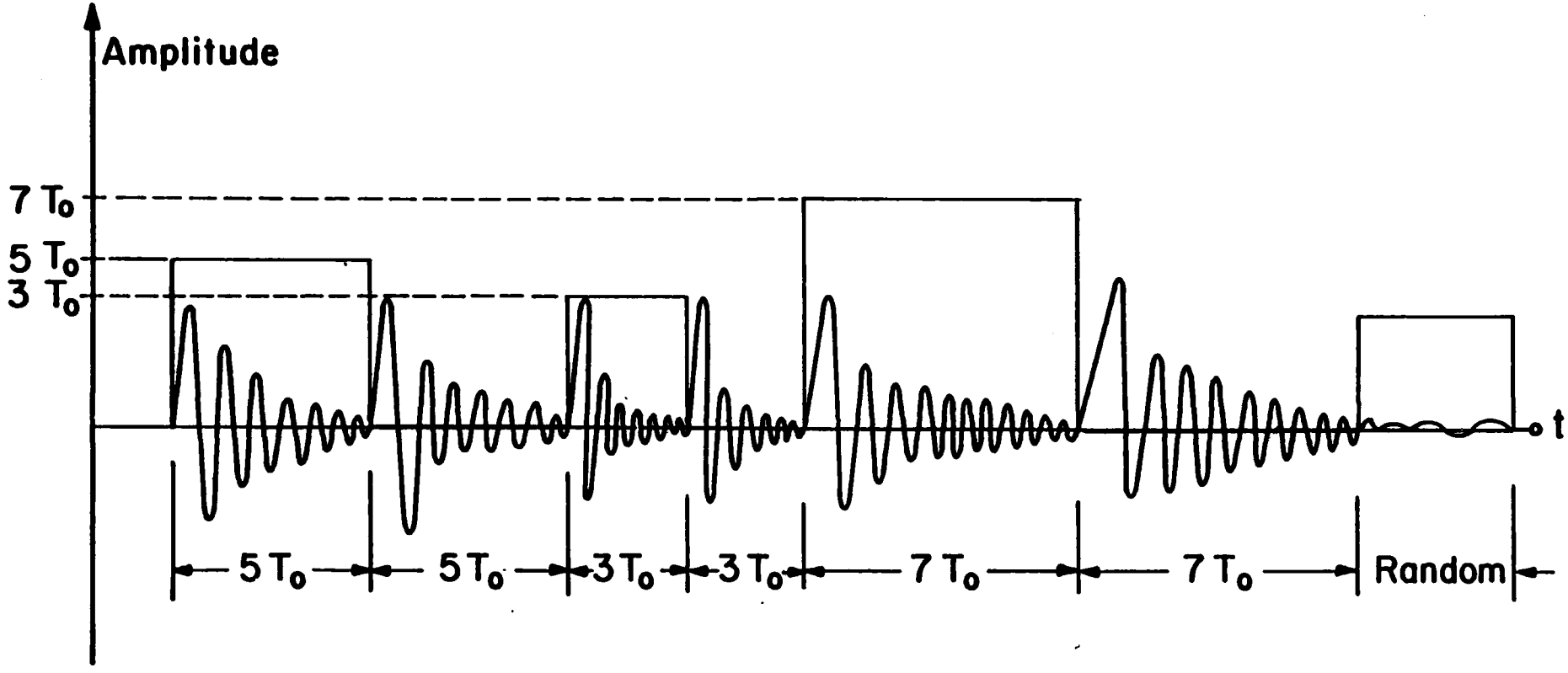
Final System Algorithm and Output Files

Figure 6.1.1

the expected pitch period length in voiced male segments is approximately 60 samples. The algorithm's agreement with this can be expeditiously evaluated from the listing. A critical condition of the pitch extraction algorithm is that it should choose the same pitch period in the receiver as it chose in the transmitter. Side-by-side comparison of the files will determine if this condition is fulfilled.

PITCH.dat illustrated the periodicity of pitch picked in silence regions by the serial pitch extraction algorithm and precipitated the implementation of the random pitch picker. PITCH.dat forms the input of the program titled PICHMAK.ftn which transforms the pitch periods into a series of rectangular pulses of height equal to the pitch period and duration equal to twice the pitch period. The output of this file is compatible input for the plotting routine used throughout the research called DATAPLOT.ftn. The usefulness of the hard copy output is that it can be laid upon a speech segment giving a pictorial account of the pitch extraction algorithm's performance. This is illustrated in Fig. 6.2.1. Although this is a more precise and reliable report of the effectiveness of the pitch extraction routine, the cost, in time, required to spool the results to the plotter, necessitates the use of the pitch period lists in multiple run testing.

<u>(T)(R)CLPSP.dat</u> is a list of the three value center clipped speech. It is also compatible with the plotting program, the output of which determined the potency of the serial clipping routine. The expected output of such a clipping routine, with speech as an input, is a series of positive one's occurring at the large positive going section of the pitch pulses and negative one's for their negative going counter parts.



GRAPHICAL PITCH EXTRACTION

Figure 6.2.1

Zeros are expected to fill in the gaps. Laying this on top of the input speech file is, again, effective in ascertaining the clipping action. The time associated with obtaining the hard copy should be evaluated before performing a multiple run evaluation. This file illustrated the proximal relationship between the block pitch extraction's clipping routine and that of the serial pitch extractor.

(T)(R)SENGY.dat is a listing of filtered speech signal energy (compressed speech, in the receiver). Used with the plotting program, this file demonstrated the effectiveness of the signal energy computation. It is necessary to compare the signal energy plot to the output of the filter and not to the original speech, since clipping will be performed on the filtered speech. Comparison of the signal energy to the filtered speech assisted in calibrating the clipping value, CLPP. CLPP is the scaling value of the signal energy; the product being the instantaneous clipping level.

(T)(R)FILTER.dat is that file whose hard copy is compared to the signal energy computation as discussed in the previous paragraph. It is the low pass filter output of the pitch extraction units. It's plot verified the filter's smoothing effect on the input speech.

The combination of these files depicted the performance of each component of the pitch extraction unit. Their aid was unparalleled in the modification of the TDHS algorithms and it is suggested that they continue to be used in the evaluation of any further simulation adjustments.

TZHAT.dat. (simply, ZHAT.dat. in the transmitter) contain the compressed speech files. A side-by-side listening session comparing a

taped version of these files, as well as, the examination of their hard copy plots greatly contributed in determining the potency of the TDHS units. The hard copy is expected to be a plot highly related to the original, whose time axis is scaled according to the compression ratio adopted. The taped version should reflect this same time axis scaling.

QUANT.dat and DECO.dat are the input and output of the channel, respectively. They are sequences of quantization levels and are equivalent when the noiseless channel approximation is engaged. When a noisy channel is employed, DECO.dat is the error infected version of QUANT.dat.

TSTATS.dat is a file output by the transmitter only. It contains a presentation of the system performance criteria employed in the original system. TSTATS.dat includes items such as the average signal-to-noise ratio of the predictor and the parameters used in the quantizer. A sample of this file is contained in Appendix E. Since the different SNR measurements pertain to the transmitter unit, this file was primarily commissioned in the ARC evaluations.

SHAT.dat contains the simulation's estimate of the original speech file. The taped reproduction of this formed the most highly weighted estimation of the system's effectiveness. Frequently, the quality of the output file, as perceived by listeners, was the only criteria used in decisions. This file, therefore, formed the most important measure of the system's performance.

The program options are illustrated in Fig. 6.2.2. There are four major options available to the user. The simulations available are an algorithm that exercises the TDHS unit only (without the ARC) and one that concatenates the TDHS and ARC blocks. Each of these options has the

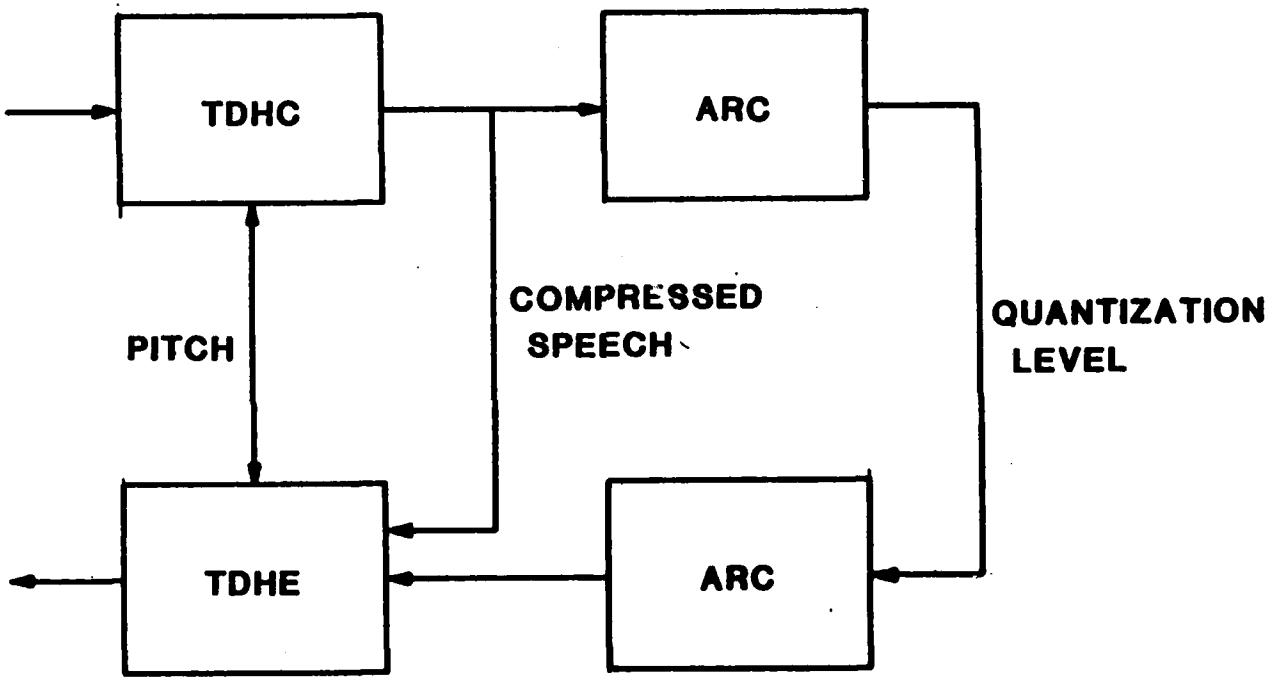


Figure 6.2.2 Algorithm Capabilities

option of passing the pitch picked in the transmitter, as side information, to the receiver. No cost to channel capacity was evaluated for the separate options with the exception of the option of TDHS-ARC with no pitch passed. This option represents the final algorithm. The separate options assisted in assessing the noise due to the TDHS algorithm, the ARC blocks, and the incorrect pitch in the receiver. The comparisons of the different options demonstrated the magnitude of each of these noise sources.

The source files for the unrestricted word length simulations are presented in Appendix A. Reference should be made to these when deciphering Appendix B, the finite word length simulations' source files. Both sets of source files are driven and fused by the included indirect command file. Appendix C contains the source files of supporting programs which aided in the development process.

CHAPTER VII

CONCLUSIONS

The final system incorporates all of the specifications of the Chapter I and forms a valid simulation of a finite word length time domain harmonic scaling-adaptive residual coder system. The final predictor SNR is in excess of 22 dB for the sentence

"Cats and dogs each hate the other."

The critical topics for those who are familiar with TDHS-ARC system are contained in Chapters III, IV, and V.

The issue herein reported still maintains questions unanswered and forms possible topics of future investigation. The optimization of the technique used to distribute scaling factors, thresholds, and expansion factors is one example. It seems that the maxima of these sets of parameters could be reduced to the extent of precipitating added accuracy in the finite word length simulations. An investigation into the removal of the random pitch generator is pressed by the effectiveness of the modified correlator. Random pitch is picked few times in the transmitter, usually at the beginning or end of the sentence and rarely is random pitch picked successively. The receiver results are similar. For the above sentence, the receiver picked successive random pitch only once. It does not seem to be worthy of the added software complexity to implement this technique. Investigations into the difference of parameter values of the receiver as compared to the transmitter need to be accomplished (such as, the time constant of the clipping signal energy computation).

The final algorithm incorporates the suggested TDHS-ARC structure with modifications to the original generated by the elements of the original proposal and by hardware restrictions. For the TDHS modifications, emphasis was placed on finding a hardware compatible pitch detector. The new adaptation scheme is the major variation found in the ARC. The final hardware simulations resulted in good quality speech in the presence of up to 0.1% BER, ascertained by side-by-side comparison of a taped reproduction.

The goal, to design a simulation by means of algorithm refinements that is easy to implement in hardware and yet does not degrade the present performance, has been met.

APPENDIX A

FINITE WORD LENGTH SIMULATION SOURCE FILES

The necessary modules to run a simulation are:

1) CMD16.CMD (Indirect command file)

2) PFWL.FTN (Parameter builder)

3) TFWL.FTN (Transmitter)

4) C8K.FTN (Channel)

5) RFWL.FTN (Receiver)

The task transmitter and receiver modules should be built from the source files using the following switches:

```
> UNITS   = 15
> ACTFILE = 15
```

All other source files use the default values. A specific file management procedure is dictated by the indirect command file. This can easily be modified to fit the specific needs of a multiple run use.

```
;
;
;
;
;       ***********************************************
;       ***********************************************
;       ***                                         ***
;       ***                                         ***
;       ***          C M D 1 6 F . C M D            ***
;       ***                                         ***
;       ***                                         ***
;       ***********************************************
;       ***********************************************
;
;
;*****************************************************************
;*****************************************************************
;
;
;     THIS COMMAND FILE IS DESIGNED TO RUN IN THE ABSENCE OF THE OPERATOR
;     THE PROGRAMS NECESSARY TO RUN THIS ARE:
;             1) P16.TSK
;             2) T16.TSK
;             3) R16.TSK
;             4) PICHMAK.TSK
;             5) C8K.TSK (REQUIRES OPERATOR MONITOR)
;     AND INPUT
;             6) E.G., S1I58#.DAT
;     NOTE THAT EARLIER VERSIONS OF THE TRANSMITTER AND RECEIVER ARE
;     NOT COMPATIBLE WITH THIS COMMAND FILE SINCE THEY REQUIRE THAT THE
;     OPERATOR MONITORS THEM FOR INTERACTIVE DATA.
;
;
;
      .ENABLE SUBSTITUTION
      .ENABLE LOWERCASE
;
;
      .SETN C1N #
      .SETN S2N #
      .SETN S1N #
      .SETN NUMFIL #
      .SETS NUM "123456789"
      .ASK TOUT DO YOU WANT TO RUN TRANSMITTER PITCH COMPARISON
      .IFT TOUT .SETN S1N 1
      .ASK ROUT DO YOU WANT TO RUN RECEIVER PITCH COMPARISON
      .IFT ROUT .SETN S2N 1
      .ASK CHN DO YOU WANT TO USE A CHANNEL
      .IFT CNN .SETN C1N 1
      .INC NUMFIL
;THIS RUN IS FOR THE TRANSMITTER OPTION FILE
      RUN P16
```

```
        .WAIT
        PIP TOPT.DAT=PARAM.DAT/RE
.1#:    .ASK Q1 DO YOU WANT THE RCVR OPT FILE=TRAN OPT FILE
        .IFF Q1 .GOTO 3#
        .SETN NEWNUM NUMFIL
        .SETS NEWDAT NUM[NEWNUM:NEWNUM]
        .SETS NEWNAM "ROPT.DAT;"+NEWDAT
        .SETS OLDNAM "TOPT.DAT;"+NEWDAT
        PIP 'NEWNAM'='OLDNAM'
        .WAIT PIP
        .GOTO 4#
.3#:    .WAIT
;THIS RUN IS FOR THE RECEIVER OPTION FILE
        RUN P16
        .WAIT
        .SETN NEWNUM NUMFIL
        .SETS NEWDAT NUM[NEWNUM:NEWNUM]
        .SETS NEWNAM "ROPT.DAT;"+NEWDAT
        PIP 'NEWNAM'=PARAM.DAT/RE
.4#:    .WAIT PIP
.5#:    .ASK Q2 DO YOU WANT ANOTHER RUN
        .IFF Q2 .GOTO 7#
        .INC NUMFIL
        .ASK Q3 DO YOU WANT THE SAME TOPT FILE AS THE PREVIOUS ONE
        .IFF Q3 .GOTO 6#
        .SETN NEWNUM NUMFIL
        .SETN OLDNUM NUMFIL-1
        .SETS NEWDAT NUM[NEWNUM:NEWNUM]
        .SETS OLDDAT NUM[OLDNUM:OLDNUM]
        .SETS NEWNAM "TOPT.DAT;"+NEWDAT
        .SETS OLDNAM "TOPT.DAT;"+OLDDAT
        PIP 'NEWNAM'='OLDNAM'
        .WAIT PIP
        .GOTO 1#
.6#:    .WAIT
;THIS RUN IS FOR THE TRANSMITTER OPTION FILE
        RUN P16
        .WAIT
        .SETS NEWDAT NUM[NUMFIL:NUMFIL]
        .SETS NEWNAM "TOPT.DAT;"+NEWDAT
        PIP 'NEWNAM'=PARAM.DAT/RE
        .WAIT PIP
        .ASK Q4 DO YOU WANT THE SAME ROPT FILE AS THE PREVIOUS ONE
        .IFF Q4 .GOTO 1#
        .SETN OLDNUM NUMFIL-1
        .SETS OLDDAT NUM[OLDNUM:OLDNUM]
        .SETS NEWNAM "ROPT.DAT;"+NEWDAT
        .SETS OLDNAM "ROPT.DAT;"+OLDDAT
        PIP 'NEWNAM'='OLDNAM'
        .WAIT PIP
        .GOTO 5#
.7#:    .WAIT
```

```
        .SETN ARG 1
.8Ø:    .SETS NDAT NUM[ARG:ARG]
        .SETS NAME "TOPT.DAT;"+NDAT
        PIP CMDPARA.DAT='NAME'
        .WAIT PIP
        RUN T16
        .WAIT
        .IF S1N NE 1 .GOTO 1ØØ
        PIP PITCH.DAT=TPITCH.DAT
        .WAIT PIP
        RUN PICHMAK
        .WAIT
        PIP PITCH.DAT;*/DE
        .WAIT PIP
        .SETS TPP "TPP.DAT;"+NDAT
        PIP 'TPP'=PPULSE.DAT;1/RE
.1ØØ:   .WAIT PIP
        .IF C1N NE 1 .GOTO 9Ø
        RUN C8K
.9Ø:    .WAIT
        .IF C1N EQ 1 .GOTO 95
        PIP DECO.DAT=QUANT.DAT;*/RE
        .WAIT PIP
.95:    .WAIT
        .SETS NAME "ROPT.DAT;"+NDAT
        PIP CMDPARA.DAT='NAME'
        .WAIT PIP
        RUN R16
        .WAIT
        PIP DECO.DAT;*,CMDPARA.DAT;*/DE
        .WAIT PIP
        .IF S2N NE 1 .GOTO 12Ø
        PIP PITCH.DAT=RPITCH.DAT
        .WAIT PIP
        RUN PICHMAK
        .WAIT
        .SETS RPP "RPP.DAT;"+NDAT
        PIP 'RPP'=PPULSE.DAT;1/RE
        .WAIT PIP
        PIP PITCH.DAT;*/DE
.12Ø:   .WAIT PIP
        .INC ARG
        .IF ARG GT NUMFIL .GOTO 13Ø
        .GOTO 8Ø
.13Ø:   .WAIT 1Ø.S
```

```
C
C
C
C
C
C     ******************************************
C     ******************************************
C     ***                                    ***
C     ***                                    ***
C     ***                                    ***
C     ***            P F W L . F T N         ***
C     ***                                    ***
C     ***                                    ***
C     ***                                    ***
C     ******************************************
C     ******************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE: PFWL.FTN
C
C     BY: JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTIVE TITLE: PARAMETER BUILDER FOR FWL SIMULATIONS
C
C     DESCRIPTION:  THIS PROGRAM GENERATES A PARAMETER FILE TO
C             BE USED WITH THE FINITE WORD LENGTH SIMULATIONS.
C             IT OUTPUTS A FILE NAMED PARAM.DAT.
C             THE FOLLOWING ARE INTERACTIVELY SELECTED:
C                     1) ORIGINAL SPEECH
C                     2) ALPHA USED IN SIGNAL ENERGY COMPUTATION
C                            FOR CLIPPING ROUTINE
C                     3) ALP=PREDICTOR RMS COMPUTING VALUE
C                     4) ALAD=PREDICTOR COEFFICIENT UPDATING
C                            FACTOR
C                     5) G=PREDICTOR COEFFICIENT UPDATING FACTOR
C                     6) N=ORDER OF THE PREDICTOR
C                     7) RMSMIN=MINIMUM RMS VALUE OF PREDICTOR
C                     8) SMIN=RMS SCALING FACTOR OF QUANTIZER
C                     9) CLPP=SIGNAL ENERGY SCALING FACTOR
C                    10) KBLK1=SEARCH LENGTH OF CORRELATOR
C                    11) ITMIN, ITMAX=MINIMUM AND MAXIMUM PITCH
C                            ALLOWED BY THE PITCH EXTRACTOR
C                    12) MSF=MAXIMUM SCALING FACTOR
C                    13) KQ=NUMBER OF QUANTIZATION LEVELS
C                    14) B=EXPANSION BASE FOR THE DETERMINATION
C                            OF SCALING FACTORS
C                    15) LOEXPN, HIEXPN=LOWEST AND HIGHEST
C                            EXPANSION FACTORS OF THE QUANTI-
C                            ZER STANDARD DEVIATION ESTIMATOR
```

```
C                    17) INF=INPUT SPEECH SCALING FACTOR
C                    18) ICF=CLIPPING FACTOR FOR FWL SIM.
C                    19) IAF=PREDICTOR COEFFICIENT FACTOR FOR FWL SIM.
C                    20) IQ1F=FWL SCALING FACTOR FOR PREDICTOR THRESH.
C                    21) IQ2F=EXPANSION FACTOR SCALING FOR FWL. SIM.
C                    22) IQ3F=FWL SCALING FACTOR FOR STANDARD DEV.
C                    23) THE USE OF A FILTER AND THE THE CHOICE OF
C                          THE PARTICULAR FILTER
C                    24) THE PROGARAM PATH INCLUDING A PITCH PASS
C                          OPTION
C
C*****************************************************************
C*****************************************************************
C
C
      INTEGER WARE,UU,VV,ANS1(6),CHK,ANS2(3),ANS(9),ANS3(8),ANS4(6)
      INTEGER*2 FNAME(16)
      REAL MSF,LFI(51),LMX,LOEXPN,HIEXPN,HIDELT,LODELT,EXPN(51),M
      REAL OUT(51),KK,LL,T(51),HF(5),H(5,4),AB(12)
C***************
C***************THE FOLLOWING DATA IS THE FILTER COEFFICIENTS FOR
C***************THE FILTERS THAT ARE INTERACTIVELY EMPLOYED BY
C***************THE SIMULATIONS.
C***************
      DATA H/.0341,.0125,.0139,.0149,.0151,.0560,.0720,.1009,
     1        .1219,.1295,.0376,.0214,.0252,.0277,.0285,.0758,.1549,.2344,
     2        .3188,.3356/
C***************
C***************PARAM.DAT IS THE OUTPUT FILE: CONTAINS THE PARAMETERS
C***************FOR THE  SIMULATIONS
C***************
      OPEN(UNIT=3,TYPE='NEW',NAME='PARAM.DAT',CARRIAGECONTROL='LIST')
      TYPE *,' ENTER THE ORIGINAL SPEECH FILE NAME:[<32CHAR]'
      ACCEPT 9,FNAME
    9 FORMAT(16A2)
C*****************************************************************
C*****************************************************************
C***************
C***************INITITALIZATION OF ALL PROGRAM PARAMETERS.
C***************
      NBL=64
      ALPHA=.992
      ABIAS=0.01
      AB(1)=1.793182
      AB(2)=-1.400654
      AB(3)=0.566023
      AB(4)=-0.146575
      DO 53 I=5,12
              AB(I)=0.0
   53 CONTINUE
      GAIN=0.950
      ALP=0.97
```

```
      ALAD=0.95
      G=0.015
      N=4
      RMSMIN=35.0
      SMIN=0.3
      MSF=8.0
      KQ=15
      B=9.
      LOEXPN=0.5
      HIEXPN=5.0
      CLPP=1.5
      KBLK1=110
      ITMIN=30
      ITMAX=110
      ITCS=0
      ITPF=0
      ITSE=0
      ITSF=0
      IRCS=0
      IRPF=0
      IRSE=0
      ICSF=0
      ITFF=0
      IRFF=0
      IOSF=1
      ISF=15
      INF=3
      IAF=13
      IQ3F=4
C*********************************************************************
C*********************************************************************
C***************
C***************THE FOLLOWING BLOCK SETS UP THE LISTED PARAMETERS
C***************
  790 DO 795 I=1,9
            ANS(I)=0
  795 CONTINUE
      TYPE *,' THE FOLLOWING ARE THE CURRENT VALUES'
      WRITE(5,100)ALP,ABIAS,ALPHA,G,N,RMSMIN,SMIN,CLPP,KBLK1,ITMIN,
     1        ITMAX,ALAD,NBL
  100 FORMAT(/6X,'ALP=',F5.3/6X,'ABIAS=',F5.3/6X,'ALPHA=',F5.3/6X,'G='
     1        ,F5.3/6X,'N=',I2/6X,'RMSMIN=',F5.1/6X,'SMIN=',F5.2/6X,'CLPP='
     2        ,F5.2/6X,'KBLK1=',I3,2X,'ITMIN=',I3,2X,'ITMAX=',I3/6X,'ALAD='
     3        ,F5.3/6X,'NBL=',I4/)
      TYPE *,' WHICH DO YOU WANT TO CHANGE?'
      TYPE *,'                    0=NO (MORE) CHANGE(S)'
      TYPE *,'                    1=ALP'
      TYPE *,'                    2=ABIAS'
      TYPE *,'                    3=ALPHA'
      TYPE *,'                    4=G'
      TYPE *,'                    5=N'
      TYPE *,'                    6=RMSMIN'
```

```
      TYPE *,'                    7=SMIN'
      TYPE *,'                    8=CLPP'
      TYPE *,'                    9=KBLK1,ITMIN,ITMAX'
      TYPE *,' TYPE--[SELECTION] (SELECTION) (SELECTION)...--UP TO 9 CHOICES'
      ACCEPT 780,ANS
  780 FORMAT(9(I1,1X))
      CHK=0
      DO 770 I=1,9
              CHK=CHK+ANS(I)
  770 CONTINUE
      IF(CHK.EQ.0)GO TO 900
      ISELEC=1
  197 DO 199 I=1,9
              IF(ANS(I).EQ.ISELEC)GO TO (201,202,203,204,205,206,207,208
     1                            ,209),ISELEC
  199 CONTINUE
      GO TO 193
  201 TYPE *,' ENTER NEW VALUE OF "ALP"'
      ACCEPT *,ALP
      GO TO 193
  202 TYPE *,' ENTER NEW VALUE OF "ABIAS"'
      ACCEPT *,ABIAS
      GO TO 193
  203 TYPE *,' ENTER NEW VALUE OF "ALPHA"'
      ACCEPT *,ALPHA
      GO TO 193
  204 TYPE *,' ENTER NEW VALUE OF "G"'
      ACCEPT *,G
      GO TO 193
  205 TYPE *,' ENTER NEW VALUE OF "N"'
      ACCEPT *,N
      GO TO 193
  206 TYPE *,' ENTER NEW VALUE OF "RMSMIN"'
      ACCEPT *,RMSMIN
      GO TO 193
  207 TYPE *,' ENTER NEW VALUE OF "SMIN"'
      ACCEPT *,SMIN
      GO TO 193
  208 TYPE *,' ENTER NEW VALUE OF "CLPP"'
      ACCEPT *,CLPP
      GO TO 193
  209 TYPE *,' ENTER NEW VALUES OF "KBLK1 ITMIN ITMAX"'
      ACCEPT *,KBLK1,ITMIN,ITMAX
  193 ISELEC=ISELEC+1
      IF(ISELEC.LE.9)GO TO 197
      GO TO 790
  900 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************THE FOLLOWING BLOCK SETS UPT THE PARAMETERS OF THE
C***************ADAPTIVE RESIDUAL CODER.  IT GENERATES THE THRESH-
```

```
C***************OLDS, SCALING FACTORS, AND EXPANSION FACTORS USING
C***************THE PARAMETERS SPECIFIED BY THE USER.
C***************
   12 DO 17 I=1,6
              ANS1(I)=Ø
   17 CONTINUE
      DO 18 I=1,6
              IF(ANS1(I).EQ.1)GO TO 3
   18 CONTINUE
      GO TO 1Ø
    3 TYPE *,'ENTER THE MAXIMUM SCALING FACTOR(>Ø)'
      ACCEPT *,MSF
      IF (MSF.LE.Ø.)GO TO 3
   1Ø DO 19 I=1,6
              IF(ANS1(I).EQ.2) GO TO 29
   19 CONTINUE
      GO TO 2Ø
   29 TYPE *,'ENTER THE NUMBER OF QUANT LEVELS (5<NUM<1Ø1) AND ODD'
      ACCEPT *,KQ
      IF (KQ.LT.5.OR.KQ.GT.1Ø1)GO TO 1Ø
   2Ø NQ=KQ/2
      NQQQ=NQ+1
C***************
C***************DETERMINATION OF THE QUATIZER PARAMETER VALUES
C***************FIRST, THE SCALING FACTORS ARE GENERATED.
C***************
      OUT(NQQQ)=MSF
      LMX=(ALOG1Ø(MSF))
      UP=LMX/(FLOAT(NQ))
      LFI(1)=Ø.
      DO 4Ø I=2,NQ
              LFI(I)=LFI(I-1)+UP
   4Ø CONTINUE
      KK=MSF+1
      KK=ALOG1Ø(KK)
      LL=LFI(NQ)
      KK=KK/LL
      BMAX=1Ø.**KK
      DO 214 I=1,6
              IF(ANS1(I).EQ.3)GO TO 213
  214 CONTINUE
      GO TO 42
  213 WRITE(5,212)BMAX,B
  212 FORMAT(' THE MAX BASE ALLOWED IS ',F5.2,' AND THE CURRENT BASE '
     1        ,'IS ',F5.2)
      TYPE *,' ENTER BASE OF LOG FNC USED FOR SCALING FACTORS(>Ø)'
      ACCEPT *,B
   42 IF(B.GT.Ø.AND.B.LT.BMAX)GO TO 43
      TYPE *,' THE BASE CHOSEN WAS OUTSIDE THE ALLOWED RANGE. TRY AGAIN!'
      GO TO 213
   43 DO 5Ø I=1,NQ
              M=LFI(I)
```

```
              OUT(I)=B**M-1.
   50 CONTINUE
      DO 60 I=1,NQ
              T(I)=(OUT(I)+OUT(I+1))/2.
   60 CONTINUE
      DO 61 I=1,6
              IF(ANS1(I).EQ.4)GO TO 62
   61 CONTINUE
      GO TO 65
C***************DETERMINATION OF EXPANSION FACTORS.
   62 TYPE *,' ENTER THE LOW EXPANSION FACTOR'
      ACCEPT *,LOEXPN
   65 DO 66 I=1,6
              IF(ANS1(I).EQ.5)GO TO 67
   66 CONTINUE
      GO TO 75
   67 TYPE *,' ENTER THE HIGH EXPANSION FACTOR'
      ACCEPT *,HIEXPN
   75 MID=NQ/2+1
      EXPN(MID)=1.
      EXPN(MID+1)=1.
      EXPN(MID-1)=1.
      EXPN(1)=LOEXPN
      EXPN(NQQQ)=HIEXPN
      HIDELT=(HIEXPN-1.)/(FLOAT(NQ-MID+1))
      LODELT=(1.-LOEXPN)/(FLOAT(MID-2))
      XX=HIDELT+1.
      YY=LODELT+LOEXPN
      UU=MID-2
      VV=MID+2
      DO 110 I=2,UU
              EXPN(I)=YY
              YY=YY+LODELT
  110 CONTINUE
      DO 120 I=VV,NQ
              EXPN(I)=XX
              XX=XX+HIDELT
  120 CONTINUE
C***************INTERMEDIATE PRINTOUT OF QUANTIZER PARAMETERS.
      TYPE *,'WHERE SHOULD THE PRINTOUT GO?[TERM=0,PRINT=1]'
      ACCEPT *,I
      IF (I.EQ.0) WARE=5
      IF (I.EQ.1) WARE=6
      WRITE(5,100)ALP,ABIAS,ALPHA,G,N,RMSMIN,SMIN,CLPP,KBLK1,ITMIN,
     1        ITMAX,ALAD,NBL
      IF (I.EQ.1)WRITE(5,160)KQ,NQQQ,MSF,B,LOEXPN,HIEXPN
      WRITE(WARE,160)KQ,NQQQ,MSF,B,LOEXPN,HIEXPN
  160 FORMAT(//' THE FOLLOWING IS A LIST OF SCALING FACTORS,'/
     1        ' EXPANSION FACTORS, AND THRESHOLDS FOR A ',I2,' LEVEL'/
     2        ' QUANTIZER(',I2,' ON A SIDE), WITH A MAXIMUM SCALING FACTOR'/
     3        ' =',F7.4,'.  A LOG SCALE WITH A BASE=',F7.4,'WAS USED.'/
     4        ' THE HIGH AND LOW EXPANSION FACTORS WERE SET AT ',F7.4/
```

```
    5  ' AND',F7.4,' .')
     WRITE(WARE,180)
     DO 190 I=1,NQQQ
            WRITE(WARE,200)I,OUT(I),T(I),EXPN(I)
  190 CONTINUE
  180 FORMAT(/3X,'LEVEL NUMBER',4X,'SCALING FACTORS',4X,'THRESHOLDS',
     1        4X,'EXPANSION FACTORS'/)
  200 FORMAT(6X,I2,17X,F7.4,9X,F7.4,10X,F7.4)
     TYPE *,'DO YOU WANT TO CHANGE ANY OF THESE?[YES=1,NO=0]'
     ACCEPT *,I
     IF(I.EQ.0)GO TO 955
     TYPE *,' THE FOLLOWING ARE THE CURRENT VALUES'
     WRITE(5,111)MSF,KQ,B,LOEXPN,HIEXPN
  111 FORMAT(/6X,'MSF=',F5.2/6X,'KQ=',I2/6X,'B=',F5.2/6X,'LOEXPN=',F5.2
     1        /6X,'HIEXPN=',F5.2/)
     TYPE *,' WHICH DO YOU WANT TO CHANGE?'
     TYPE *,'        0=NO (MORE) CHANGE(S)'
     TYPE *,'        1=MSF'
     TYPE *,'        2=KQ'
     TYPE *,'        3=B'
     TYPE *,'        4=LOEXPN'
     TYPE *,'        5=HIEXPN'
     TYPE *,' TYPE--[SELECTION] (SELECTION) (SELECTION)...--UP TO 6 CHOICES'
     ACCEPT 59,ANS1
   59 FORMAT(6(I1,1X))
C*****************************************************************
C*****************************************************************
C***************
C***************OUTPUT FILES ARE CHOSEN.
C***************
     CHK=0
     DO 221 I=1,6
            CHK=CHK+ANS1(I)
  221 CONTINUE
     IF(CHK.EQ.0)GO TO 955
     GO TO 17
  955 CONTINUE
  423 DO 426 I=1,3
            ANS2(I)=0
  426 CONTINUE
     TYPE *,' THE FOLLOWING INDICATE WHICH SPEECH AND NONPRINTABLE FILES'
     TYPE *,' THIS COMMAND FILE WILL OUTPUT [0=NOT OUTPUT, 1=WILL BE OUTPUT'
     TYPE 500
     TYPE 510,ITSF
     TYPE 520,ICSF
     TYPE 530,IOSF
  500 FORMAT('        0)NO CHANGES DESIRED')
  510 FORMAT('        1)TRANSMITTER STATISTICS FILE        SET=',I1)
  520 FORMAT('        2)COMPRESSED SPEECH FILE             SET=',I1)
  530 FORMAT('        3)OUTPUT SPEECH FILE                 SET=',I1)
     TYPE *,' TO CHANGE ANY OR ALL OF THESE TYPE---'
     TYPE *,' [SELECTION] (SELECTION) (SELECTION)...--UP TO 3 CHOICES'
```

```
      ACCEPT 535,ANS2
  535 FORMAT(3(I1,1X))
      CHK=0
      DO 540 I=1,3
              CHK=CHK+ANS2(I)
  540 CONTINUE
      IF(CHK.EQ.0)GO TO 575
      IFILE=ITSF
      ISELEC=1
  545 DO 550 I=1,3
              IF(ANS2(I).EQ.ISELEC)GO TO 555
  550 CONTINUE
      GO TO 570
  555 IF(IFILE.NE.0)GO TO 560
      IFILE=1
      GO TO 565
  560 IFILE=0
  565 IF(ISELEC.EQ.1)ITSF=IFILE
      IF(ISELEC.EQ.2)ICSF=IFILE
      IF(ISELEC.EQ.3)IOSF=IFILE
  570 ISELEC=ISELEC+1
      IF(ISELEC.EQ.2)IFILE=ICSF
      IF(ISELEC.EQ.3)IFILE=IOSF
      IF(ISELEC.LE.3)GO TO 545
      GO TO 423
  575 CONTINUE
  580 DO 585 I=1,8
              ANS3(I)=0
  585 CONTINUE
      TYPE *,' THE FOLLOWING INDICATE WHICH PLOTTABLE FILES WILL BE'
      TYPE *,' OUTPUT BY THIS COMMAND FILE'
      TYPE *,' [0=NOT OUTPUT, 1=WILL BE OUTPUT]'
      TYPE 500
      TYPE 590,ITFF
      TYPE 595,ITSE
      TYPE 600,ITCS
      TYPE 605,ITPF
      TYPE 610,IRFF
      TYPE 615,IRSE
      TYPE 620,IRCS
      TYPE 625,IRPF
  590 FORMAT('          1)TRANSMITTER FILTERED SPEECH    SET=',I1)
  595 FORMAT('          2)TRANSMITTER SIGNAL ENERGY      SET=',I1)
  600 FORMAT('          3)TRANSMITTER CLIPPED SPEECH     SET=',I1)
  605 FORMAT('          4)TRANSMITTER PITCH FILE         SET=',I1)
  610 FORMAT('          5)RECEIVER FILTERED SPEECH       SET=',I1)
  615 FORMAT('          6)RECEIVER SIGNAL ENERGY         SET=',I1)
  620 FORMAT('          7)RECEIVER CLIPPED SPEECH        SET=',I1)
  625 FORMAT('          8)RECEIVER PITCH FILE            SET=',I1)
      TYPE *,' TO CHANGE ANY OR ALL OF THESE TYPE---'
      TYPE *,' [SELECTION] (SELECTION) (SELECTION)...--UP TO 8 CHOICES'
      ACCEPT 630,ANS3
```

```
630 FORMAT(8(I1,1X))
    CHK=0
    DO 635 I=1,8
            CHK=CHK+ANS3(I)
635 CONTINUE
    IF(CHK.EQ.0)GO TO 685
    IFILE=ITFF
    ISELEC=1
640 DO 645 I=1,8
            IF(ANS3(I).EQ.ISELEC)GO TO 650
645 CONTINUE
    GO TO 675
650 IF(IFILE.NE.0)GO TO 655
    IFILE=1
    GO TO 665
655 IFILE=0
665 IF(ISELEC.EQ.1)ITFF=IFILE
    IF(ISELEC.EQ.2)ITSE=IFILE
    IF(ISELEC.EQ.3)ITCS=IFILE
    IF(ISELEC.EQ.4)ITPF=IFILE
    IF(ISELEC.EQ.5)IRFF=IFILE
    IF(ISELEC.EQ.6)IRSE=IFILE
    IF(ISELEC.EQ.7)IRCS=IFILE
    IF(ISELEC.EQ.8)IRPF=IFILE
675 ISELEC=ISELEC+1
    IF(ISELEC.EQ.2)IFILE=ITSE
    IF(ISELEC.EQ.3)IFILE=ITCS
    IF(ISELEC.EQ.4)IFILE=ITPF
    IF(ISELEC.EQ.5)IFILE=IRFF
    IF(ISELEC.EQ.6)IFILE=IRSE
    IF(ISELEC.EQ.7)IFILE=IRCS
    IF(ISELEC.EQ.8)IFILE=IRPF
    IF(ISELEC.LE.8)GO TO 640
    GO TO 580
685 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************DETERMINATION OF FINITE WORD LENGTH SIMULATION
C***************SCALING FACTORS.
C***************
      ICF=ISF-1-IFIX(ALOG10(CLPP)/ALOG10(2.)+1.)
      IQ1F=ISF-1-IFIX(ALOG10(MSF)/ALOG10(2.)+1.)
      IQ2F=ISF-1-IFIX(ALOG10(HIEXPN)/ALOG10(2.)+1.)
      TYPE *,' THE FOLLOWING ARE THE BIT SHIFTS USED IN THE'
      TYPE *,' FINITE WORD LENGTH INVESTIGATION'
      TYPE 800,INF
      TYPE 810,ICF
      TYPE 820,IAF
      TYPE 830,IQ1F
      TYPE 840,IQ2F
      TYPE 850,IQ3F
```

```
  800 FORMAT('          1.)SPEECH FACTOR:INF=',I2)
  810 FORMAT('          2.)CLIPPING FACTOR:ICF=',I2)
  820 FORMAT('          3.)PREDICTOR COEFFICIENTS:IAF=',I2)
  830 FORMAT('          4.)SCALING FACTORS & THRESHOLDS:IQ1F=',I2)
  840 FORMAT('          5.)EXPANSION VALUES:IQ2F=',I2)
  850 FORMAT('          6.)STANDARD DEVIATION:IQ3F=',I2)
      TYPE *,' TO CHANGE ANY OR ALL OF THESE TYPE---'
      TYPE *,' [SELECTION] (SELECTION) (SELECTION)...-UP TO 6 CHOICES'
      DO 855 I=1,6
              ANS4(I)=0
  855 CONTINUE
      ACCEPT 860,ANS4
  860 FORMAT(6(I1,1X))
      CHK=0
      DO 870  I=1,6
              CHK=CHK+ANS4(I)
  870 CONTINUE
      IF(CHK.EQ.0)GO TO 940
      ISELEC=1
  875 DO 880 I=1,6
              IF(ANS4(I).EQ.ISELEC)GO TO 890
  880 CONTINUE
      GO TO 930
  890 IF(ISELEC.EQ.1)TYPE *,' ENTER NEW VALUE OF INF'
      IF(ISELEC.EQ.1)ACCEPT *,INF
      IF(ISELEC.EQ.2)TYPE *,' ENTER NEW VALUE OF ICF'
      IF(ISELEC.EQ.2)ACCEPT *,ICF
      IF(ISELEC.EQ.3)TYPE *,' ENTER NEW VALUE OF IAF'
      IF(ISELEC.EQ.3)ACCEPT *,IAF
      IF(ISELEC.EQ.4)TYPE *,' ENTER NEW VALUE OF IQ1F'
      IF(ISELEC.EQ.4)ACCEPT *,IQ1F
      IF(ISELEC.EQ.5)TYPE *,' ENTER NEW VALUE OF IQ2F'
      IF(ISELEC.EQ.5)ACCEPT *,IQ2F
      IF(ISELEC.EQ.6)TYPE *,' ENTER NEW VALUE OF IQ3F'
      IF(ISELEC.EQ.6)ACCEPT *,IQ3F
  930 ISELEC=ISELEC+1
      IF(ISELEC.LE.6)GO TO 875
      GO TO 790
  940 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************CHOICE OF FILTERING
C***************
      TYPE *,' DO YOU WANT TO FILTER THE PITCH SPEECH?'
      TYPE *,' [YES=1,NO=0]'
      ACCEPT *,NOFLT
      IF(NOFLT.EQ.0)GO TO 700
      TYPE *,' WHICH FILTER DO YOU WANT?'
      TYPE *,'          1)8KHZ INPUT SPEECH WITH FILTER CUTOFF FREQ=500HZ'
      TYPE *,'          2)8KHZ INPUT SPEECH WITH FILTER CUTOFF FREQ=1KHZ'
      TYPE *,'          3)6.4KHZ INPUT SPEECH WITH FILTER CUTOFF FREQ=500HZ'
```

```
      TYPE *,'          4)6.4KHZ INPUT SPEECH WITH FILTER CUTOFF FREQ=1KHZ'
      ACCEPT *,I
      DO 66# J=1,5
              HF(J)=H(J,I)
  66# CONTINUE
      GO TO 71#
C*****************************************************************
C*****************************************************************
C****************
C****************CHOICE OF PROGRAM PATHS
C****************
  7## DO 71# J=1,5
              HF(J)=#
  71# CONTINUE
      TYPE *,' WHICH PROGRAM PATH DO YOU WANT TO TAKE?'
      TYPE *,'          1)TDHS-ARC WITH PITCH PASSED TO THE RCVR'
      TYPE *,'          2)TDHS-ARC WITH PITCH INDEPENDENTLY GENERATED'
      TYPE *,'                  IN THE RCVR'
      TYPE *,'          3)TDHS ONLY WITH PITCH PASSED TO THE TDHE ALGORITHM'
      TYPE *,'          4)TDHS ONLY WITH PITCH INDEPENDENTLY GENERATED'
      TYPE *,'                  IN THE TDHE ALGORITHM'
      ACCEPT *,IPATH
C*****************************************************************
C*****************************************************************
C****************
C****************CHOICE OF INITIAL PREDICTOR COEFFICIENTS AND GAIN
C****************
  845 TYPE *,' THE FOLLOWING ARE THE INITIAL PREDICTOR COEFFICIENTS'
      TYPE *,'                    AND GAIN FACTOR'
      DO 8#5 I=1,N
              TYPE 815,I,AB(I)
  8#5 CONTINUE
      TYPE 825,GAIN
  815 FORMAT('      AB(',I2,')=',2X,F8.4)
  825 FORMAT('      GAIN=',2X,F8.4)
      TYPE *,' DO YOU WNAT TO ENTER NEW VALUES?'
      TYPE *,'   YES=1          NO=#'
      ACCEPT *,NAI
      IF(NAI.EQ.#)GO TO 835
      TYPE *,' ENTER THE NEW INITIAL PREDICTOR COEFFICIENTS'
      ACCEPT *,(AB(I),I=1,N)
      TYPE *,' ENTER THE GAIN FACTOR'
      ACCEPT *,GAIN
      GO TO 845
C*****************************************************************
C*****************************************************************
C****************
C****************OUTPUT BLOCK
C****************
  835 WRITE(3,9)(FNAME(J),J=1,16)
      WRITE(3,*)INF,ICF,IAF,IQ1F,IQ2F,IQ3F
      WRITE(3,*)NBL
```

```
      WRITE(3,*)CLPP
      WRITE(3,*)KBLK1,ITMIN,ITMAX
      WRITE(3,*)ALP,ALAD,ABIAS,G,N,RMSMIN,SMIN
      WRITE(3,*)KQ
      WRITE(3,*)(T(I),I=1,NQQQ)
      WRITE(3,*)(OUT(I),I=1,NQQQ)
      WRITE(3,*)(EXPN(I),I=1,NQQQ)
      WRITE(3,*)ALPHA
      WRITE(3,*)ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
      WRITE(3,*)NOFLT
      WRITE(3,*)(HF(I),I=1,5)
      WRITE(3,*)IPATH
      WRITE(3,*)(AB(I),I=1,N)
      WRITE(3,*)GAIN
      CLOSE(UNIT=3)
      STOP
      END
C
C
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C       ********************************************
C       ********************************************
C       ***                                      ***
C       ***                                      ***
C       ***            T F W L . F T N           ***
C       ***                                      ***
C       ***                                      ***
C       ********************************************
C       ********************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE:  TFWL.FTN
C
C     BY:  JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTIVE TITLE: TRANSMITTER FINITE WORD LENGTH
C                              SIMULATION
C
C     DESCRIPTION:  THIS PROGRAM USES THE OUTPUT PARAMETER FILE
C                     OF PFWL.FTN, RENAMED CMDPARA.DAT, TO
C                     SET ALL THE OPTIONS OF THE TRANSMITTER.
C                     FOR THIS REASON, THE OPERATOR IS NOT
C                     PROMPTED FOR ANY INFORMATION DURING A RUN.
C                     THIS PROGRAM REPRESENTS THE FINITE WORD
C                     LENGTH SIMULATION OF THE TRANSMITTER FOR
C                     THE TDHS-ARC ALGORTIHM.  THE WORD LENGTH
C                     HAS BEEN RESTRICTED TO 16 BITS.  THE
C                     TRANSMISSION RATE IS ASSUMED TO BE 16KB/S.
C
C*****************************************************************
C*****************************************************************
C
C
      INTEGER HD(40),IPICH(400),FNAME(16),SQ,Q,WARE6
      INTEGER*2 CLPSP(512),STAT
      INTEGER*4 STRT,BUFSZ,ISF,ITF,IA4,IB4,IAF,ICF
      DIMENSION  H(400), SQ(16),ISBUF(512),IFSP(512),IYY(16)
      REAL ALPHA
C***************
C***************
C***************
      COMMON /PRED/G,ALP,ALAD,ABIAS,EP,EV,B(12)
     1         ,SPERB1(200),SPERB2(200),SNRB(200),SNRQB(200),N,KQ,NSPSAM
     2         ,ISTAT1(40),IVHAT(12),IV(12),IRMSMN,IA(12)
      COMMON /RMSB/ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG,IRMS
```

```
C
C
      INTEGER HD(40),IPICH(400),FNAME(16),SQ,O,WARE6
      INTEGER*2 CLPSP(512),STAT
      INTEGER*4 STRT,BUFSZ,ISF,ITF,IA4,IB4,IAF,ICF
      DIMENSION  H(400), SQ(16),ISBUF(512),IFSP(512),IYY(16)
      REAL ALPHA
C***************
C***************
C***************
      COMMON /PRED/G,ALP,ALAD,ABIAS,EF,EV,B(12)
     1         ,SPERB1(200),SPERB2(200),SNRB(200),SNRQB(200),N,KQ,NSPSAM
     2         ,ISTAT1(40),IVHAT(12),IV(12),IRMSMN,IA(12)
      COMMON /RMSB/ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG,IRMS
      COMMON /CLIPP/ALPHA,ICLPP,ISEC
      COMMON /ADDN/STAT(40)
      COMMON /WARE6/WARE6
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1         ,IPATH
      COMMON /FILT/HF(5),IX(9),IFSC,NOFLT
      COMMON /FWL/ISF,INF,ITF,ICF,IAF,IQ1F,IQ2F,IQ3F
      COMMON /IOCNT/IOCNT,IHSZ,ILSZ
C***************
C***************
C***************
      MODN(K) = K - (K-1)/16 * 16
C*****************************************************************
C*****************************************************************
C***************
C***************INITIALIZATION BLOCK.
C***************
      OPEN(UNIT=8,TYPE='OLD',NAME='CMDPARA.DAT')    !OPEN THE OPTION FILE.
C***************
C***************OPEN THE SPEECH FILE; READ THE HEADER AND SPEECH.
C***************
      READ(8,10)FNAME
10    FORMAT(16A2)
      OPEN(UNIT=1, TYPE='OLD', READONLY, NAME=FNAME, SHARED)! OPEN SPEECH FILE
      READ(1,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
20    FORMAT(6I5,10X,40A1)
30    FORMAT(16I5)
40    CONTINUE
C***************
C***************CREATE NEW FILE FOR QUANTIZER OUTPUT. WRITE
C***************THE HEADER.
C***************
      OPEN(UNIT=2, TYPE='NEW', NAME='QUANT.DAT',CARRIAGECONTROL='LIST')
   50 FORMAT(40A1)
      NSPSAM = NSAMP/2            !NUMBER OF COMPRESSED SPEECH SAMPLES
      WRITE(2,20)NSENT,IRATE,NSPSAM,IUPPR,ILOWR,NTERMS,HD
C***************
C***************DEFINE AND INITIALIZE VARIOUS PARAMETERS.
```

```
C***************
      K1=773            !SEEDS FOR RANDOM # GENERATOR
      K2=119
      DO 6# JJ=1,1##            !WARM UP GENERATOR
   6#          CALL RANDU(K1,K2,XX)
      NUMP = #
      NADD  = #
      ISEC=#
      IFSC=#
C***************
C***************INITIALIZE THE FWL SHIFTING PARAMETERS.
C***************
      READ(8,*)INF,ICF,IAF,IQ1F,IQ2F,IQ3F
      INF=2**INF
      IQ1F=2**IQ1F
      IQ2F=2**IQ2F
      IQ3F=2**IQ3F
      ISF=2**15
      ITF=ISF-1
      ICF=2**ICF
      IAF=2**IAF
      READ(8,*)NBL
      READ(8,*)CLPP
      IA4=CLPP*FLOAT(ICF)
      IB4=ABS(IA4)
      IF(IB4.GT.ITF)TYPE *,' OVERFLOW INITIAL ICLPP'
      ICLPP=FLOAT(IA4)
      READ(8,*)KBLK1,ITMIN,ITMAX
C***************
C***************INITIALIZATION SUBROUTINE.
C***************
      CALL INSTRT
C***************
C***************FILE MANAGEMENT FOR RUNS.
C***************
      IF(ITCS.NE.1)GO TO 1##
           OPEN(UNIT=1#,TYPE='NEW',NAME='TCLPSP.DAT',CARRIAGECONTROL='LIST')
              WRITE(1#,2#)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  1## IF(ITSE.NE.1)GO TO 2##
           OPEN(UNIT=7,TYPE='NEW',NAME='TSENGY.DAT',CARRIAGECONTROL='LIST')
              WRITE(7,2#)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  2## IF(ITPF.NE.1.AND.IPATH.NE.1.AND.IPATH.NE.3)GO TO 3##
           OPEN(UNIT=4,TYPE='NEW',NAME='TPITCH.DAT',CARRIAGECONTROL='LIST')
              WRITE(4,2#)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  3## IF(ITFF.NE.1)GO TO 4##
           OPEN(UNIT=11,TYPE='NEW',NAME='TFILTER.DAT',CARRIAGECONTROL='LIST')
              WRITE(11,2#)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  4## IF(IPATH.NE.3.AND.IPATH.NE.4)GO TO 6##
           OPEN(UNIT=12,TYPE='NEW',NAME='TZHAT.DAT',CARRIAGECONTROL='LIST')
              WRITE(12,2#)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  6## CONTINUE
C*****************************************************************
```

```
C*****************************************************************
C***************
C***************BEGIN MAIN LOOP. SPEECH BUFFER CREATED AND CLIPPED
C***************SPEECH GENERATED BY SUBROUTINE.
C***************
      ISENGY=Ø
      BUFSZ=2*ITMAX
      BUFSZ=(BUFSZ/16)*16+16
      READ(1,3Ø)(ISBUF(I),I=1,BUFSZ)
      DO 44 I=1,BUFSZ
              ISBUF(I)=ISBUF(I)*INF
   44 CONTINUE
      STRT=1
   22 CONTINUE
      IF(NOFLT.EQ.1)GO TO 49Ø
      DO 5Ø5 I=STRT,BUFSZ
              IFSP(I)=ISBUF(I)
  5Ø5 CONTINUE
      GO TO 485
C***************
C***************LOW PASS FILTER SUBROUTINE.
C***************
  49Ø CALL IFILT(ISBUF,STRT,BUFSZ,IFSP)
C***************
C***************SUBROUTINE TO CLIP SPEECH.
C***************
  485 CALL CLP(IFSP,STRT,BUFSZ,CLPSP,ISENGY)
C*****************************************************************
C*****************************************************************
C***************
C***************PITCH DETERMINED BY AUTOCORRELATION
C***************
   21 IBIGG=-1ØØØ
      I2SUM=-1ØØØ
      DO 12 IT=ITMIN,ITMAX+1
              ISUM=Ø
              DO 13 J=1,KBLK1
                      IF(((CLPSP(J+IT).LT.Ø).AND.(CLPSP(J).LT.Ø)).OR.
     1                        ((CLPSP(J+IT).GT.Ø).AND.(CLPSP(J).GT.Ø)))
     2                        ISUM=ISUM+1
                      IF(((CLPSP(J+IT).LT.Ø).AND.(CLPSP(J).GT.Ø)).OR.
     1                        ((CLPSP(J+IT).GT.Ø).AND.(CLPSP(J).LT.Ø)))
     2                        ISUM=ISUM-1
   13         CONTINUE
              IF(I2SUM.GT.IBIGG.AND.I2SUM.GT.ISUM)NP=IT-1
              IF(I2SUM.GT.IBIGG.AND.I2SUM.GT.ISUM)IBIGG=I2SUM
              I2SUM=ISUM
   12 CONTINUE
      NUMP=NUMP+1
C***************
C***************RANDOM PITCH GENERATOR.
C***************
```

```
      NINC=MODN(NUMP)
      IF(NP.GT.ITMIN) GO TO 29
              CALL RANDU(K1,K2,XX)
              NP=XX*(ITMAX-ITMIN)/2+ITMIN
              WRITE(5,462)NUMP,NP
   29 IPICH(NINC)=NP
  462 FORMAT(//6X,'NUMP=',I4,6X,'NP IS=',I4)
      IF(ITPF.NE.1.AND.IPATH.NE.1.AND.IPATH.NE.3)GO TO 500
      IF(NINC.EQ.16)WRITE(4,30)(IPICH(J),J=1,16)
C***************
C***************WINDOW FORMED BY SUBROUTINE FOR SPEECH COMPRESSION
C***************FROM PITCH GENERATED.
C***************
  500 CALL WINDOW(H,NP)
C*****************************************************************
C*****************************************************************
C***************
C***************SPEECH IS COMPRESSED AND ARC IS EMPLOYED.
C***************
      DO 14 I=1,NP
              NUM4=NP+I
              IARG=NADD+I
              N1=MODN(IARG)
              IA4=ISBUF(NUM4)
              IB4=ISBUF(I)
              IA4=IB4-IA4
              IB4=ABS(IA4)
              IF(IB4.GT.ITF)TYPE *,' OVERFLOW COMP ADD'
              IA4=FLOAT(ISBUF(NUM4))+H(I)*FLOAT(IA4)
              IB4=ABS(IA4)
              IF(IB4.GT.ITF)TYPE *,' OVERFLOW COMP ADD 2'
              IYYY=IA4
              IYY(N1)=IYYY/INF
              IF((IPATH.EQ.3.OR.IPATH.EQ.4).AND.N1.EQ.16)WRITE(12,30)
     1                        (IYY(J),J=1,16)
              CALL ARC(IYYY,Q,IYHAT)
              JQQ=Q
              STAT(JQQ)=STAT(JQQ)+1
              SQ(N1)=Q
              IF(N1.EQ.16)WRITE(2,33)SQ
   14 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************UPDATE FOR NEXT BLOCK OF SPEECH.  BUFFERS ARE
C***************SHIFTED OVER 2*NP PLACES AND THEN FILLED AGAIN.
C***************IF THE END OF THE FILE IS REACHED, ZEROS ARE AP-
C***************PENDED TO FINISH PROCESSING THE FILE.
C***************
      NADD=NADD+NP
      NP2=2*NP
      INBUFF=BUFSZ-NP2
```

```
      DO 15 I=1,INBUFF
              ISBUF(I)=ISBUF(NP2+I)
              CLPSP(I)=CLPSP(NP2+I)
              IFSP(I)=IFSP(NP2+I)
   15 CONTINUE
      BUFSZ=2*ITMAX
      MORE=BUFSZ-INBUFF
      IF(MOD(MORE,16).EQ.Ø)GO TO 25
      MORE=(MORE/16)*16+16
   25 BUFSZ=INBUFF+MORE
      NLEFT=NSAMP-2*NADD
      IF(MORE.GT.NLEFT)GO TO 23
      READ(1,3Ø,END=26)(ISBUF(INBUFF+I),I=1,MORE)
      DO 45 I=1,MORE
              ISBUF(INBUFF+I)=ISBUF(INBUFF+I)*INF
   45 CONTINUE
   26 STRT=INBUFF+1
      GO TO 22
   23 CHK=Ø.
      DO 17 I=1,INBUFF
              CHK=CHK+FLOAT(ABS(ISBUF(I)))
   17 CONTINUE
      IF(CHK.EQ.Ø.)GO TO 24
      NZEROS=MORE-NLEFT
      READ(1,3Ø,END=27)(ISBUF(INBUFF+I),I=1,NLEFT)
      DO 46 I=1,NLEFT
              ISBUF(INBUFF+I)=ISBUF(INBUFF+I)*INF
   46 CONTINUE
   27 DO 16 I=1,NZEROS
              ISBUF(INBUFF+NLEFT+I)=Ø
   16 CONTINUE
      STRT=INBUFF+1
      GO TO 22
   24 NONES=16-N1
      DO 18 I=1,NONES
              SQ(N1+I)=1
   18 CONTINUE
      WRITE(2,33)SQ
      IF(ITPF.EQ.1.OR.IPATH.EQ.1.OR.IPATH.EQ.3)WRITE(4,3Ø)
     1                (IPICH(J),J=1,NINC)
      IF(IPATH.GE.3)WRITE(12,3Ø)(IYY(J),J=1,N1)
C***************
C***************STATISTICS GENERATED FOR OUTPUT FILE BY SUBROUTINE.
C***************
      CALL INEND
C*****************************************************************
C*****************************************************************
C***************
C***************CLOSE FILES.
C***************
   33 FORMAT(16I2)
      WRITE(6,*)IHSZ,ILSZ
```

```
CLOSE(UNIT=7)
CLOSE(UNIT=1)
CLOSE(UNIT=2)
CLOSE(UNIT=8)
CLOSE(UNIT=4)
CLOSE(UNIT=11)
STOP
END
```

```
C
C
C
C
C     ************************************
C     *                                  *
C     *   W I N D O W   F U N C T I O N  *
C     *                                  *
C     ************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE PRODUCES THE VALUES "H", WHICH
C                  CORRESPOND TO A TRIANGULAR WINDOW.  THIS
C                  WINDOW IS GENERATED EVERYTIME A PITCH (NP)
C                  IS PASSED TO IT.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE WINDOW(H,NP)
C***************
C***************
C***************
      DIMENSION H(4ØØ)
      DO 1Ø   I = 1, NP
              H(I) = 1.Ø - FLOAT(I-1)/FLOAT(NP-1)
1Ø    CONTINUE
      RETURN
      END
```

```
C
C
C
C
C     ******************************
C     *                            *
C     *   Q U A N T I Z E R        *
C     *                            *
C     ******************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE MODELS A FIXED QUANTIZER. THE INPUT
C              IS THE ERROR DETERMINED BY THE PREDICTOR IN THE
C              ADAPTIVE RESIDUAL CODER AND THE OUTPUTS ARE THE
C              QUANTIZED ERROR AND THE ASSOCIATED THRESHOLD LEVEL
C              NUMBER INTO WHICH THIS ERROR MAPPED.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE QUANT(IX,IY,I)
C***************
C***************
C***************
      INTEGER*2 STAT
      INTEGER*4 ISF,ITF,ICF,IAF,IA4,IB4
C***************
C***************
      COMMON /QUAN/SMIN,NQ,IT(51),IOUT(51),ISIZE,IEXPN(51)
      COMMON /RMSB/ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG,IRMS
      COMMON /ADDN/STAT(40)
      COMMON /FWL/ISF,INF,ITF,ICF,IAF,IQ1F,IQ2F,IQ3F
      COMMON /IOCNT/IOCNT,IHSZ,ILSZ
C***************
C***************
      IF(ISIZE.GT.IHSZ)IHSZ=ISIZE
      IF(ISIZE.LT.ILSZ)ILSZ=ISIZE
      SCALE=FLOAT(INF)/(FLOAT(IQ3F)*FLOAT(IQ1F))
      F=0.5
      IF(IX.LT.0.)F=-.5
C*****************************************************************
C*****************************************************************
C***************
C***************DETERMINATION OF APPROPRIATE LEVEL.
C***************
      I=1
      IXX=IABS(IX)
```

```
      DO 20 K=1,NQ
      TNEW=FLOAT(IT(K))*FLOAT(ISIZE)*SCALE*2.
   20 IF(IXX.GE.TNEW) I=2*K+.5+F
      J=(I+2)/2
C*****************************************************************
C*****************************************************************
C***************
C***************THE VALUE OF THE QUANTIZED ERROR IS DETERMINED.
C***************
      IY=2.*F*FLOAT(IOUT(J))*FLOAT(ISIZE)/FLOAT(ISF)
      SCALE=FLOAT(ISF)*SCALE
      IY=IY*SCALE*2.
C*****************************************************************
C*****************************************************************
C***************
C***************THE STANDARD DEVIATION ESTIMATE IS UPDATED.
C***************
      ISIZE=FLOAT(IEXPN(J))*FLOAT(ISIZE)/FLOAT(ISF)
      SCALE=FLOAT(ISF)/FLOAT(IQ2F)
      ISIZE=FLOAT(ISIZE)*SCALE
      IRRMS=IRMS/2
      IF(IQ3F.GT.INF)ISIZE=FLOAT(ISIZE)*(FLOAT(INF)/FLOAT(IQ3F))
      IF(INF.GT.IQ3F)IRRMS=FLOAT(IRRMS)*(FLOAT(IQ3F)/FLOAT(INF))
      ISIZE=IMAX0(ISIZE,IRRMS)
      IF(IQ3F.GT.INF)ISIZE=FLOAT(ISIZE)*(FLOAT(IQ3F)/FLOAT(INF))
      IF(INF.GT.IQ3F)ISIZE=FLOAT(ISIZE)*(FLOAT(INF)/FLOAT(IQ3F))
      RETURN
      END
```

```
C
C
C
C
C     ********************************
C     *                              *
C     *  I N I T I A L I Z A T I O N  *
C     *                              *
C     ********************************
C
C
C*******************************************************************
C*******************************************************************
C
C     DESCRIPTION: SUBROUTINE INITIALIZES MOST OF THE PARAMETERS
C             BY READING FROM CMDPARA.DAT, THE RENAMED OUTPUT OF
C             PFWL.FTN.  OTHER PARAMETERS ARE ALSO INITIALIZED
C             TO THEIR APPROPRIATE STARTING VALUES.
C
C*******************************************************************
C*******************************************************************
C***************
C***************
C***************
      SUBROUTINE INSTRT
C***************
C***************
C***************
      INTEGER WARE6
      INTEGER*4 ISF,ITF,ICF,IAF
      DIMENSION T(51),OUT(51),EXPN(51)
C***************
C***************
      COMMON /QUAN/SMIN,NQ,IT(51),IOUT(51),ISIZE,IEXPN(51)
      COMMON /PRED/G,ALP,ALAD,ABIAS,EP,EV,B(12)
     1        ,SPERB1(200),SPERB2(200),SNRB(200),SNRQB(200),N,KQ,NSPSAM
     2        ,ISTAT1(40),IVHAT(12),IV(12),IRMSMN,IA(12)
      COMMON /RMSB/ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG,IRMS
      COMMON /INIT/JI
      COMMON /WARE6/WARE6
      COMMON /CLIPP/ALPHA,ICLPP,ISEC
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /FILT/HF(5),IX(9),IFSC,NOFLT
      COMMON /FWL/ISF,INF,ITF,ICF,IAF,IQ1F,IQ2F,IQ3F
      COMMON /IOCNT/IOCNT,IHSZ,ILSZ
      COMMON /GAIN/GAIN,EQ
      COMMON /STATS/ENGY5,ENGY6,ENGY7,ENGY8,NDIV
C***************
C***************
      IOCNT=0
      IHSZ=0
```

```
      ILSZ=10000
      WARE6=9
      READ(8,*)ALP,ALAD,ABIAS,G,N,RMSMIN,SMIN !ARC PARAMETERS
      IRMSMN=RMSMIN*FLOAT(INF)
    2 FORMAT(/6X,'ALP=',F5.2,2X,'ALAD=',F5.2,2X,'G='
     1        F5.3,2X,'N=',I2,2X,'RMSMIN=',F5.1,2X,'SMIN=',F5.2,2X,'ABIAS='
     2        F5.3/)
      READ(8,*)KQ
      NQ=KQ/2
      NQQQ=NQ+1
C*****************************************************************
C*****************************************************************
C***************
C***************QUANTIZER VALUES ARE SET AND APPROPRIATELY
C***************SHIFTED.
C***************
      READ(8,*)(T(I),I=1,NQQQ)  !THRESHOLD VALUES
      READ(8,*)(OUT(I),I=1,NQQQ)   !SCALING FACTORS
      READ(8,*)(EXPN(I),I=1,NQQQ)   !EXPANSION FACTORS
      DO 400 I=1,NQQQ
              IOUT(I)=OUT(I)*FLOAT(IQ1F)*SMIN
              IEXPN(I)=EXPN(I)*FLOAT(IQ2F)
              IT(I)=T(I)*FLOAT(IQ1F)*SMIN
  400 CONTINUE
      READ(8,*)ALPHA   !SIGNAL ENERGY PARAMETER
C***************FILE MANAGEMENT FLAGS.
      READ(8,*)ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
      READ(8,*)NOFLT   !FILTER FLAG
      READ(8,*)(HF(I),I=1,5)    !FILTER COEFFICIENTS
      READ(8,*)IPATH !DETERMINES MAJOR ROUTE OF PROGRAM
      READ(8,*)(B(I),I=1,N) !INITIAL PREDICTOR COEFFICIENTS
      READ(8,*)GAIN
C*****************************************************************
C*****************************************************************
C***************
C***************OPEN STATISTICS FILE AND WRITE PARAMETERS.
C***************
      IF(ITSF.NE.1)GO TO 201
      OPEN(UNIT=9,TYPE='NEW',NAME='TSTATS.DAT')
      WRITE(WARE6,2)ALP,ALAD,G,N,RMSMIN,SMIN,ABIAS
      WRITE(WARE6,160)KQ,NQQQ
  160 FORMAT(//' THE FOLLOWING IS A LIST OF SCALING FACTORS,'/
     1        ' EXPANSION FACTORS, AND THRESHOLDS FOR A ',I2,' LEVEL'/
     2        ' QUANTIZER(',I2,' ON A SIDE)'//)
      WRITE(WARE6,180)
      DO 190 I=1,NQQQ
              WRITE(WARE6,200)I,OUT(I),T(I),EXPN(I)
  190 CONTINUE
  180 FORMAT(/3X,'LEVEL NUMBER',4X,'SCALING FACTORS',4X,'THRESHOLDS',
     1        4X,'EXPANSION FACTORS'/)
  200 FORMAT(6X,I2,17X,F7.4,9X,F7.4,10X,F7.4)
  201 ISIZE=100*IQ3F  !INITIAL STANDARD DEVIATION ESTIMATE
```

```
      DO 118 I=1,12
      IVHAT(I)=0
      IV(I)=0
      B(I)=B(I)*FLOAT(IAF)
      IA(I)=B(I)
  118 CONTINUE
      ABIAS=ABIAS*FLOAT(IAF)
      DO 119 I=1,9
              IX(I)=0
  119 CONTINUE
      IRMS=IRMSMN
C*********************************************************************
C*********************************************************************
C***************
C***************ENERGY VALUES FOR STATISTICS ARE INITIALLY SET TO
C***************ZERO.
C***************
      EV=0.
      EP=0.
      ENGY1=0.
      ENGY2=0.
      JI=0
      ENGY3=0.
      ENGY4=0.
      ENGY5=0.
      ENGY6=0.
      ENGY7=0.
      ENGY8=0.
      NDIV=0
      SPER1=0.
      SPER2=0.
      DO 621 I=1,KQ
  621 ISTAT1(I)=0
      RETURN
      END
```

```
C
C
C
C
C     ***************************************************
C     *                                                 *
C     *  A D A P T I V E   R E S I D U A L   C O D E R  *
C     *                                                 *
C     ***************************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION:  SUBROUTINE MODELS THE ADAPTIVE RESIDUAL
C             CODER.  THE INPUT IS A COMPRESSED SPEECH SAMPLE
C             AND  A QUANTIZATION LEVEL IS RETURNED TO BE
C             PASSED ON TO THE CHANNEL.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE ARC(IY,Q,IYHAT)
C***************
C***************
C***************
      INTEGER Q,EQ
      INTEGER*4 ISF,ITF,IA4,IB4,ICF,IAF,IC4
C***************
C***************
      COMMON /PRED/G,ALP,ALAD,ABIAS,EP,EV,B(12)
     1        ,SPERB1(200),SPERB2(200),SNRB(200),SNRQB(200),N,KQ,NSPSAM
     2        ,ISTAT1(40),IVHAT(12),IV(12),IRMSMN,IA(12)
      COMMON /RMSB/ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG,IRMS
      COMMON /INIT/JI
      COMMON /FWL/ISF,INF,ITF,ICF,IAF,IQ1F,IQ2F,IQ3F
      COMMON /IOCNT/IOCNT,IHSZ,ILSZ
      COMMON /GAIN/GAIN,EQ
      COMMON /STATS/ENGY5,ENGY6,ENGY7,ENGY8,NDIV
C*****************************************************************
C*****************************************************************
C***************
C***************PREDICTION IS COMPUTED AND AN ERROR SMAPLE IS
C***************GENERATED TO BE PASSED ON TO THE QUANTIZER.
C***************
C
      IOCNT=IOCNT+1
      IPRE=0
      IPRE1=0
      DO 120 I=1,N
```

```
                IPRE1=FLOAT(IPRE1)+IA(I)*FLOAT(IV(I))/FLOAT(IAF)
                IPRE=FLOAT(IPRE)+IA(I)*FLOAT(IVHAT(I))/FLOAT(IAF)
  120 CONTINUE
      IRMS=ALP*FLOAT(IRMS-IRMSMN)+FLOAT(IRMSMN)
      IRMS=FLOAT(IRMS)+(1.-ALP)*ABS(FLOAT(IVHAT(1)))
C***************
C***************ERROR SAMPLE GENERATED.
C***************
      IERROR=IY-IPRE
      IERRR1=IY-IPRE1
C***************
C***************QUANTIZER SUBROUTINE IS ENGAGED.
C***************
      CALL QUANT(IERROR,EQ,IOUT)
C***************
C***************UPDATE PREDICTOR VALUES.
C***************
      ISTAT1(IOUT)=ISTAT1(IOUT)+1
      Q=IOUT
      DO 125 I=1,N
      J=N+2-I
      IV(J)=IV(J-1)
  125 IVHAT(J)=IVHAT(J-1)
      IVHAT(1)=IPRE+EQ
      IV(1)=IY
      IYHAT=IVHAT(1)
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,*)IOCNT
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,*)IY,IPRE,EQ,IYHAT
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,244)
  244 FORMAT('    ')
C*****************************************************************
C*****************************************************************
C***************
C***************ADAPTATION OF PREDICTOR COEFFICIENTS IS ACCOMPLISHED
C***************BY A FIXED PERCENTAGE COMPUTATION.
C***************
      IF(G.EQ.0.0)GO TO 134
      DO 100 I=1,N
                IA(I)=IA(I)-B(I)
  100 CONTINUE
      DO 130 I=1,N
                SVL=ISIGN(1,EQ)*ISIGN(1,IVHAT(I+1))
                IF((EQ.EQ.0.OR.IVHAT(I+1).EQ.0).AND.IA(I).GE.0)SVL=-1.0
                IF((EQ.EQ.0.OR.IVHAT(I+1).EQ.0).AND.IA(I).LT.0)SVL=+1.0
                AINT=SVL*AMAX1(FLOAT(IABS(IA(I)))*G,ABIAS)
                IA(I)=ALAD*(IA(I)+AINT)
  130 CONTINUE
      DO 150 I=1,N
                IA(I)=IA(I)+B(I)
  150 CONTINUE
  134 IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,*)IRMS
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,*)(IA(KKK),KKK=1,4)
```

```
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,244)
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,244)
C*****************************************************************
C*****************************************************************
C***************
C***************ENERGY STATISTICS ARE GENERATED FOR DETERMINATION
C***************OF SNR VALUES REPORTED IN STATISTICS FILE.
C***************
      IRM=MOD(IARG,NBL)
      ENGY1=ENGY1+FLOAT(IY)**2.
      ENGY2=ENGY2+(FLOAT(IERROR)-FLOAT(EQ))**2.
      ENGY3=ENGY3+FLOAT(IERROR)**2
      ENGY4=ENGY4+FLOAT(IERRR1)**2
      ENGY5=ENGY5+FLOAT(IY)*FLOAT(IYHAT)
      ENGY6=ENGY6+FLOAT(IYHAT)**2
      ENGY7=ENGY7+FLOAT(IYHAT)
      ENGY8=ENGY8+FLOAT(IY)
      NDIV=NDIV+1
      IF(IRM .NE. 0)GOTO 133
      JI=JI+1
      EV=EV+ENGY1
      EP=EP+ENGY2
      IF(ENGY3.NE.0..AND.ENGY1.NE.0.)SPERB1(JI)=10.*ALOG10(ENGY1/ENGY3)
      IF(ENGY4.NE.0..AND.ENGY1.NE.0.)SPERB2(JI)=10.*ALOG10(ENGY1/ENGY4)
      IF(ENGY2.NE.0..AND.ENGY1.NE.0.)SNRB(JI)=10.*ALOG10(ENGY1/ENGY2)
      IF(ENGY2.NE.0..AND.ENGY3.NE.0.)SNRQB(JI)=10.*ALOG10(ENGY3/ENGY2)
      SPER1=SPER1+ENGY3
      SPER2=SPER2+ENGY4
      ENGY1=0.
      ENGY2=0.
      ENGY3=0.
      ENGY4=0.
133   CONTINUE
      RETURN
      END
```

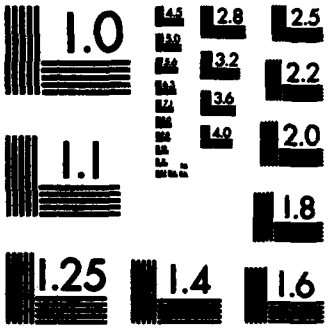
```
C
C
C
C
C     ****************************************
C     *                                      *
C     *    STATISTICS   AND   RESULTS        *
C     *                                      *
C     ****************************************
C
C
C*******************************************************************
C*******************************************************************
C
C     DESCRIPTION: SUBROUTINE GGENERATES THE SNR VALUES FOR THE
C             STATISTICS FILE.
C             SNR INLOOP= SUM OF SQUARED SIGNAL ENERGY/SUM OF
C                     SQUARED VALUE OF (TRUE ERROR-QUANTIZED
C                     ERROR.
C             PREDICTOR PERFORMANCE= SUM OF SQUARED SIGNAL ENERGY/
C                     SUM OF SQUARED VALUE OF TRUE ERROR.
C             PREDICTOR IDEAL PERFORMANCE= SUM OF SQUARED SIGNAL
C                     ENERGY/SUM OF SQUARED VALUE OF TRUE ERROR
C                     GENERATED USING TRUE PREDICTOR VALUES.
C             SIGNAL TO NOISE RATIO= SUM OF SQUARED VALUE OF TRUE
C                     ERROR/SUM OF SQUARED VALUE OF (TRUE ERROR-
C                     QUANTIZED ERROR).
C
C*******************************************************************
C*******************************************************************
C***************
C***************
C***************
      SUBROUTINE INEND
C***************
C***************
C***************
      REAL PROB(40),ALPHA
      INTEGER*2 STAT
      INTEGER WARE6
C***************
C***************
      COMMON /PRED/G,ALP,ALAD,ABIAS,EP,EV,B(12)
     1        ,SPERB1(200),SPERB2(200),SNRB(200),SNRQB(200),N,KQ,NSPSAM
     2        ,ISTAT1(40),IVHAT(12),IV(12),IRMSMN,IA(12)
      COMMON /RMSB/ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG,IRMS
      COMMON /ADDN/STAT(40)
      COMMON /WARE6/WARE6
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /STATS/ENGY5,ENGY6,ENGY7,ENGY8,NDIV
C***************
```

```
C***************
      IF(ITSF.NE.1)RETURN
      SNR=10.*ALOG10(EV/EP)
      SPER=10.*ALOG10(EV/SPER1)
      SPERI=10.*ALOG10(EV/SPER2)
      SNRQ=10.*ALOG10(SPER1/EP)
      ISUM=0
      DO 300 II=1,KQ
300   ISUM=ISUM+ISTAT1(II)
      ARG1=ISUM
      SUM=ALOG(ARG1)
      DO 500 I=1,KQ
      ARG=ISTAT1(I)+0.001
  500 SUM=SUM-(ARG*ALOG(ARG))/ARG1
      BITS=SUM/ALOG(2.)
      WRITE(WARE6,402)SNR,BITS,(ISTAT1(I),I=1,KQ)
  402 FORMAT(6X,'SNR INLOOP=',F7.3,3X,'H=',F4.2,3X,'OP',
     1        10I5,3(/22X,10I5)//)
      IF(WARE6.EQ.6)WRITE(5,402)SNR,BITS,(ISTAT1(I),I=1,KQ)
      WRITE(WARE6,404)
404   FORMAT(///6X,' SAMPLE NUMBER ',3X,' SNR ',7X,' SPER ',5X,
     1        ' SPERI ',4X,' SNRQ '//)
      NB=ISUM/NBL
      DO 408  I=1,NB
              IS=(I-1)*NBL+1
              IE=IS+NBL-1
              WRITE(WARE6,412)IS,IE,SNRB(I),SPERB1(I),SPERB2(I),SNRQB(I)
408   CONTINUE
412   FORMAT(6X,I5,'-',I5,6X,F7.2,4X,F7.2,4X,F7.2,4X,F7.2)
      WRITE(WARE6,416)SPER,SPERI,SNRQ
416   FORMAT(//6X,' PREDICTOR PERFORMANCE =',F8.2/6X,
     1        ' PREDICTOR IDEAL PERFORMANCE= ',F8.2/6X,' SIGNAL TO NOISE
     2        RATIO=',F8.2)
      NQUA=0
      DO 418  I=1,32
418   NQUA=NQUA+STAT(I)
      DO 420  I=1,32
              PROB(I)=FLOAT(STAT(I))/FLOAT(NQUA)
420   CONTINUE
      WRITE(WARE6,422)
422   FORMAT(//,6X,'LEVEL NUMBER',6X,'PROBABILITY',6X,' FREQUENCY',/)
424   FORMAT(9X,I2,12X,F7.4,10X,I5)
      DO 426  I=1,32
      WRITE(WARE6,424)I,PROB(I),STAT(I)
426   CONTINUE
      RETURN
      END
```

```
C
C
C
C
C     ************************************
C     *                                  *
C     *      C L I P P I N G             *
C     *                                  *
C     ************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE ACCEPTS BLOCK OF SPEECH SAMPLES
C              AS INPUT AND THREE VALUE CENTER CLIPS THIS
C              SPEECH BY A FIXED PERCENTAGE OF THE COMPUTED
C              SIGNAL ENERGY.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE CLP(IFSP,STRT,FNSH,CLPSP,SENGY)
C***************
C***************
C***************
      REAL ALPHA
      INTEGER*4 STRT,FNSH,ISF,ITF,IA4,IB4,ICF,IAF
      INTEGER*2 CLPSP(512),CLPLVL,CLM
      INTEGER SENG(16),CLIPFI(16)
      DIMENSION IFSP(512)
C***************
C***************
      COMMON /CLIPP/ALPHA,ICLPP,ISEC
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /FWL/ISF,INF,ITF,ICF,IAF,IQ1F,IQ2F,IQ3F
C***************
C***************
      DO 21 I=STRT,FNSH
              MSEC=MOD(ISEC,16)+1
              ISEC=ISEC+1
              IFLTR=IFSP(I)
              IA4=FLOAT(ISENGY)*ALPHA+(1.-ALPHA)*(ABS(FLOAT(IFLTR)))
              IB4=ABS(IA4)
              IF(IB4.GT.ITF)TYPE *,' OVERFLOW ISENGY'
              ISENGY=FLOAT(IA4)
              SENG(MSEC)=ISENGY/INF
              IF(ITSE.NE.1)GO TO 20
              IF(MSEC.EQ.16)WRITE(7,10)(SENG(JJ),JJ=1,16)
```

AD-A135 232 REAL-TIME IMPLEMENTATION OF A SPEECH DIGITIZATION ALGORITHM COMBINING TIM..(U) NOTRE DAME UNIV IN DEPT OF ELECTRICAL ENGINEERING J L MELSA ET AL. JUN 83 2/2

UNCLASSIFIED DCA100-82-C-0026 F/G 9/4 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

```
20          IA4=FLOAT(ICLPP)*FLOAT(ISENGY)/ICF
            IB4=ABS(IA4)
            IF(IB4.GT.ITF)TYPE *,' OVERFLOW CLPP'
            CLPLVL=FLOAT(IA4)
            CLM=-CLPLVL
            CLPSP(I)=0
            IF (IFLTR.GT.CLPLVL)CLPSP(I)=+1
            IF (IFLTR.LT.CLM)CLPSP(I)=-1
            CLIPFI(MSEC)=CLPSP(I)
            IF(ITCS.NE.1)GO TO 21
            IF(MSEC.EQ.16)WRITE(10,10)(CLIPFI(JJ),JJ=1,16)
 21 CONTINUE
 10 FORMAT(16I5)
    RETURN
    END
```

```
C
C
C
C
C     ************************************************
C     *                                              *
C     *   C L I P P I N G    F I L T E R             *
C     *                                              *
C     ************************************************
C
C
C*******************************************************************
C*******************************************************************
C
C     DESCRIPTION: SUBROUTINE ACCEPTS BLOCK OF SPEECH SAMPLES
C             AND FILTERS THEM WITH A 9TH ORDER LINEAR PHASE
C             FINITE IMPULSE RESPONSE FILTER.
C
C*******************************************************************
C*******************************************************************
C***************
C***************
C***************
      SUBROUTINE IFILT(ISBUF,STRT,FNSH,IFSP)
C***************
C***************
C***************
      DIMENSION ISBUF(512),IFSP(512),IFO(16)
      INTEGER*4 STRT,FNSH,IA4,ISF,ITF,IB4,ICF,IAF
C***************
C***************
      COMMON /FILT/HF(5),IX(9),IFSC,NOFLT
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /FWL/ISF,INF,ITF,ICF,IAF,IQ1F,IQ2F,IQ3F
C***************
C***************
   10 FORMAT(16I5)
      DO 500 I=STRT,FNSH
              IFINC=MOD(IFSC,16)+1
              IFSC=IFSC+1
              DO 100 J=2,9
                      JIJ=11-J
                      IX(JIJ)=IX(JIJ-1)
  100         CONTINUE
              IX(1)=ISBUF(I)
              IFSP(I)=0
              DO 200 KIK=1,4
                      IA4=FLOAT(IFSP(I))+HF(KIK)*(FLOAT(IX(KIK))+
     1                                FLOAT(IX(10-KIK)))
                      IB4=ABS(IA4)
                      IF(IB4.GT.ITF)TYPE *,' OVERFLOW IFSP LOOP'
```

```
                    IFSP(I)=FLOAT(IA4)
  200         CONTINUE
                    IA4=FLOAT(IFSP(I))+HF(5)*FLOAT(IX(5))
                    IB4=ABS(IA4)
                    IF(IB4.GT.ITF)TYPE *,' OVERFLOW IFSP'
                    IFSP(I)=FLOAT(IA4)
              IFO(IFINC)=IFSP(I)/INF
              IF(ITFF.NE.1)GOTO 500
              IF(IFINC.EQ.16)WRITE(11,10)(IFO(JJJ),JJJ=1,16)
  500 CONTINUE
      RETURN
      END
C
C
C
C
C*********************************************************************
C*********************************************************************
```

```
C
C
C
C
C       ********************************************
C       ********************************************
C       ***                                      ***
C       ***                                      ***
C       ***              C B K . F T N           ***
C       ***                                      ***
C       ***                                      ***
C       ********************************************
C       ********************************************
C
C
C*******************************************************************
C*******************************************************************
C
C     TITLE: CBK.FTN
C
C     DESCRIPTION: THIS PROGRAM INDUCES ERRORS INTO THE
C             SEQUENCE OF QUANTIZATION LEVELS.  IT IS RES-
C             TRICTED TO INDUCING ONLY ONE ERROR PER CODEWORD
C             (FOUR BIT STRING) AND OPERATES AT AN INTERACTIVE
C             BER.  IT IS TO BE USED AT A 16KHZ TRANSMISSION
C             RATE AND AN 8KHZ SAMPLED SPEECH RATE TDHS-ARC
C             SYSTEM SINCE IT WILL INDUCE ERRORS ONLY IN THE
C             FIRST 15 LEVELS.  THE CODEWORD ASSIGNMENTS WERE
C             CHOSEN BY ARVIND ARORA AND ARE BASED ON A PSEUDO-MSE
C             FOR A ONE BIT ERROR PER CODEWORD SCHEME.
C
C
C*******************************************************************
C*******************************************************************
C
C
      DIMENSION IN(16),IOUT(16)
C***************
C***************FORMATS USED.
C***************
   1# FORMAT(16I2)
   2# FORMAT(' # ERRORS= ',F5.#,3X,F5.#,4X,F5.#,3X,F5.#)
   3# FORMAT(6I5,1#X,4#A1)
   4# FORMAT(' % ERRORS= ',F5.4,3X,F5.4,4X,F5.4,3X,F5.4)
   5# FORMAT(' TOTAL ERROR  = ',F5.4,' COMPARED TO DESIGN =',F5.4)
C***************
C***************FILE MANAGEMENT.
C***************
      OPEN(UNIT=2,TYPE='NEW',NAME='CSTATS.DAT')
      OPEN(UNIT=3,TYPE='OLD',NAME='QUANT.DAT')
      OPEN(UNIT=4,TYPE='NEW',NAME='DECO.DAT',CARRIAGECONTROL='LIST')
      READ(3,3#)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
```

```
      WRITE(4,30)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
C***************
C***************INITIALIZATION.
C***************
      E1=0.
      E2=0.
      E3=0.
      E4=0.
      CNT=0.
      K1=101
      K2=222
      TYPE *,' ENTER BER IN X'
      ACCEPT *,BER
      PERR=1.-(1.-(BER/100.))**4.
      DO 100 I=1,100
              CALL RANDU(K1,K2,XX)
  100 CONTINUE
      READ(3,10,END=990)(IN(J),J=1,16)
C***************
C***************DETERMINE IF ERROR EXPECTED IN THIS CODEWORD.
C***************
  200 DO 900 I=1,16
      CNT=CNT+1.
              IQ=IN(I)
              CALL RANDU(K1,K2,XX)
              IF(XX.GT.PERR)IQQ=IQ
              IF(XX.GT.PERR)GO TO 800
      CALL RANDU(K1,K2,XX)
      IF(XX.GT..25)GO TO 300
C*****************************************************************
C*****************************************************************
C***************
C***************BIT ERROR IN LEFTMOST BIT.
C***************
      E1=E1+1.
      IF(IQ.NE.1)GO TO 250
              CALL RANDU(K1,K2,XX)
              IF(XX.GT..5)IQQ=11
              IF(XX.LE..5)IQQ=10
  250 IF(IQ.EQ.2)IQQ=13
      IF(IQ.EQ.3)IQQ=12
      IF(IQ.EQ.4)IQQ=7
      IF(IQ.EQ.5)IQQ=6
      IF(IQ.EQ.6)IQQ=5
      IF(IQ.EQ.7)IQQ=4
      IF(IQ.EQ.8)IQQ=15
      IF(IQ.EQ.9)IQQ=14
      IF(IQ.EQ.10)IQQ=1
      IF(IQ.EQ.11)IQQ=1
      IF(IQ.EQ.12)IQQ=3
      IF(IQ.EQ.13)IQQ=2
      IF(IQ.EQ.14)IQQ=9
```

```
      IF(IQ.EQ.15)IQQ=8
      WRITE(2,*)CNT,IQ,IQQ,XX
      GO TO 800
  300 IF(XX.GT..50)GO TO 400
C*****************************************************************
C*****************************************************************
C***************
C***************BIT ERROR IN SECOND MSB.
C***************
      E2=E2+1.
      IF(IQ.NE.1)GO TO 350
              CALL RANDU(K1,K2,XX)
              IF(XX.GT..5)IQQ=5
              IF(XX.LE..5)IQQ=4
  350 IF(IQ.EQ.2)IQQ=12
      IF(IQ.EQ.3)IQQ=13
      IF(IQ.EQ.4)IQQ=1
      IF(IQ.EQ.5)IQQ=1
      IF(IQ.EQ.6)IQQ=10
      IF(IQ.EQ.7)IQQ=11
      IF(IQ.EQ.8)IQQ=14
      IF(IQ.EQ.9)IQQ=15
      IF(IQ.EQ.10)IQQ=6
      IF(IQ.EQ.11)IQQ=7
      IF(IQ.EQ.12)IQQ=2
      IF(IQ.EQ.13)IQQ=3
      IF(IQ.EQ.14)IQQ=8
      IF(IQ.EQ.15)IQQ=9
      WRITE(2,*)CNT,IQ,IQQ,XX
      GO TO 800
  400 IF(XX.GT..75)GO TO 500
C*****************************************************************
C*****************************************************************
C***************
C***************BIT ERROR IN THIRD MSB.
C***************
      E3=E3+1.
      IF(IQ.NE.1)GO TO 450
              CALL RANDU(K1,K2,XX)
              IF(XX.GT..5)IQQ=9
              IF(XX.LE..5)IQQ=2
  450 IF(IQ.EQ.2)IQQ=1
      IF(IQ.EQ.3)IQQ=7
      IF(IQ.EQ.4)IQQ=12
      IF(IQ.EQ.5)IQQ=15
      IF(IQ.EQ.6)IQQ=8
      IF(IQ.EQ.7)IQQ=3
      IF(IQ.EQ.8)IQQ=6
      IF(IQ.EQ.9)IQQ=1
      IF(IQ.EQ.10)IQQ=14
      IF(IQ.EQ.11)IQQ=13
      IF(IQ.EQ.12)IQQ=4
```

```
      IF(IQ.EQ.13)IQQ=11
      IF(IQ.EQ.14)IQQ=10
      IF(IQ.EQ.15)IQQ=6
      WRITE(2,*)CNT,IQ,IQQ,XX
      GO TO 800
  500 CONTINUE
C*****************************************************************
C*****************************************************************
C****************
C****************ERROR IN LSB.
C****************
      E4=E4+1.
      IF(IQ.NE.1)GO TO 550
              CALL RANDU(K1,K2,XX)
              IF(XX.GT..5)IQQ=8
              IF(XX.LE..5)IQQ=3
  550 IF(IQ.EQ.2)IQQ=6
      IF(IQ.EQ.3)IQQ=1
      IF(IQ.EQ.4)IQQ=14
      IF(IQ.EQ.5)IQQ=13
      IF(IQ.EQ.6)IQQ=2
      IF(IQ.EQ.7)IQQ=9
      IF(IQ.EQ.8)IQQ=1
      IF(IQ.EQ.9)IQQ=7
      IF(IQ.EQ.10)IQQ=12
      IF(IQ.EQ.11)IQQ=15
      IF(IQ.EQ.12)IQQ=10
      IF(IQ.EQ.13)IQQ=5
      IF(IQ.EQ.14)IQQ=4
      IF(IQ.EQ.15)IQQ=11
      WRITE(2,*)CNT,IQ,IQQ,XX
  800 ICNT=IFIX(CNT)
      ICNT=ICNT-((ICNT-1)/16)*16
      IOUT(ICNT)=IQQ
      IF(ICNT.EQ.16)WRITE(4,10)(IOUT(J),J=1,16)
  900 CONTINUE
      GO TO 100
C*****************************************************************
C*****************************************************************
C****************
C****************STATISTICS GENERATED.
C****************
  990 PC1=E1/CNT
      PC2=E2/CNT
      PC3=E3/CNT
      PC4=E4/CNT
      PTOT=PC1+PC2+PC3+PC4
      TYPE *,'              MSB    2ND MSB   3RD MSB    LSB'
      TYPE 20,E1,E2,E3,E4
      TYPE 40,PC1,PC2,PC3,PC4
      TYPE 50,PTOT,PERR
      CLOSE(UNIT=2)
```

```
      CLOSE(UNIT=3)
      CLOSE(UNIT=4)
      STOP
      END
C
C
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C      ********************************************
C      ********************************************
C      ***                                      ***
C      ***                                      ***
C      ***            R F W L . F T N           ***
C      ***                                      ***
C      ***                                      ***
C      ********************************************
C      ********************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE:  RFWL.FTN
C
C     BY:  JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTIVE TITLE: RECEIVER FINITE WORD LENGTH
C                              SIMULATION
C
C     DESCRIPTION:  THIS PROGRAM USES THE OUPTUT PARAMETER FILE
C                     OF PFWL.FTN, RENAMED CMDPARA.DAT, TO
C                     SET ALL THE OPTIONS OF THE RECEIVER.
C                     FOR THIS REASON, THE OPERATOR IS NOT
C                     PROMPTED FOR ANY INFORMATION DURING A RUN.
C                     THIS PROGRAM REPRESENTS THE FINITE WORD
C                     LENGTH SIMULATION OF THE RECEIVER FOR
C                     THE TDHS-ARC ALGORITHM.  THE WORD LENGTH
C                     HAS BEEN RESTRICTED TO 16 BITS.  THE
C                     TRANSMISSION RATE IS ASSUMED TO BE 16KB/S.
C
C*****************************************************************
C*****************************************************************
C
C
      INTEGER HD(40),Q(256),IZHAT(16),CLPSP(512),CLIPFI(16)
      INTEGER IPICH(400),SQ(16),FNAME1(16),P1,BUFSZ,WARE5
      DIMENSION SHAT(256),H(256),IFS(512)
      INTEGER*4 ISF,ITF,IA4,IB4
      INTEGER IVHAT(512),QCOMP(512)
C***************
C***************
C***************
      COMMON /CLIPP/ALPHA,ICLPP,ISENG(16),MSEC,ISEC
      COMMON /PRED/ALP,ALAD,ABIAS,EV,EP,G,B(12)
     1         ,NSPSAM,KQ,N,IVHAT(12),IRMSMN,IA(12)
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
```

```
C
C
      INTEGER MD(40),Q(256),IZMAT(16),CLPSP(512),CLIPFI(16)
      INTEGER IPICH(400),SQ(16),FNAME1(16),PI,SMPSZ,WARES
      DIMENSION SMAT(256),R(256),IFS(512)
      INTEGER*4 ISF,ITF,IA4,IB4
      INTEGER IVMAT(512),OCOMP(512)
C***************
C***************
C***************
      COMMON /CLIPP/ALPHA,ICLPP,ISENG(16),NSEC,ISEC
      COMMON /PRED/ALP,ALAD,ABIAS,EV,EP,G,B(12)
     1        ,BSPSAM,KQ,N,IVMAT(12),IRMSMN,IA(12)
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /FILT/HF(5),IXXX(9),IFSC,IFINC,NOFLT
      COMMON /FVE/ISF,ITF,INF,IAF,ICF,IQ1F,IQ2F,IQ3F
      COMMON /DBG/YL,YH
C*****************************************************************
C*****************************************************************
C***************
C***************FUNCTIONS:
C***************      MODM:DIVIDES ARGUMENT BY 16 AND RETURNS
C***************                REMAINDER.
C***************      MODM:RETURNS NEXT LARGER NUMBER WHICH IS
C***************                A FACTOR OF 16.
C***************
      MODM(K)=K-(K-1)/16*16
      MODM(K)=((K-1)/16)*16+16
C*****************************************************************
C*****************************************************************
C***************
C***************FORMATS USED IN MAIN PROGRAM.
C***************
   10 FORMAT(16A2)
   20 FORMAT(6I5,10X,40A1)
   30 FORMAT(40A1)
   40 FORMAT(16I2)
   50 FORMAT(16I5)
   60 FORMAT(///,40A1,//)
   70 FORMAT(////6X,' SAMPLE NUMBER ',6X,' PITCH PERIOD '//)
   80 FORMAT(6X,I5,1X,'-',1X,I5,10X,I3)
C*****************************************************************
C*****************************************************************
C***************
C***************INITIALIZE PARAMETERS.
C***************
      OPEN(UNIT=4,TYPE='OLD',NAME='CMDPARA.DAT')
      READ(4,10)FNAME1
      READ(4,*)INF,ICF,IAF,IQ1F,IQ2F,IQ3F
      INF=2**INF
      ICF=2**ICF
```

```
      IAF=2**IAF
      IQ1F=2**IQ1F
      IQ2F=2**IQ2F
      IQ3F=2**IQ3F
      ISF=2**15
      ITF=ISF-1
      READ(4,*)NBL
      READ(4,*)CLPP
      ICLPP=CLPP*FLOAT(ICF)
      READ(4,*)KBLK1,ITMIN,ITMAX
C***************
C***************INITIALIZATION SUBROUTINE.
C***************
      CALL INSTRT
C*****************************************************************
C*****************************************************************
C***************
C***************OPEN FILES AND WRITE HEADERS.
C***************
            OPEN(UNIT=1,TYPE='OLD',NAME='DECO.DAT')
               READ(1,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
               NSPSAM=NSAMP+NSAMP
      IF(ICSF.NE.1)GO TO 400
            OPEN(UNIT=3,TYPE='NEW',NAME='ZHAT.DAT',CARRIAGECONTROL='LIST')
               WRITE(3,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  400 IF(IOSF.NE.1)GO TO 410
            OPEN(UNIT=7,TYPE='NEW',NAME='SHAT.DAT',CARRIAGECONTROL='LIST')
               WRITE(7,20)NSENT,IRATE,NSPSAM,IUPPR,ILOWR,NTERMS,HD
  410 IF(IRSE.NE.1)GO TO 420
            OPEN(UNIT=9,TYPE='NEW',NAME='RSENGY.DAT',CARRIAGECONTROL='LIST')
               WRITE(9,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  420 IF(IRPF.NE.1)GO TO 430
            OPEN(UNIT=2,TYPE='NEW',NAME='RPITCH.DAT',CARRIAGECONTROL='LIST')
               WRITE(2,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  430 IF(IRCS.NE.1)GO TO 440
            OPEN(UNIT=10,TYPE='NEW',NAME='RCLPSP.DAT',CARRIAGECONTROL='LIST')
               WRITE(10,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  440 IF(IRFF.NE.1)GO TO 445
            OPEN(UNIT=11,TYPE='NEW',NAME='RFILTER.DAT',CARRIAGECONTROL='LIST')
               WRITE(11,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  445 IF(IPATH.EQ.1.OR.IPATH.EQ.2)GO TO 435
            OPEN(UNIT=12,TYPE='OLD',NAME='TZHAT.DAT')
               READ(12,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  435 IF(IPATH.EQ.2.OR.IPATH.EQ.4)GO TO 425
            OPEN(UNIT=13,TYPE='OLD',NAME='TPITCH.DAT')
               READ(13,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  425 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************INITIALIZE REMAINING PARAMETERS.
C***************
```

```
      ISEC=0            !SETS SIGNAL ENERGY COUTNER TO ZERO
      IFSC=0            !SETS FILTERED SPEECH COUNTER TO ZERO
      JJ=0
      NLEFT=0
      INC1=0
      INC2=0
      P1=0
      NADD=0
      ISENGY=0
      K1=773
      K2=119
      DO 450 JQ=1,100
  450         CALL RANDU(K1,K2,XX)
C*****************************************************************
C*****************************************************************
C***************
C***************SET UP MAIN LOOPS INITIAL SPEECH BLOCKS AND
C***************VARIALBLES.
C***************
      BUFSZ=2*ITMAX
      BUFSZ=MODM(BUFSZ)
      DO 480 I=1,ITMAX
              QCOMP(I)=0.
  480 CONTINUE
      IF(IPATH.LE.2)GO TO 470
              READ(12,500)(Q(J),J=1,BUFSZ)
              DO 460 I=1,BUFSZ
                      Q(J)=Q(J)*INF
  460         CONTINUE
  470 IF(IPATH.LE.2)READ(1,400)(Q(J),J=1,BUFSZ)
      NNEED=BUFSZ
      INC3=ITMAX
      ISTRTI=1
      IFNSHI=NNEED+ITMAX
      NSTOP=BUFSZ+ITMAX+16
C*****************************************************************
C*****************************************************************
C***************
C***************BEGIN MAIN LOOP.  INVERSE ARC ENGAGED FOR EACH
C***************SAMPLE AND RESULTING BLOCKS OF SPEECH ARE
C***************EXPANDED USING TDHS.
C***************
  495 DO 500 I=1,NNEED
              IF(IPATH.LE.2)CALL ARCR(Q(I),IY)
              IF(IPATH.GE.3)IY=Q(I)
              INC2=INC2+1
              INC1=MODN(INC2)
              QCOMP(INC3+I)=IY
              IY=FLOAT(IY)/FLOAT(INF)
              IZHAT(INC1)=IY
              IF(ICSF.NE.1)GO TO 500
                      IF(INC1.EQ.16)WRITE(3,500)(IZHAT(K),K=1,16)
```

```
  500 CONTINUE
      IF(IPATH.EQ.1.OR.IPATH.EQ.3)GO TO 350
C****************
C****************COMPRESSED SPEECH IS FILTERED BY SUBROUTINE.
C****************
      IF(NOFLT.EQ.1)GO TO 490
      DO 505 I=ISTRTI,IFNSHI
              IFS(I)=QCOMP(I)
  505 CONTINUE
      GO TO 485
  490 CALL IFLT(QCOMP,ISTRTI,IFNSHI,IFS)
C****************
C****************CLIP THE COMPRESSED SPEECH BY SUBROUTINE.
C****************
  485 CALL CLPPP(IFS,ISTRTI,IFNSHI,CLPSP,ISENGY)
C*********************************************************************
C*********************************************************************
C****************
C****************PITCH DETERMINED BY AUTOCORRELATION.
C****************
  510 IBIGG=-10000
      I2SUM=-10000
      DO 520 IT=ITMIN,ITMAX+1
              ISUM=0
              IBGN=ITMAX
              ISTP=3*KBLK1
              DO 525 J=IBGN,ISTP
                      IF(((CLPSP(J-IT).LT.0).AND.(CLPSP(J).LT.0)).OR.
     1                        ((CLPSP(J-IT).GT.0).AND.(CLPSP(J).GT.0)))
     2                        ISUM=ISUM+1
                      IF(((CLPSP(J-IT).LT.0).AND.(CLPSP(J).GT.0)).OR.
     1                        ((CLPSP(J-IT).GT.0).AND.(CLPSP(J).LT.0)))
     2                        ISUM=ISUM-1
  525         CONTINUE
              IF(I2SUM.GT.IBIGG.AND.I2SUM.GT.ISUM)NP=IT-1
              IF(I2SUM.GT.IBIGG.AND.I2SUM.GT.ISUM)IBIGG=I2SUM
              I2SUM=ISUM
  520 CONTINUE
  350 JJ=JJ+1
C****************
C****************RANDOM PITCH GENERATOR.
C****************
      JINC=MODN(JJ)
      IF(IPATH.EQ.2.OR.IPATH.EQ.4)GO TO 355
              IF(JINC.EQ.1)READ(13,500,END=365)(IPICH(J),J=1,16)
  360         NP=IPICH(JINC)
              GO TO 515
  365 NP=ITMIN
  355 IF(NP.GT.ITMIN)GO TO 515
              CALL RANDU(K1,K2,XX)
              NP=XX*(ITMAX-ITMIN)/2+ITMIN
              WRITE(5,288)JJ,NP
```

```
  515 IPICH(JINC)=NP
  288 FORMAT(6X,'NUMP=',I4,6X,'NP IS=',I4,12X,'RANDOM PITCH!!!!!!')
      IF(IRPF.NE.1)GO TO 530
      IF(JINC.EQ.16)WRITE(2,50)(IPICH(K),K=1,16)
  530 NP2=NP+NP
C***************
C***************WINDOW FORMED FOR SPEECH EXPANSION
C***************FROM PITCH GENERATED.
C***************
      DO 540 I=1,NP2
              H(I)=1.-FLOAT(I-1)/FLOAT(NP2-1)
  540 CONTINUE
I
C*****************************************************************
C*****************************************************************
C***************
C***************SPEECH IS EXPANDED.
C***************
  550 P1=ITMAX-NP
      NP3=NP2+NP
      DO 555 I=1,NP3
              IYHAT(I)=QCOMP(P1+I)
  555 CONTINUE
      DO 560 I=1,NP2
              IARG=NADD+I
              N1=MODN(IARG)
              IA4=IYHAT(I+NP)
              IB4=IYHAT(I)
              IA4=IA4-IB4
              IB4=ABS(IA4)
              IF(IB4.GT.ITF)TYPE *,' OVERFLOW EXPN ADD'
              IA4=FLOAT(IYHAT(I))+H(I)*FLOAT(IA4)
              IB4=ABS(IA4)
              IF(IB4.GT.ITF)TYPE *,' OVERFLOW EXPN ADD 2'
              SQ(N1)=FLOAT(IA4)/FLOAT(INF)
              IF(IOSF.NE.1)GO TO 560
              IF(N1.EQ.16)WRITE(7,50)(SQ(J),J=1,16)
  560 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************SHIFT ALL VECTORS APPROPRIATELY BASED ON PITCH
C***************VALUE USED TO EXPAND SPEECH.  SET UP VECTORS
C***************AND PARAMETERS FOR NEXT RUN THORUGH MAIN LOOP.
C***************
      NADD=NADD+NP+NP
      INC3=ITMAX-NP+NNEED+NLEFT
      ISTRTI=INC3+1
      NLEFT=NNEED-NP+NLEFT
      DO 565 I=1,INC3
              QCOMP(I)=QCOMP(I+NP)
              CLPSP(I)=CLPSP(I+NP)
```

```
                IFS(I)=IFS(I+NP)
  565 CONTINUE
      NNEED=MODM(NP)
      NTOT=INC3+NNEED
      IF(NTOT.GT.NSTOP)NNEED=NNEED-16
      IFNSNI=INC3+NNEED
      IF(NADD.GE.NSPSAM)GO TO 600
      IF(IPATH.LE.2)GO TO 595
                READ(12,500,END=580)(Q(I),I=1,NNEED)
                DO 570 I=1,NNEED
                        Q(I)=Q(I)*INF
  570           CONTINUE
  595 IF(IPATH.LE.2)READ(1,400,END=580)(Q(I),I=1,NNEED)
      GO TO 590
  580 DO 585 J=I,NNEED
                Q(I)=0
  585 CONTINUE
  590 GO TO 495
C***************
C***************AT END OF FILE, FINISH WRITING VALUES TO APPROPRATE
C***************FILES.
C***************
  600 IF(IOSF.NE.1)GO TO 610
      WRITE(7,500)(SQ(J),J=1,N1)
  610 IF(IRSE.NE.1)GO TO 620
      WRITE(9,500)(ISENG(J),J=1,INC1)
  620 IF(ICSF.NE.1)GO TO 630
      WRITE(3,500)(IZHAT(J),J=1,INC1)
  630 IF(IRPF.NE.1)GO TO 640
      WRITE(2,500)(IPICH(J),J=1,JINC)
  640 IF(IRCS.NE.1)GO TO 650
      WRITE(10,500)(CLPSP(J),J=1,MSEC)
  650 IF(IRFF.NE.1)GO TO 670
      WRITE(11,500)(IFS(J),J=1,IFINC)
  670 CONTINUE
      WRITE(5,*)IFSC,YL,YH
C*****************************************************************
C*****************************************************************
C***************
C***************CLOSE FILES.
C***************
      CLOSE(UNIT=1)
      CLOSE(UNIT=3)
      CLOSE(UNIT=4)
      CLOSE(UNIT=7)
      CLOSE(UNIT=2)
      CLOSE(UNIT=11)
      CLOSE(UNIT=10)
      STOP
      END
```

```
C
C
C
C
C     ************************************************************
C     *                                                          *
C     *     I N V E R S E     Q U A N T I Z E R                  *
C     *                                                          *
C     ************************************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE MODELS A FIXED INVERSE QUANTIZER.
C              THE INPUT IS THE QUANTIZER SEQUENCE AND THE OUTPUT
C              IS THE ESTIMATED ERROR FOR THE PREDICTED VALUE OF
C              THE SPEECH ESTIMATE.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE INVQUA(QQ,IEQ)
C***************
C***************
C***************
      INTEGER QQ
      INTEGER*4 ISF,ITF,IA4,IB4,IC4
C***************
C***************
      COMMON /RMS/IRMS
      COMMON /QUAN/SMIN,IT(51),IOUT(51),IEXPN(51),ISIZE,NQ
      COMMON /FWL/ISF,ITF,INF,IAF,ICF,IQ1F,IQ2F,IQ3F
C***************
C***************
      ISIGN=1
      IF(MOD(QQ,2).EQ.0)ISIGN=-1
      J=(QQ+2)/2
C*****************************************************************
C*****************************************************************
C***************
C***************DETERMINATION OF ESTIMATED ERROR.
C***************
      IEQ=FLOAT(ISIGN)*FLOAT(IOUT(J))*FLOAT(ISIZE)/FLOAT(ISF)
      SCALE=FLOAT(ISF)*FLOAT(INF)/(FLOAT(IQ1F)*FLOAT(IQ3F))
      IEQ=FLOAT(IEQ)*SCALE*2.
C*****************************************************************
C*****************************************************************
C***************
C***************THE STANDARD DEVIATION ESTIMATE IS UPDATED.
```

```
C***************
      ISIZE=FLOAT(IEXPN(J))*FLOAT(ISIZE)/FLOAT(ISF)
      SCALE=FLOAT(ISF)/FLOAT(IQ2F)
      ISIZE=FLOAT(ISIZE)*SCALE
      IRRMS=IRMS/2.
      IF(IQ3F.GT.INF)ISIZE=FLOAT(ISIZE)*(FLOAT(INF)/FLOAT(IQ3F))
      IF(INF.GT.IQ3F)IRRMS=FLOAT(IRRMS)*(FLOAT(IQ3F)/FLOAT(INF))
      ISIZE=IMAX0(ISIZE,IRRMS)
      IF(IQ3F.GT.INF)ISIZE=FLOAT(ISIZE)*(FLOAT(IQ3F)/FLOAT(INF))
      IF(INF.GT.IQ3F)ISIZE=FLOAT(ISIZE)*(FLOAT(INF)/FLOAT(IQ3F))
      RETURN
      END
```

```
C
C
C
C
C     ********************************
C     *                              *
C     *  I N I T I A L I Z A T I O N  *
C     *                              *
C     ********************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE INITIALIZES MOST OF THE PARAMETERS
C              BY READING FROM CMDPARA.DAT, THE RENAMED OUTPUT OF
C              PFWL.FTN.  OTHER PARAMETERS ARE ALSO INITIALIZED
C              TO THEIR APPROPRIATE STARTING VALUES.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE INSTRT
C***************
C***************
C***************
      INTEGER WARES
      INTEGER*4 ISF,ITF,IA4,IB4
      DIMENSION T(51),OUT(51),EXPN(51)
C***************
C***************
      COMMON /CLIPP/ALPHA,ICLPP,ISENG(16),MSEC,ISEC
      COMMON /RMS/IRMS
      COMMON /QUAN/SMIN,IT(51),IOUT(51),IEXPN(51),ISIZE,NQ
      COMMON /PRED/ALP,ALAD,ABIAS,EV,EP,G,B(12)
     1        ,NSPSAM,KQ,N,IVHAT(12),IRMSMN,IA(12)
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /FILT/HF(5),IXXX(9),IFSC,IFINC,NOFLT
      COMMON /FWL/ISF,ITF,INF,IAF,ICF,IQ1F,IQ2F,IQ3F
      COMMON /DBG/YL,YH
      COMMON /IOCNT/IOCNT
      COMMON /GAIN/GAIN,IEQ
C***************
C***************
      IOCNT=0
      YL=10000
      YH=0
      READ(4,*)ALP,ALAD,ABIAS,G,N,RMSMIN,SMIN
      IRMSMN=FLOAT(INF)*RMSMIN
```

```
      READ(4,*)KQ
      NQ=KQ/2
      NQQQ=NQ+1
C*****************************************************************
C*****************************************************************
C***************
C***************QUANTIZER VALUES ARE SET AND APPROPRIATELY
C***************SHIFTED.
C***************
      READ(4,*)(T(I),I=1,NQQQ)
      READ(4,*)(OUT(I),I=1,NQQQ)
      READ(4,*)(EXPN(I),I=1,NQQQ)   !EXPANSION FACTORS
      DO 400 I=1,NQQQ
              IOUT(I)=OUT(I)*FLOAT(IQ1F)*SMIN
              IEXPN(I)=EXPN(I)*FLOAT(IQ2F)
  400 CONTINUE
      READ(4,*)ALPHA   !SIGNAL ENERGY PARAMETER
C***************FILE MANAGEMENT FLAGS.
      READ(4,*)ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
      READ(4,*)NOFLT   !FILTER FLAG
      READ(4,*)(HF(I),I=1,5)   !FILTER COEFFICIENTS
      READ(4,*)IPATH  !PATH DETERMINES MAJOR ROUTE OF PROGRAM
      READ(4,*)(B(I),I=1,N)  !INITIAL PREDICTOR COEFFICIENTS
      READ(4,*)GAIN
      ISIZE=100*IQ3F  !INITIAL STANDARD DEVIATION ESTIMATE
      DO 118 I=1,12
      IVHAT(I)=0
      B(I)=B(I)*FLOAT(IAF)
      IA(I)=B(I)
  118 CONTINUE
      ABIAS=ABIAS*FLOAT(IAF)
      DO 119 I=1,9
              IXXX(I)=0
  119 CONTINUE
      IRMS=IRMSMN
      RETURN
      END
```

```
C
C
C
C
C     ************************************
C     *                                  *
C     *     A R C   R E C E I V E R      *
C     *                                  *
C     ************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION:  SUBROUTINE MODELS THE ADAPTIVE RESIDUAL
C             CODER.  THE INPUT IS A QUANTIZATION LEVEL AND
C             A  SPEECH SAMPLE ESTIMATE IS RETURNED.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE ARCR(Q,IVHAT1)
C***************
C***************
C***************
      INTEGER Q
      INTEGER*4 ISF,ITF,IA4,IB4
C***************
C***************
      COMMON /RMS/IRMS
      COMMON /PRED/ALP,ALAD,ABIAS,EV,EP,G,B(12)
     1         ,NSPSAM,KQ,N,IVHAT(12),IRMSMN,IA(12)
      COMMON /FWL/ISF,ITF,INF,IAF,ICF,IQ1F,IQ2F,IQ3F
      COMMON /IOCNT/IOCNT
      COMMON /GAIN/GAIN,IEQ
C*****************************************************************
C*****************************************************************
C***************
C***************PREDICTION IS COMPUTED AND AN ERROR SAMPLE
C***************GENERATED BY THE INVERSE QUANTIZER IS ADDED TO
C***************OBTAIN A SPEECH ESTIMATE.
C***************
      IOCNT=IOCNT+1
      PRE=0.
      DO 120 I=1,N
              PRE=PRE+FLOAT(IA(I))*FLOAT(IVHAT(I))/FLOAT(IAF)
  120 CONTINUE
      IPRE=PRE
      IRMS=ALP*(FLOAT(IRMS)-FLOAT(IRMSMN))+FLOAT(IRMSMN)
      IRMS=FLOAT(IRMS)+(1.-ALP)*ABS(FLOAT(IVHAT(1)))
```

```
C****************
C****************ERROR SAMPLE GENERATED BY SUBROUTINE.
C****************
      CALL INVQUA(Q,IEQ)
C****************
C****************UPDATE PREDICTOR VALUES.
C****************
      DO 125 I=1,N
      J=N+2-I
  125 IVHAT(J)=IVHAT(J-1)
      IVHAT(1)=IPRE+IEQ
      IVHAT1=IVHAT(1)
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,*)IOCNT
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,*)IPRE,IEQ,IVHAT1
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,244)
  244 FORMAT('      ')
C*****************************************************************
C*****************************************************************
C****************
C****************ADAPTATION OF PREDICTOR COEFFICIENTS IS ACCOMPLISHED
C****************BY A FIXED PERCENTAGE COMPUTATION.
C****************
      IF(G.EQ.0.0)GO TO 131
      DO 100 I=1,N
              IA(I)=IA(I)-B(I)
  100 CONTINUE
      DO 130 I=1,N
              SVL=ISIGN(1,IEQ)*ISIGN(1,IVHAT(I+1))
              IF((IEQ.EQ.0.OR.IVHAT(I+1).EQ.0).AND.IA(I).GE.0)SVL=-1.
              IF((IEQ.EQ.0.OR.IVHAT(I+1).EQ.0).AND.IA(I).LT.0)SVL=+1.
              AINT=SVL*AMAX1(FLOAT(IABS(IA(I)))*G,ABIAS)
              IA(I)=ALAD*(IA(I)+AINT)
  130 CONTINUE
      DO 150 I=1,N
              IA(I)=IA(I)+B(I)
  150 CONTINUE
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,*)ERR,IRMS
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,*)(IA(I),I=1,4)
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,244)
      IF(IOCNT.GT.165.AND.IOCNT.LE.220)WRITE(6,244)
  131 RETURN
      END
```

```
C
C
C
C
C     ***************************************************
C     *                                                 *
C     *      C L I P P I N G                            *
C     *                                                 *
C     ***************************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION:  SUBROUTINE ACCEPTS BLOCK OF COMPRESSED
C             SPEECH SAMPLES AS INPUT AND THREE VALUE CENTER
C             CLIPS THIS SPEECH BY A FIXED PERCENTAGE OF THE
C             COMPUTED SIGNAL ENERGY.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE CLPPP(IFS,STRT,FNSH,CLPSP,ISENGY)
C***************
C***************
C***************
      INTEGER*2 CLPSP(512),CLPLVL,CLM
      INTEGER*4 ISF,ITF,IA4,IB4
      DIMENSION IFS(512)
      INTEGER CLIPFI(16),STRT,FNSH
C***************
C***************
      COMMON /CLIPP/ALPHA,ICLPP,ISENG(16),MSEC,ISEC
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /FWL/ISF,ITF,INF,IAF,ICF,IQ1F,IQ2F,IQ3F
C***************
C***************
      DO 21 I=STRT,FNSH
              MSEC=MOD(ISEC,16)+1
              ISEC=ISEC+1
              IFSP=IFS(I)
              IA4=FLOAT(ISENGY)*ALPHA+(1.-ALPHA)*ABS(FLOAT(IFSP))
              IB4=ABS(IA4)
              IF(IB4.GT.ITF)TYPE *,' OVERFLOW ISENGY'
              ISENGY=FLOAT(IA4)
              ISENG(MSEC)=ISENGY/INF
              IF(IRSE.NE.1)GO TO 20
              IF(MSEC.EQ.16)WRITE(9,10)(ISENG(JJ),JJ=1,16)
   20         IA4=FLOAT(ICLPP)*FLOAT(ISENGY)/FLOAT(ISF)
```

```
      SCALE=FLOAT(ISF)/FLOAT(ICF)
      IA4=FLOAT(IA4)*SCALE
      IB4=ABS(IA4)
      IF(IB4.GT.ITF)TYPE *,' OVERFLOW CLPP'
      CLPLVL=FLOAT(IA4)
      CLM=-CLPLVL
      CLPSP(I)=0
      IF(IFSP.GT.CLPLVL)CLPSP(I)=+1
      IF(IFSP.LT.CLM)CLPSP(I)=-1
      CLIPFI(MSEC)=CLPSP(I)
      IF(IRCS.NE.1)GO TO 21
      IF(MSEC.EQ.16)WRITE(10,10)(CLIPFI(JJ),JJ=1,16)
21 CONTINUE
10 FORMAT(16I5)
   RETURN
   END
```

```
C
C
C
C
C     ***************************************************
C     *                                                 *
C     *     C L I P P I N G     F I L T E R             *
C     *                                                 *
C     ***************************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION:  SUBROUTINE ACCEPTS BLOCK OF COMPRESSED
C             SPEECH SAMPLES AND FILTERS THEM WITH A 9TH
C             ORDER LINEAR PHASE FINITE IMPULSE RESPONSE
C             FILTER.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE IFLT(QCOMP,STRT,FNSH,IFS)
C***************
C***************
C***************
      DIMENSION IFS(512),IFO(16)
      INTEGER STRT,FNSH,QCOMP(512)
      INTEGER*4 ISF,ITF,IA4,IB4
C***************
C***************
      COMMON /FILT/HF(5),IXXX(9),IFSC,IFINC,NOFLT
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /FWL/ISF,ITF,INF,IAF,ICF,IQ1F,IQ2F,IQ3F
      COMMON /DBG/YL,YH
C***************
C***************
   1# FORMAT(16I5)
      DO 5## I=STRT,FNSH
              IFINC=MOD(IFSC,16)+1
              IFSC=IFSC+1
              DO 1## J=2,9
                      JIJ=11-J
                      IXXX(JIJ)=IXXX(JIJ-1)
  1##         CONTINUE
              IXXX(1)=QCOMP(I)
              IF(IXXX(1).GT.YH)YH=IXXX(1)
              IF(IXXX(1).LT.YL)YL=IXXX(1)
              IFS(I)=#
```

```
            DO 200 KIK=1,4
                    IA4=FLOAT(IFS(I))+HF(KIK)*(FLOAT(IXXX(KIK))+
     1                          FLOAT(IXXX(10-KIK)))
                    IB4=ABS(IA4)
                    IF(IB4.GT.ITF)TYPE *,' OVERFLOW IFS LOOP'
                    IFS(I)=FLOAT(IA4)
  200       CONTINUE
                    IA4=FLOAT(IFS(I))+HF(5)*FLOAT(IXXX(5))
                    IB4=ABS(IA4)
                    IF(IB4.GT.ITF)TYPE *,' OVERFLOW IFS'
                    IFS(I)=FLOAT(IA4)
            IFO(IFINC)=IFS(I)/INF
            IF(IRFF.NE.1)GO TO 500
                    IF(IFINC.EQ.16)WRITE(11,10)(IFO(JJJ),JJJ=1,16)
  500 CONTINUE
      RETURN
      END
C
C
C
C
C*****************************************************************
C*****************************************************************
```

APPENDIX B

UNRESTRICTED WORD LENGTH SOURCE FILE

```
C
C
C
C
C       ***********************************************
C       ***********************************************
C       ***                                         ***
C       ***                                         ***
C       ***            P 1 6 F . F T N              ***
C       ***                                         ***
C       ***                                         ***
C       ***********************************************
C       ***********************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE: P16F.FTN
C
C     BY: JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTIVE TITLE: PARAMETER BUILDER FOR 16KB/S NON-FINITE
C                        WORD LENGTH SIMULATION.
C
C     DESCRIPTION:  THIS PROGRAM GENERATES A PARAMETER FILE TO
C             BE USED WITH THE NON-FINITE WORD LENGTH SIMULATIONS.
C             IT OUTPUTS A FILE OF PARAMETER VALUES NAMED
C             PARAM.DAT.  THE INTERACTIVELY SELECTED PARAMETER
C             LIST IS EQUIVALENT TO THOSE OF PFWL.FTN EXCLUDING
C             THE FWL SCALING FACTORS.
C
C*****************************************************************
C*****************************************************************
```

```
C
C
      INTEGER WARE,UU,VV,ANS1(6),CHK,ANS2(3),ANS(9),ANS3(8)
      INTEGER*2 FNAME(16)
      REAL MSF,LFI(51),LMX,LOEXPN,HIEXPN,HIDELT,LODELT,EXPN(51),M
      REAL OUT(51),KK,LL,T(51),HF(5),H(5,4),AB(12)
C***************
C***************THE FOLLOWING DATA IS THE FILTER COEFFICIENTS FOR
C***************THE FILTERS THAT ARE INTERACTIVELY EMPLOYED BY
C***************THE SIMULATIONS.
C***************
      DATA H/.0341,.0125,.0139,.0149,.0151,.0560,.0720,.1009,
     1        .1219,.1295,.0376,.0214,.0252,.0277,.0285,.0758,.1549,.2344,
     2        .3188,.3356/
C***************
C***************PARAM.DAT IS THE OUTPUT FILE; CONTAINS THE PARAMETERS
C***************FOR THE SIMULATIONS.
C***************
      OPEN(UNIT=3,TYPE='NEW',NAME='PARAM.DAT',CARRIAGECONTROL='LIST')
      TYPE *,' ENTER THE ORIGINAL SPEECH FILE NAME:[<32CHAR]'
      ACCEPT 9,FNAME
    9 FORMAT(16A2)
C*****************************************************************
C*****************************************************************
C***************
C***************INITIALIZATION OF ALL PROGRAM PARAMETERS.
C***************
      NBL=64
      ALPHA=.992
      ALP=0.97
      ALAD=0.95
      ABIAS=.001
      G=0.15
      N=4
      RMSMIN=35.0
      SMIN=0.3
      MSF=8.
      KQ=15
      B=9.
      LOEXPN=0.5
      HIEXPN=5.0
      CLPP=1.5
      KBLK1=110
      ITMIN=30
      ITMAX=110
      ITCS=0
      ITPF=0
      ITSE=0
      ITSF=0
      IRCS=0
      IRPF=0
      IRSE=0
```

```
      ICSF=0
      ITFF=0
      IRFF=0
      IOSF=1
      AB(1)=1.793182
      AB(2)=-1.400654
      AB(3)=0.566023
      AB(4)=-0.146575
      DO 53 I=5,12
              AB(I)=0.0
   53 CONTINUE
      GAIN=0.950
C*****************************************************************
C*****************************************************************
C****************
C****************THE FOLLOWING BLOCK SETS UP THE LISTED PARAMETERS
C****************
  790 DO 795 I=1,9
              ANS(I)=0
  795 CONTINUE
      TYPE *,' THE FOLLOWING ARE THE CURRENT VALUES'
      WRITE(5,100)ALP,ABIAS,ALPHA,G,N,RMSMIN,SMIN,CLPP,KBLK1,ITMIN,
     1        ITMAX,ALAD,NBL
  100 FORMAT(/6X,'ALP=',F6.3/6X,'ABIAS=',F6.3/6X,'ALPHA=',F6.3/6X,'G='
     1        ,F5.3/6X,'N=',I2/6X,'RMSMIN=',F5.1/6X,'SMIN=',F5.2/6X,'CLPP='
     2        ,F5.2/6X,'KBLK1=',I3,2X,'ITMIN=',I3,2X,'ITMAX=',I3/6X,'ALAD='
     3        ,F5.3/6X,'NBL=',I4/)
      TYPE *,' WHICH DO YOU WANT TO CHANGE?'
      TYPE *,'                    0=NO (MORE) CHANGE(S)'
      TYPE *,'                    1=ALP'
      TYPE *,'                    2=ABIAS'
      TYPE *,'                    3=ALPHA'
      TYPE *,'                    4=G'
      TYPE *,'                    5=N'
      TYPE *,'                    6=RMSMIN'
      TYPE *,'                    7=SMIN'
      TYPE *,'                    8=CLPP'
      TYPE *,'                    9=KBLK1,ITMIN,ITMAX'
      TYPE *,' TYPE--[SELECTION] (SELECTION) (SELECTION)...--UP TO 9 CHOICES'
      ACCEPT 780,ANS
  780 FORMAT(9(I1,1X))
      CHK=0
      DO 770 I=1,9
              CHK=CHK+ANS(I)
  770 CONTINUE
      IF(CHK.EQ.0)GO TO 900
      ISELEC=1
  197 DO 199 I=1,9
              IF(ANS(I).EQ.ISELEC)GO TO (201,202,203,204,205,206,207,208
     1                    ,209),ISELEC
  199 CONTINUE
      GO TO 193
```

```
  2#1 TYPE *,' ENTER NEW VALUE OF "ALP"'
      ACCEPT *,ALP
      GO TO 193
  2#2 TYPE *,' ENTER NEW VALUE OF "ABIAS"'
      ACCEPT *,ABIAS
      GO TO 193
  2#3 TYPE *,' ENTER NEW VALUE OF "ALPHA"'
      ACCEPT *,ALPHA
      GO TO 193
  2#4 TYPE *,' ENTER NEW VALUE OF "G"'
      ACCEPT *,G
      GO TO 193
  2#5 TYPE *,' ENTER NEW VALUE OF "N"'
      ACCEPT *,N
      GO TO 193
  2#6 TYPE *,' ENTER NEW VALUE OF "RMSMIN"'
      ACCEPT *,RMSMIN
      GO TO 193
  2#7 TYPE *,' ENTER NEW VALUE OF "SMIN"'
      ACCEPT *,SMIN
      GO TO 193
  2#8 TYPE *,' ENTER NEW VALUE OF "CLPP"'
      ACCEPT *,CLPP
      GO TO 193
  2#9 TYPE *,' ENTER NEW VALUES OF "KBLK1 ITMIN ITMAX"'
      ACCEPT *,KBLK1,ITMIN,ITMAX
  193 ISELEC=ISELEC+1
      IF(ISELEC.LE.9)GO TO 197
      GO TO 79#
  9## CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************THE FOLLOWING BLOCK SETS UP THE PARAMETERS OF THE
C***************ADAPTIVE RESIDUAL CODER.  IT GENERATES THE THRESH-
C***************OLDS, SCALING FACTORS, AND EXPANSION FACTORS USING
C***************THE PARAMETERS SPECIFIED BY THE USER.
C***************
   12 DO 17 I=1,6
            ANS1(I)=#
   17 CONTINUE
      DO 18 I=1,6
            IF(ANS1(I).EQ.1)GO TO 3
   18 CONTINUE
      GO TO 1#
    3 TYPE *,'ENTER THE MAXIMUM SCALING FACTOR(>#)'
      ACCEPT *,MSF
      IF (MSF.LE.#.)GO TO 3
   1# DO 19 I=1,6
            IF(ANS1(I).EQ.2) GO TO 29
   19 CONTINUE
      GO TO 2#
```

```
   29 TYPE *,'ENTER THE NUMBER OF QUANT LEVELS (5<NUM<101) AND ODD'
      ACCEPT *,KQ
      IF (KQ.LT.5.OR.KQ.GT.101)GO TO 10
   20 NQ=KQ/2
      NQQQ=NQ+1
C***************
C***************DETERMINATION OF THE QUANTIZER PARAMETER VALUES.
C***************FIRST, THE SCALING FACTORS ARE GENERATED.
C***************
      OUT(NQQQ)=MSF
      LMX=(ALOG10(MSF))
      UP=LMX/(FLOAT(NQ))
      LFI(1)=0.
      DO 40 I=2,NQ
              LFI(I)=LFI(I-1)+UP
   40 CONTINUE
      KK=MSF+1
      KK=ALOG10(KK)
      LL=LFI(NQ)
      KK=KK/LL
      BMAX=10.**KK
      DO 214 I=1,6
              IF(ANS1(I).EQ.3)GO TO 213
  214 CONTINUE
      GO TO 42
  213 WRITE(5,212)BMAX,B
  212 FORMAT(' THE MAX BASE ALLOWED IS ',F5.2,' AND THE CURRENT BASE '
     1        ,'IS ',F5.2)
      TYPE *,' ENTER BASE OF LOG FNC USED FOR SCALING FACTORS(>0)'
      ACCEPT *,B
   42 IF(B.GT.0.AND.B.LT.BMAX)GO TO 43
      TYPE *,' THE BASE CHOSEN WAS OUTSIDE THE ALLOWED RANGE. TRY AGAIN!'
      GO TO 213
   43 DO 50 I=1,NQ
              M=LFI(I)
              OUT(I)=B**M-1.
   50 CONTINUE
      DO 60 I=1,NQ
              T(I)=(OUT(I)+OUT(I+1))/2.
   60 CONTINUE
      DO 61 I=1,6
              IF(ANS1(I).EQ.4)GO TO 62
   61 CONTINUE
      GO TO 65
C***************DETERMINATION OF EXPANSION FACTORS.
   62 TYPE *,' ENTER THE LOW EXPANSION FACTOR'
      ACCEPT *,LOEXPN
   65 DO 66 I=1,6
              IF(ANS1(I).EQ.5)GO TO 67
   66 CONTINUE
      GO TO 75
   67 TYPE *,' ENTER THE HIGH EXPANSION FACTOR'
```

```
      ACCEPT *,HIEXPN
   75 MID=NQ/2+1
      EXPN(MID)=1.
      EXPN(MID+1)=1.
      EXPN(MID-1)=1.
      EXPN(1)=LOEXPN
      EXPN(NQQQ)=HIEXPN
      HIDELT=(HIEXPN-1.)/(FLOAT(NQ-MID+1))
      LODELT=(1.-LOEXPN)/(FLOAT(MID-2))
      XX=HIDELT+1.
      YY=LODELT+LOEXPN
      UU=MID-2
      VV=MID+2
      DO 110 I=2,UU
              EXPN(I)=YY
              YY=YY+LODELT
  110 CONTINUE
      DO 120 I=VV,NQ
              EXPN(I)=XX
              XX=XX+HIDELT
  120 CONTINUE
C***************INTERMEDIATE PRINTOUT OF QUANTIZER PARAMETERS.
      TYPE *,'WHERE SHOULD THE PRINTOUT GO?[TERM=0,PRINT=1]'
      ACCEPT *,I
      IF (I.EQ.0) WARE=5
      IF (I.EQ.1) WARE=6
      WRITE(5,100)ALP,ABIAS,ALPHA,G,N,RMSMIN,SMIN,
     1                CLPP,KBLK1,ITMIN,ITMAX,ALAD,NBL
      WRITE(5,160)KQ,NQQQ,MSF,B,LOEXPN,HIEXPN
      IF(I.EQ.1)WRITE(WARE,100)ALP,ABIAS,ALPHA,G,N,RMSMIN,SMIN,
     1                CLPP,KBLK1,ITMIN,ITMAX,ALAD,NBL
      IF(I.EQ.1)WRITE(WARE,160)KQ,NQQQ,MSF,B,LOEXPN,HIEXPN
  160 FORMAT(//' THE FOLLOWING IS A LIST OF SCALING FACTORS,'/
     1        ' EXPANSION FACTORS, AND THRESHOLDS FOR A ',I2,' LEVEL'/
     2        ' QUANTIZER(',I2,' ON A SIDE), WITH A MAXIMUM SCALING FACTOR'/
     3        ' =',F7.4,'.  A LOG SCALE WITH A BASE=',F7.4,'WAS USED.'/
     4        ' THE HIGH AND LOW EXPANSION FACTORS WERE SET AT ',F7.4/
     5  ' AND',F7.4,' .')
      WRITE(WARE,180)
      DO 190 I=1,NQQQ
              WRITE(WARE,200)I,OUT(I),T(I),EXPN(I)
  190 CONTINUE
  180 FORMAT(/3X,'LEVEL NUMBER',4X,'SCALING FACTORS',4X,'THRESHOLDS',
     1        4X,'EXPANSION FACTORS'/)
  200 FORMAT(6X,I2,17X,F7.4,9X,F7.4,10X,F7.4)
      TYPE *,'DO YOU WANT TO CHANGE ANY OF THESE?[YES=1,NO=0]'
      ACCEPT *,I
      IF(I.EQ.0)GO TO 955
      TYPE *,' THE FOLLOWING ARE THE CURRENT VALUES'
      WRITE(5,111)MSF,KQ,B,LOEXPN,HIEXPN
  111 FORMAT(/6X,'MSF=',F5.2/6X,'KQ=',I2/6X,'B=',F5.2/6X,'LOEXPN=',F5.2
     1        /6X,'HIEXPN=',F5.2/)
```

```
      TYPE *,' WHICH DO YOU WANT TO CHANGE?'
      TYPE *,'          0=NO (MORE) CHANGE(S)'
      TYPE *,'          1=MSF'
      TYPE *,'          2=KQ'
      TYPE *,'          3=B'
      TYPE *,'          4=LOEXPN'
      TYPE *,'          5=HIEXPN'
      TYPE *,' TYPE--[SELECTION] (SELECTION) (SELECTION)...--UP TO 6 CHOICES'
      ACCEPT 59,ANS1
   59 FORMAT(6(I1,1X))
C*****************************************************************
C*****************************************************************
C***************
C***************OUTPUT FILES ARE CHOSEN.
C***************
      CHK=0
      DO 221 I=1,6
              CHK=CHK+ANS1(I)
  221 CONTINUE
      IF(CHK.EQ.0)GO TO 955
      GO TO 17
  955 CONTINUE
  423 DO 426 I=1,3
              ANS2(I)=0
  426 CONTINUE
      TYPE *,' THE FOLLOWING INDICATE WHICH SPEECH AND NONPRINTABLE FILES'
      TYPE *,' THIS COMMAND FILE WILL OUTPUT [0=NOT OUTPUT, 1=WILL BE OUTPUT'
      TYPE 500
      TYPE 510,ITSF
      TYPE 520,ICSF
      TYPE 530,IOSF
  500 FORMAT('          0)NO CHANGES DESIRED')
  510 FORMAT('          1)TRANSMITTER STATISTICS FILE        SET=',I1)
  520 FORMAT('          2)COMPRESSED SPEECH FILE             SET=',I1)
  530 FORMAT('          3)OUTPUT SPEECH FILE                 SET=',I1)
      TYPE *,' TO CHANGE ANY OR ALL OF THESE TYPE---'
      TYPE *,' [SELECTION] (SELECTION) (SELECTION)...--UP TO 3 CHOICES'
      ACCEPT 535,ANS2
  535 FORMAT(3(I1,1X))
      CHK=0
      DO 540 I=1,3
              CHK=CHK+ANS2(I)
  540 CONTINUE
      IF(CHK.EQ.0)GO TO 575
      IFILE=ITSF
      ISELEC=1
  545 DO 550 I=1,3
              IF(ANS2(I).EQ.ISELEC)GO TO 555
  550 CONTINUE
      GO TO 570
  555 IF(IFILE.NE.0)GO TO 560
      IFILE=1
```

```
      GO TO 565
560   IFILE=0
565   IF(ISELEC.EQ.1)ITSF=IFILE
      IF(ISELEC.EQ.2)ICSF=IFILE
      IF(ISELEC.EQ.3)IOSF=IFILE
570   ISELEC=ISELEC+1
      IF(ISELEC.EQ.2)IFILE=ICSF
      IF(ISELEC.EQ.3)IFILE=IOSF
      IF(ISELEC.LE.3)GO TO 545
      GO TO 423
575   CONTINUE
580   DO 585 I=1,8
              ANS3(I)=0
585   CONTINUE
      TYPE *,' THE FOLLOWING INDICATE WHICH PLOTTABLE FILES WILL BE'
      TYPE *,' OUTPUT BY THIS COMMAND FILE'
      TYPE *,' [0=NOT OUTPUT, 1=WILL BE OUTPUT]'
      TYPE 500
      TYPE 590,ITFF
      TYPE 595,ITSE
      TYPE 600,ITCS
      TYPE 605,ITPF
      TYPE 610,IRFF
      TYPE 615,IRSE
      TYPE 620,IRCS
      TYPE 625,IRPF
590   FORMAT('          1)TRANSMITTER FILTERED SPEECH     SET=',I1)
595   FORMAT('          2)TRANSMITTER SIGNAL ENERGY       SET=',I1)
600   FORMAT('          3)TRANSMITTER CLIPPED SPEECH      SET=',I1)
605   FORMAT('          4)TRANSMITTER PITCH FILE          SET=',I1)
610   FORMAT('          5)RECEIVER FILTERED SPEECH        SET=',I1)
615   FORMAT('          6)RECEIVER SIGNAL ENERGY          SET=',I1)
620   FORMAT('          7)RECEIVER CLIPPED SPEECH         SET=',I1)
625   FORMAT('          8)RECEIVER PITCH FILE             SET=',I1)
      TYPE *,' TO CHANGE ANY OR ALL OF THESE TYPE---'
      TYPE *,' [SELECTION] (SELECTION) (SELECTION)...--UP TO 8 CHOICES'
      ACCEPT 630,ANS3
630   FORMAT(8(I1,1X))
      CHK=0
      DO 635 I=1,8
              CHK=CHK+ANS3(I)
635   CONTINUE
      IF(CHK.EQ.0)GO TO 685
      IFILE=ITFF
      ISELEC=1
640   DO 645 I=1,8
              IF(ANS3(I).EQ.ISELEC)GO TO 650
645   CONTINUE
      GO TO 675
650   IF(IFILE.NE.0)GO TO 655
      IFILE=1
      GO TO 665
```

```
  655 IFILE=0
  665 IF(ISELEC.EQ.1)ITFF=IFILE
      IF(ISELEC.EQ.2)ITSE=IFILE
      IF(ISELEC.EQ.3)ITCS=IFILE
      IF(ISELEC.EQ.4)ITPF=IFILE
      IF(ISELEC.EQ.5)IRFF=IFILE
      IF(ISELEC.EQ.6)IRSE=IFILE
      IF(ISELEC.EQ.7)IRCS=IFILE
      IF(ISELEC.EQ.8)IRPF=IFILE
  675 ISELEC=ISELEC+1
      IF(ISELEC.EQ.2)IFILE=ITSE
      IF(ISELEC.EQ.3)IFILE=ITCS
      IF(ISELEC.EQ.4)IFILE=ITPF
      IF(ISELEC.EQ.5)IFILE=IRFF
      IF(ISELEC.EQ.6)IFILE=IRSE
      IF(ISELEC.EQ.7)IFILE=IRCS
      IF(ISELEC.EQ.8)IFILE=IRPF
      IF(ISELEC.LE.8)GO TO 640
      GO TO 580
  685 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************CHOICE OF FILTERS.
C***************
      TYPE *,' DO YOU WANT TO FILTER THE PITCH SPEECH?'
      TYPE *,' [YES=1,NO=0]'
      ACCEPT *,NOFLT
      IF(NOFLT.EQ.0)GO TO 700
      TYPE *,' WHICH FILTER DO YOU WANT?'
      TYPE *,'          1)8KHZ INPUT SPEECH WITH FILTER CUTOFF FREQ=500HZ'
      TYPE *,'          2)8KHZ INPUT SPEECH WITH FILTER CUTOFF FREQ=1KHZ'
      TYPE *,'          3)6.4KHZ INPUT SPEECH WITH FILTER CUTOFF FREQ=500HZ'
      TYPE *,'          4)6.4KHZ INPUT SPEECH WITH FILTER CUTOFF FREQ=1KHZ'
      ACCEPT *,I
      DO 660 J=1,5
            HF(J)=H(J,I)
  660 CONTINUE
      GO TO 710
  700 DO 710 J=1,5
            HF(J)=0
  710 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************CHOICE OF PROGRAM PATHS.
C***************
      TYPE *,' WHICH PROGRAM PATH DO YOU WANT TO TAKE?'
      TYPE *,'          1)TDHS-ARC WITH PITCH PASSED TO THE RCVR'
      TYPE *,'          2)TDHS-ARC WITH PITCH INDEPENDENTLY GENERATED'
      TYPE *,'                  IN THE RCVR'
      TYPE *,'          3)TDHS ONLY WITH PITCH PASSED TO THE TDHE ALGORITHM'
```

```
      TYPE *,'        4)TDHS ONLY WITH PITCH INDEPENDENTLY GENERATED'
      TYPE *,'                    IN THE TDHE ALGORITHM'
      ACCEPT *,IPATH
C*****************************************************************
C*****************************************************************
C***************
C***************CHOICE OF INITIAL PREDICTOR COEFFICIENTS AND GAIN
C***************
  84# TYPE *,' THE FOLLOWING ARE THE INITIAL PREDICTOR'
      TYPE *,' COEFFICIENTS AND PREDICTOR GAIN'
      DO 8## I=1,N
            TYPE 81#,I,AB(I)
  8## CONTINUE
      TYPE 82#,GAIN
  81# FORMAT('      B(',I2,')=',2X,F8.4)
  82# FORMAT('      GAIN=',2X,F8.4)
      TYPE *,' DO YOU WANT TO ENTER NEW VALUES?'
      TYPE *,' YES=1      NO=#'
      ACCEPT *,NAI
      IF(NAI.EQ.#)GO TO 83#
      TYPE *,' ENTER THE NEW INITIAL PREDICTOR COEFFICIENTS'
      ACCEPT *,(AB(I),I=1,N)
      TYPE *,' ENTER THE NEW GAIN FACTOR'
      ACCEPT *,GAIN
      GO TO 84#
C*****************************************************************
C*****************************************************************
C***************
C***************OUTPUT BLOCK.
C***************
  83# WRITE(3,9)(FNAME(J),J=1,16)
      WRITE(3,*)NBL
      WRITE(3,*)CLPP
      WRITE(3,*)KBLK1,ITMIN,ITMAX
      WRITE(3,*)ALP,ALAD,ABIAS,G,N,RMSMIN,SMIN
      WRITE(3,*)KQ
      WRITE(3,*)(T(I),I=1,NQQQ)
      WRITE(3,*)(OUT(I),I=1,NQQQ)
      WRITE(3,*)(EXPN(I),I=1,NQQQ)
      WRITE(3,*)ALPHA
      WRITE(3,*)ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
      WRITE(3,*)NOFLT
      WRITE(3,*)(HF(I),I=1,5)
      WRITE(3,*)IPATH
      WRITE(3,*)(AB(I),I=1,N)
      WRITE(3,*)GAIN
      CLOSE(UNIT=3)
      STOP
      END
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C      ********************************************
C      ********************************************
C      ***                                      ***
C      ***                                      ***
C      ***            T 1 6 F . F T N           ***
C      ***                                      ***
C      ***                                      ***
C      ********************************************
C      ********************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE: T16F.FTN
C
C     BY: JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTIVE TITLE:  16KB/S TRANSMITTER SIMULATION FOR
C             NON-FINITE WORD LENGTH ALGORTIHM.
C
C     DESCRIPTION:  THIS PROGRAM USES THE OUTPUT PARAMETER FILE
C             OF P16F.FTN, RENAMED CMDPARA.DAT, TO SET ALL THE
C             OPTIONS OF THE TRANSMITTER.  FOR THIS REASON, THE
C             OPERATOR IS NOT PROMPTED FOR ANY INFORMATION DURING
C             A RUN.  THIS PROGRAM REPRESENTS THE NON-FINITE
C             WORD LENGTH SIMULATION OF THE TRANSMITTER FOR
C             THE TDHS-ARC ALGORITHM.  THE TRANSMISSION RATE
C             IS ASSUMED TO BE 16KB/S.
C
C*****************************************************************
C*****************************************************************
```

```
C
C
      INTEGER HD(40),IPICH(400),FNAME(16),SQ,Q,WARE6
      INTEGER*2 CLPSP(512),STAT
      INTEGER*4 STRT,BUFSZ
      DIMENSION  H(400), SQ(16),ISBUF(512),IFSP(512),IYY(16)
      REAL ALPHA
C***************
C***************
C***************
      COMMON /PRED/G,RMSMIN,ALP,ALAD,ABIAS,A(12),VHAT(12),EP,EV
     1         ,SPERB1(200),SPERB2(200),V(12),SNRB(200),SNRQB(200),B(12),NSPSAM
     2         ,ISTAT1(40),N,KQ
      COMMON /RMSB/RMS,ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG
      COMMON /CLIPP/CLPP,ALPHA,ISEC
      COMMON /ADDN/STAT(40)
      COMMON /WARE6/WARE6
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1         ,IPATH
      COMMON /FILT/HF(5),IX(9),IFSC,NOFLT
C***************
C***************
C***************
      MODN(K) = K - (K-1)/16 * 16
C*****************************************************************
C*****************************************************************
C***************
C***************INITIALIZATION BLOCK.
C***************
      OPEN(UNIT=8,TYPE='OLD',NAME='CMDPARA.DAT')    !OPEN THE OPTION FILE.
C***************
C***************OPEN THE SPEECH FILE; READ THE HEADER AND SPEECH.
C***************
      READ(8,10)FNAME
10    FORMAT(16A2)
      OPEN(UNIT=1, TYPE='OLD', READONLY, NAME=FNAME, SHARED)! OPEN SPEECH FILE
      READ(1,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
20    FORMAT(6I5,10X,40A1)
30    FORMAT(16I5)
40    CONTINUE
C***************
C***************CREATE NEW FILE FOR QUANTIZER OUTPUT. WRITE
C***************THE HEADER.
C***************
      OPEN(UNIT=2, TYPE='NEW', NAME='QUANT.DAT',CARRIAGECONTROL='LIST')
   50 FORMAT(40A1)
      NSPSAM = NSAMP/2
      WRITE(2,20)NSENT,IRATE,NSPSAM,IUPPR,ILOWR,NTERMS,HD
C***************
C***************DEFINE AND INITIALIZE VARIOUS PARAMETERS.
C***************
      K1=773
```

```
      K2=119
      DO 6Ø JJ=1,1ØØ
   6Ø         CALL RANDU(K1,K2,XX)
      NUMP = Ø
      NADD  = Ø
      ISEC=Ø          ISETS THE SIGNAL ENERGY COUNTER INITIALLY TO ZERO
      IFSC=Ø            ISETS THE FILTERED SPEECH COUNTER INITIALLY TO ZERO
      READ(8,*)NBL
      READ(8,*)CLPP
      READ(8,*)KBLK1,ITMIN,ITMAX
C***************
C***************INITIALIZATION SUBROUTINE.
C***************
      CALL INSTRT
C***************
C***************FILE MANAGEMENT FOR RUNS.
C***************
      IF(ITCS.NE.1)GO TO 1ØØ
            OPEN(UNIT=1Ø,TYPE='NEW',NAME='TCLPSP.DAT',CARRIAGECONTROL='LIST')
               WRITE(1Ø,2Ø)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  1ØØ IF(ITSE.NE.1)GO TO 2ØØ
            OPEN(UNIT=7,TYPE='NEW',NAME='TSENGY.DAT',CARRIAGECONTROL='LIST')
               WRITE(7,2Ø)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  2ØØ IF(ITPF.NE.1.AND.IPATH.NE.1.AND.IPATH.NE.3)GO TO 3ØØ
            OPEN(UNIT=4,TYPE='NEW',NAME='TPITCH.DAT',CARRIAGECONTROL='LIST')
               WRITE(4,2Ø)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  3ØØ IF(ITFF.NE.1)GO TO 4ØØ
            OPEN(UNIT=11,TYPE='NEW',NAME='TFILTER.DAT',CARRIAGECONTROL='LIST')
               WRITE(11,2Ø)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  4ØØ IF(IPATH.NE.3.AND.IPATH.NE.4)GO TO 6ØØ
            OPEN(UNIT=12,TYPE='NEW',NAME='TZHAT.DAT',CARRIAGECONTROL='LIST')
               WRITE(12,2Ø)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  6ØØ CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************BEGIN MAIN LOOP. SPEECH BUFFER CREATED AND CLIPPED
C***************SPEECH GENERATED BY SUBROUTINE.
C***************
      SENGY=Ø.
      BUFSZ=2*ITMAX
      BUFSZ=(BUFSZ/16)*16+16
      READ(1,3Ø)(ISBUF(I),I=1,BUFSZ)
      STRT=1
   22 CONTINUE
      IF(NOFLT.EQ.1)GO TO 49Ø
      DO 5Ø5 I=STRT,BUFSZ
               IFSP(I)=ISBUF(I)
  5Ø5 CONTINUE
      GO TO 485
C***************
C***************LOW PASS FILTER SUBROUTINE.
```

```
C***************
  49Ø CALL IFILT(ISBUF,STRT,BUFSZ,IFSP)
C***************
C***************SUBROUTINE TO CLIP SPEECH.
C***************
  485 CALL CLP(IFSP,STRT,BUFSZ,CLPSP,SENGY)
C*****************************************************************
C*****************************************************************
C***************
C***************PITCH DETERMINED BY AUTOCORRELATION.
C***************
   21 IBIGG=-1ØØØ
      I2SUM=-1ØØØ
      DO 12 IT=ITMIN,ITMAX+1
              ISUM=Ø
              DO 13 J=1,KBLK1
                      IF(((CLPSP(J+IT).LT.Ø).AND.(CLPSP(J).LT.Ø)).OR.
     1                        ((CLPSP(J+IT).GT.Ø).AND.(CLPSP(J).GT.Ø)))
     2                        ISUM=ISUM+1
                      IF(((CLPSP(J+IT).LT.Ø).AND.(CLPSP(J).GT.Ø)).OR.
     1                        ((CLPSP(J+IT).GT.Ø).AND.(CLPSP(J).LT.Ø)))
     2                        ISUM=ISUM-1
   13         CONTINUE
              IF(I2SUM.GT.IBIGG.AND.I2SUM.GT.ISUM)NP=IT-1
              IF(I2SUM.GT.IBIGG.AND.I2SUM.GT.ISUM)IBIGG=I2SUM
              I2SUM=ISUM
   12 CONTINUE
      NUMP=NUMP+1
C***************
C***************RANDOM PITCH GENERATOR.
C***************
      NINC=MODN(NUMP)
      IF(NP.GT.ITMIN) GO TO 29
              CALL RANDU(K1,K2,XX)
              NP=XX*(ITMAX-ITMIN)/2+ITMIN
              WRITE(5,462)NUMP,NP
   29 IPICH(NINC)=NP
  462 FORMAT(//6X,'NUMP=',I4,6X,'NP IS=',I4)
      IF(ITPF.NE.1.AND.IPATH.NE.1.AND.IPATH.NE.3)GO TO 5ØØ
      IF(NINC.EQ.16)WRITE(4,3Ø)(IPICH(J),J=1,16)
C***************
C***************WINDOW FORMED BY SUBROUTINE FOR SPEECH COMPRESSION
C***************FROM PITCH GENERATED.
C***************
  5ØØ CALL WINDOW(H,NP)
C*****************************************************************
C*****************************************************************
C***************
C***************SPEECH IS COMPRESSED AND ARC IS EMPLOYED.
C***************
      DO 14 I=1,NP
              NUM4=NP+I
```

```
                YY=FLOAT(ISBUF(NUM4))+H(I)*FLOAT(ISBUF(I)-ISBUF(NUM4))
                IARG=NADD+I
                N1=MODN(IARG)
                YYY=YY
                IF(YY.GT.0.)YYY=YY+0.5
                IF(YY.LT.0.)YYY=YY-0.5
                IF(YY.GT.2047.)YYY=2047.0
                IF(YY.LT.-2048.)YYY=-2048.0
                IYY(N1)=IFIX(YYY)
                IF((IPATH.EQ.3.OR.IPATH.EQ.4).AND.N1.EQ.16)WRITE(12,30)
     1                          (IYY(J),J=1,16)
                CALL ARC(YY,Q,YHAT)
                JQQ=Q
                STAT(JQQ)=STAT(JQQ)+1
                SQ(N1)=Q
                IF(N1.EQ.16)WRITE(2,33)SQ
   14 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************UPDATE FOR NEXT BLOCK FO SPEECH. BUFFERS ARE
C***************SHIFTED OVER 2*NP PLACES AND THEN FILLED AGAIN.
C***************IF THE END OF THE FILE IS REACHED, ZEROS ARE AP-
C***************PENDED TO FINISH PROCESSING THE FILE.
C***************
      NADD=NADD+NP
      NP2=2*NP
      INBUFF=BUFSZ-NP2
      DO 15 I=1,INBUFF
                ISBUF(I)=ISBUF(NP2+I)
                CLPSP(I)=CLPSP(NP2+I)
                IFSP(I)=IFSP(NP2+I)
   15 CONTINUE
      BUFSZ=2*ITMAX
      MORE=BUFSZ-INBUFF
      IF(MOD(MORE,16).EQ.0)GO TO 25
      MORE=(MORE/16)*16+16
   25 BUFSZ=INBUFF+MORE
      NLEFT=NSAMP-2*NADD
      IF(MORE.GT.NLEFT)GO TO 23
      READ(1,30,END=26)(ISBUF(INBUFF+I),I=1,MORE)
   26 STRT=INBUFF+1
      GO TO 22
   23 CHK=0.
      DO 17 I=1,INBUFF
                CHK=CHK+FLOAT(ABS(ISBUF(I)))
   17 CONTINUE
      IF(CHK.EQ.0.)GO TO 24
      NZEROS=MORE-NLEFT
      READ(1,30,END=27)(ISBUF(INBUFF+I),I=1,NLEFT)
   27 DO 16 I=1,NZEROS
                ISBUF(INBUFF+NLEFT+I)=0
```

```
   16 CONTINUE
      STRT=INBUFF+1
      GO TO 22
   24 NONES=16-N1
      DO 18 I=1,NONES
               SQ(N1+I)=1
   18 CONTINUE
      WRITE(2,33)SQ
      IF(ITPF.EQ.1.OR.IPATH.EQ.1.OR.IPATH.EQ.3)WRITE(4,30)
     1                    (IPICH(J),J=1,NINC)
      IF(IPATH.GE.3)WRITE(12,30)(IYY(J),J=1,N1)
C***************
C***************STATISTICS GENERATED FOR OUTPUT FILE BY SUBROUTINE.
C***************
      CALL INEND
C*****************************************************************
C*****************************************************************
C***************
C***************CLOSE FILES.
C***************
   33 FORMAT(16I2)
      CLOSE(UNIT=7)
      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      CLOSE(UNIT=8)
      CLOSE(UNIT=4)
      CLOSE(UNIT=11)
      STOP
      END
```

```
C
C
C
C
C     ************************************
C     *                                  *
C     *   W I N D O W   F U N C T I O N  *
C     *                                  *
C     ************************************
C
C
C*******************************************************************
C*******************************************************************
C
C     DESCRIPTION: SUBROUTINE PRODUCES THE VALUES "H", WHICH
C             CORRESPOND TO A TRIANGULAR WINDOW.  THIS
C             WINDOW IS GENERATED EVERYTIME A PITCH (NP)
C             IS PASSED TO IT.
C
C*******************************************************************
C*******************************************************************
C***************
C***************
C***************
      SUBROUTINE WINDOW(H,NP)
C***************
C***************
C***************
      DIMENSION H(400)
      DO 10   I = 1, NP
              H(I) = 1.0 - FLOAT(I-1)/FLOAT(NP-1)
10    CONTINUE
      RETURN
      END
```

```
C
C
C
C
C     ******************************
C     *                            *
C     *   Q U A N T I Z E R        *
C     *                            *
C     ******************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION:  SUBROUTINE MODELS A FIXED QUANTIZER. THE INPUT
C             IS THE ERROR DETERMINED BY THE PREDICTOR IN THE
C             ADAPTIVE RESIDUAL CODER AND THE OUTPUTS ARE THE
C             QUANTIZED ERROR AND THE ASSOCIATED THRESHOLD LEVEL
C             NUMBER INTO WHICH THIS ERROR MAPPED.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE QUANT(X,Y,I)
C***************
C***************
C***************
      INTEGER*2 STAT
C***************
C***************
      COMMON /QUAN/T(51),OUT(51),EXPN(51),SIZE,SMIN,NQ
      COMMON /RMSB/RMS,ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG
      COMMON /ADDN/STAT(40)
C***************
C***************
10    X1=ABS(X/SIZE)
      F=0.5
      IF(X.LT.0.)F=-.5
C*****************************************************************
C*****************************************************************
C***************
C***************DETERMINATION OF APPROPRIATE LEVEL.
C***************
      I=1
      DO 20 K=1,NQ
      TNEW=T(K)
   20 IF(X1.GE.TNEW) I=2*K+.5+F
      J=(I+2)/2
      IF(X1.GT.T(NQ+1))T(NQ+2)=T(NQ+2)+1.
C*****************************************************************
```

```
C*****************************************************************
C****************
C****************THE VALUE OF THE QUANTIZED ERROR IS DETERMINED.
C****************
      Y=2.*F*OUT(J)*SIZE
C*****************************************************************
C*****************************************************************
C****************
C****************THE STANDARD DEVIATION ESTIMATE IS UPDATED.
C****************
      SIZE=EXPN(J)*SIZE
      SIZE=AMAX1(SIZE,RMS)
      RETURN
      END
```

```
C
C
C
C
C     ********************************
C     *                              *
C     *  I N I T I A L I Z A T I O N  *
C     *                              *
C     ********************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE INITIALIZES MOST OF THE PARAMETERS
C             BY READING FROM CMDPARA.DAT, THE RENAMED OUTPUT OF
C             PFWL.FTN.  OTHER PARAMETERS ARE ALSO INITIALIZED
C             TO THEIR APPROPRIATE STARTING VALUES.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE INSTRT
C***************
C***************
C***************
      INTEGER WARE6
C***************
C***************
      COMMON /QUAN/T(51),OUT(51),EXPN(51),SIZE,SMIN,NQ
      COMMON /PRED/G,RMSMIN,ALP,ALAD,ABIAS,A(12),VHAT(12),EP,EV
     1         ,SPERB1(200),SPERB2(200),V(12),SNRB(200),SNRQB(200),B(12),NSPSAM
     2         ,ISTAT1(40),N,KQ
      COMMON /RMSB/RMS,ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG
      COMMON /INIT/JI
      COMMON /WARE6/WARE6
      COMMON /CLIPP/CLPP,ALPHA,ISEC
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1         ,IPATH
      COMMON /FILT/HF(5),IX(9),IFSC,NOFLT
      COMMON /GAIN/GAIN,EQ
      COMMON /STATS/ENGY5,ENGY6,ENGY7,ENGY8,NDIV
C***************
C***************
      WARE6=9
      READ(8,*)ALP,ALAD,ABIAS,G,N,RMSMIN,SMIN
    2 FORMAT(/6X,'ALP=',F5.2,2X,'ALAD=',F5.2,2X,'G='
     1        F5.3,2X,'N=',I2,2X,'RMSMIN=',F5.1,2X,'SMIN=',F5.2,2X,'ABIAS='
     2        F5.3/)
      READ(8,*)KQ
```

```
      NQ=KQ/2
      NQQQ=NQ+1
C*****************************************************************
C*****************************************************************
C****************
C****************QUANTIZER VALUES ARE SET.
C****************
      READ(8,*)(T(I),I=1,NQQQ)
      T(NQ+2)=0.
      READ(8,*)(OUT(I),I=1,NQQQ)
      READ(8,*)(EXPN(I),I=1,NQQQ)
      READ(8,*)ALPHA
C****************FILE MANAGEMENT FLAGS.
      READ(8,*)ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
      READ(8,*)NOFLT
      READ(8,*)(HF(I),I=1,5)
      READ(8,*)IPATH
      READ(8,*)(B(I),I=1,N)
      READ(8,*)GAIN
C*****************************************************************
C*****************************************************************
C****************
C****************OPEN STATISTICS FILE AND WRITE PARAMETERS.
C****************
      IF(ITSF.NE.1)GO TO 201
      OPEN(UNIT=9,TYPE='NEW',NAME='TSTATS.DAT')
      WRITE(WARE6,2)ALP,ALAD,G,N,RMSMIN,SMIN,ABIAS
      WRITE(WARE6,160)KQ,NQQQ
  160 FORMAT(//' THE FOLLOWING IS A LIST OF SCALING FACTORS,'/
     1         ' EXPANSION FACTORS, AND THRESHOLDS FOR A ',I2,' LEVEL'/
     2         ' QUANTIZER(',I2,' ON A SIDE)'//)
      WRITE(WARE6,180)
      DO 190 I=1,NQQQ
              WRITE(WARE6,200)I,OUT(I),T(I),EXPN(I)
  190 CONTINUE
  180 FORMAT(/3X,'LEVEL NUMBER',4X,'SCALING FACTORS',4X,'THRESHOLDS',
     1         4X,'EXPANSION FACTORS'/)
  200 FORMAT(6X,I2,17X,F7.4,9X,F7.4,10X,F7.4)
  201 SIZE=100.
      DO 118 I=1,12
      VHAT(I)=0.
      V(I)=0.
      A(I)=B(I)
  118 CONTINUE
      DO 119 I=1,9
              IX(I)=0.0
  119 CONTINUE
      DO 120 I=1,NQQQ
              T(I)=T(I)*SMIN
              OUT(I)=OUT(I)*SMIN
  120 CONTINUE
      RMS=RMSMIN
```

```
C*****************************************************************
C*****************************************************************
C****************
C****************ENERGY VALUES FOR STATISTICS ARE INITIALLY SET TO
C****************ZERO.
C****************
      EV=0.
      EP=0.
      ENGY1=0.
      ENGY2=0.
      JI=0
      ENGY3=0.
      ENGY4=0.
      ENGY5=0.
      ENGY6=0.
      ENGY7=0.
      ENGY8=0.
      NDIV=0
      SPER1=0.
      SPER2=0.
      DO 621 I=1,KQ
  621 ISTAT1(I)=0
      RETURN
      END
```

```
C
C
C
C
C     ***************************************************
C     *                                                 *
C     *  A D A P T I V E   R E S I D U A L   C O D E R  *
C     *                                                 *
C     ***************************************************
C
C
C*************************************************************
C*****************************************************************
C
C     DESCRIPTION:  SUBROUTINE MODELS THE ADAPTIVE RESIDUAL
C             CODER. THE IMPUT IS A COMPRESSED SPEECH SAMPLE
C             AND A QUANTIZATION LEVEL IS RETURNED TO BE
C             PASSED ON TO THE CHANNEL.
C
C*************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE ARC(Y,Q,YHAT)
C***************
C***************
C***************
      INTEGER Q
C***************
C***************
      COMMON /PRED/G,RMSMIN,ALP,ALAD,ABIAS,A(12),VHAT(12),EP,EV
     1        ,SPERB1(200),SPERB2(200),V(12),SNRB(200),SNRQB(200),B(12),NSPSAM
     2        ,ISTAT1(40),N,KQ
      COMMON /RMSB/RMS,ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG
      COMMON /INIT/JI
      COMMON /GAIN/GAIN,EQ
      COMMON /STATS/ENGY5,ENGY6,ENGY7,ENGY8,NDIV
C*************************************************************
C*****************************************************************
C***************
C***************PREDICTION IS COMPUTED AND AN ERROR SAMPLE IS
C***************GENERATED TO BE PASSED ON TO THE QUANTIZER.
C***************
      PRE=0.
      PRE1=0.
      DO 120 I=1,N
              PRE1=PRE1+A(I)*V(I)
  120 PRE=PRE+A(I)*VHAT(I)
      PRE=PRE*GAIN
      PRE1=PRE1*GAIN
      RMS=ALP*(RMS-RMSMIN)+(1.-ALP)*ABS(EQ)+RMSMIN
```

```
C***************
C***************ERROR SAMPLE GENERATED.
C***************
      ERROR=Y-PRE
      ERROR1=Y-PRE1
C***************
C***************QUANTIZER SUBROUTINE IS ENGAGED.
C***************
      CALL QUANT(ERROR,EQ,IOUT)
C***************
C***************UPDATE PREDICTOR VALUES.
C***************
      ISTAT1(IOUT)=ISTAT1(IOUT)+1
      Q=IOUT
      DO 125 I=1,N
      J=N+2-I
      V(J)=V(J-1)
  125 VHAT(J)=VHAT(J-1)
      VHAT(1)=PRE+EQ
      V(1)=Y
      YHAT=VHAT(1)
C*****************************************************************
C*****************************************************************
C***************
C***************ADAPTATION OF PREDICTOR COEFFICIENTS IS ACCOMPLISHED
C***************BY A FIXED PERCENTAGE COMPUTATION.
C***************
      IF(G.EQ.0.0)GO TO 134
      DO 100 I=1,N
              A(I)=A(I)-B(I)
  100 CONTINUE
      DO 130 I=1,N
      SVL=SIGN(1.,EQ)*SIGN(1.,VHAT(I+1))
      IF((EQ.EQ.0..OR.VHAT(I+1).EQ.0.).AND.A(I).GE.0.)SVL=-1.
      IF((EQ.EQ.0..OR.VHAT(I+1).EQ.0.).AND.A(I).LT.0.)SVL=+1.
      AINT=SVL*AMAX1(ABS(A(I))*G,ABIAS)
      A(I)=ALAD*(A(I)+AINT)
  130 CONTINUE
      DO 150 I=1,N
              A(I)=A(I)+B(I)
  150 CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************ENERGY STATISTICS ARE GENERATED FOR DETERMINATION
C***************OF SNR VALUES REPORTED IN STATISTICS FILE.
C***************
  134 IRM=MOD(IARG,NBL)
      ENGY1=ENGY1+Y**2
      ENGY2=ENGY2+(ERROR-EQ)**2
      ENGY3=ENGY3+ERROR**2
      ENGY4=ENGY4+ERROR1**2
```

```
      ENGY5=ENGY5+Y*YHAT
      ENGY6=ENGY6+YHAT**2
      ENGY7=ENGY7+YHAT
      ENGY8=ENGY8+Y
      NDIV=NDIV+1
      IF(IRM .NE. 0)GOTO 133
      JI=JI+1
      EV=EV+ENGY1
      EP=EP+ENGY2
      IF(ENGY3.NE.0..AND.ENGY1.NE.0.)SPERB1(JI)=10.*ALOG10(ENGY1/ENGY3)
      IF(ENGY4.NE.0..AND.ENGY1.NE.0.)SPERB2(JI)=10.*ALOG10(ENGY1/ENGY4)
      IF(ENGY2.NE.0..AND.ENGY1.NE.0.)SNRB(JI)=10.*ALOG10(ENGY1/ENGY2)
      IF(ENGY2.NE.0..AND.ENGY3.NE.0.)SNRQB(JI)=10.*ALOG10(ENGY3/ENGY2)
      SPER1=SPER1+ENGY3
      SPER2=SPER2+ENGY4
      ENGY1=0.
      ENGY2=0.
      ENGY3=0.
      ENGY4=0.
133   CONTINUE
      RETURN
      END
```

```
C
C
C
C
C     ****************************************
C     *                                      *
C     *    STATISTICS   AND   RESULTS        *
C     *                                      *
C     ****************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION:  SUBROUTINE GENERATES THE SNR VALUES FOR THE
C                   STATISTICS FILE.  FOR MORE IN DEPTH DESCRIPTION
C                   OF THE STATISTICS SEE THE DESCRIPTION IN TFWL.FTN.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE INEND
C***************
C***************
C***************
      REAL PROB(40),ALPHA
      INTEGER*2 STAT
      INTEGER WARE6
C***************
C***************
      COMMON /PRED/G,RMSMIN,ALP,ALAD,ABIAS,A(12),VHAT(12),EP,EV
     1        ,SPERB1(200),SPERB2(200),V(12),SNRB(200),SNRQB(200),B(12),NSPSAM
     2        ,ISTAT1(40),N,KQ
      COMMON /RMSB/RMS,ENGY1,ENGY2,ENGY3,ENGY4,SPER1,SPER2,NBL,IARG
      COMMON /ADDN/STAT(40)
      COMMON /WARE6/WARE6
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /QUAN/T(51),OUT(51),EXPN(51),SIZE,SMIN,NQ
      COMMON /STATS/ENGY5,ENGY6,ENGY7,ENGY8,NDIV
C***************
C***************
      IF(ITSF.NE.1)RETURN
      SNR=10.*ALOG10(EV/EP)
      SPER=10.*ALOG10(EV/SPER1)
      SPERI=10.*ALOG10(EV/SPER2)
      SNRQ=10.*ALOG10(SPER1/EP)
      ISUM=0
      DO 300 II=1,KQ
300   ISUM=ISUM+ISTAT1(II)
```

```
      ARG1=ISUM
      SUM=ALOG(ARG1)
      DO 500 I=1,KQ
      ARG=ISTAT1(I)+0.001
  500 SUM=SUM-(ARG*ALOG(ARG))/ARG1
      BITS=SUM/ALOG(2.)
      WRITE(WARE6,402)SNR,BITS,(ISTAT1(I),I=1,KQ)
  402 FORMAT(6X,'SNR INLOOP=',F7.3,3X,'H=',F4.2,3X,'OP',
     1        10I5,3(/22X,10I5)//)
      IF(WARE6.EQ.6)WRITE(5,402)SNR,BITS,(ISTAT1(I),I=1,KQ)
      WRITE(WARE6,404)
404   FORMAT(///6X,' SAMPLE NUMBER ',3X,' SNR ',7X,' SPER ',5X,
     1        ' SPERI ',4X,' SNRQ '//)
      NB=ISUM/NBL
      DO 408  I=1,NB
              IS=(I-1)*NBL+1
              IE=IS+NBL-1
              WRITE(WARE6,412)IS,IE,SNRB(I),SPERB1(I),SPERB2(I),SNRQB(I)
408   CONTINUE
412   FORMAT(6X,I5,'-',I5,6X,F7.2,4X,F7.2,4X,F7.2,4X,F7.2)
      WRITE(WARE6,416)SPER,SPERI,SNRQ
416   FORMAT(//6X,' PREDICTOR PERFORMANCE =',F8.2/6X,
     1        ' PREDICTOR IDEAL PERFORMANCE= ',F8.2/6X,' SIGNAL TO NOISE
     2        RATIO=',F8.2)
      NQUA=0
      DO 418  I=1,32
418   NQUA=NQUA+STAT(I)
      DO 420  I=1,32
              PROB(I)=FLOAT(STAT(I))/FLOAT(NQUA)
420   CONTINUE
      WRITE(WARE6,422)
422   FORMAT(//,6X,'LEVEL NUMBER',6X,'PROBABILITY',6X,' FREQUENCY',/)
424   FORMAT(9X,I2,12X,F7.4,10X,I5)
      DO 426  I=1,32
      WRITE(WARE6,424)I,PROB(I),STAT(I)
426   CONTINUE
      WRITE(WARE6,505)T(NQ+1),T(NQ+2)
  505 FORMAT(' THE NUMBER OF TIMES THE THRESHOLD VALUE OF ',F7.4,'WAS'
     1        'EXCEEDED IS ',F10.1)
      DEN1=ENGY7
      DEN2=(ENGY7**2)-FLOAT(NDIV)*ENGY6
      IF(DEN1.EQ.0.)TYPE *,' BETA DENOM = 0. CANNOT DIVIDE'
      IF(DEN2.EQ.0.)TYPE *,' K DENOM = 0. CANNOT DIVIDE'
      IF(DEN1.EQ.0..OR.DEN2.EQ.0.)GO TO 700
      TOP1=ENGY6*ENGY8*ENGY7-FLOAT(NDIV)*ENGY5*ENGY6
      BETA=(ENGY5/DEN1)-(TOP1/(DEN1*DEN2))
      TOP2=ENGY8*ENGY7-FLOAT(NDIV)*ENGY5
      SNRKAY=TOP2/DEN2
      SNR1=EV-2.*SNRKAY*ENGY5-2.*BETA*ENGY8+2.*BETA*SNRKAY*ENGY7
      SNR2=(BETA**2)*FLOAT(NDIV)+(SNRKAY**2)*ENGY6
      DENSNR=SNR1+SNR2
      IF(DENSNR.EQ.0.)TYPE *,' NOISE OF MAXIMIZED SNR = 0.'
```

```
      IF(DENSNR.EQ.0.)GO TO 700
      SNRMAX=10.*ALOG10(EV/DENSNR)
      WRITE(WARE6,750)SNRKAY,BETA,SNRMAX
750   FORMAT(' THE SNR SCALING FACTOR IS K=',F10.5/' THE SNR DC',
     1        ' OFFSET IS BETA =',F10.5/' AND THE MAXIMIZED SNR IS =',F10.5)
700   RETURN
      END
```

```
C
C
C
C
C     ************************************
C     *                                  *
C     *      C L I P P I N G             *
C     *                                  *
C     ************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE ACCEPTS BLOCK OF SPEECH SAMPLES
C              AS INPUT AND THREE VALUE CENTER CLIPS THIS SPEECH
C              BY A FIXED PERCENTAGE OF THE COMPUTED SIGNAL
C              ENERGY.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE CLP(IFSP,STRT,FNSH,CLPSP,SENGY)
C***************
C***************
C***************
      REAL ALPHA
      INTEGER*4 STRT,FNSH
      INTEGER*2 CLPSP(512)
      DIMENSION IFSP(512)
      INTEGER SENG(16),CLIPFI(16)
C***************
C***************
      COMMON /CLIPP/CLPP,ALPHA,ISEC
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
C***************
C***************
      DO 21 I=STRT,FNSH
              MSEC=MOD(ISEC,16)+1
              ISEC=ISEC+1
              SENGY=SENGY*ALPHA+(1.-ALPHA)*(ABS(FLOAT(IFSP(I))))
              IF(SENGY.LT.Ø.)SVECEL=SENGY-Ø.5
              IF(SENGY.GT.Ø.)SVECEL=SENGY+Ø.5
              IF(SENGY.GT.2Ø47.)SVECEL=2Ø47.Ø
              IF(SENGY.LT.-2Ø48.)SVECEL=-2Ø48.Ø
              SENG(MSEC)=IFIX(SVECEL)
              IF(ITSE.NE.1)GO TO 2Ø
              IF(MSEC.EQ.16)WRITE(7,1Ø)(SENG(JJ),JJ=1,16)
   2Ø         CLPLVL=CLPP*SENGY
```

```
      CLM=-CLPLVL
      XFLOT=FLOAT(IFSP(I))
      CLPSP(I)=0
      IF (XFLOT.GT.CLPLVL)CLPSP(I)=+1
      IF (XFLOT.LT.CLM)CLPSP(I)=-1
      CLIPFI(MSEC)=CLPSP(I)
      IF(ITCS.NE.1)GO TO 21
      IF(MSEC.EQ.16)WRITE(10,10)(CLIPFI(JJ),JJ=1,16)
21 CONTINUE
10 FORMAT(16I5)
   RETURN
   END
```

```
C
C
C
C
C     ************************************************
C     *                                              *
C     *   C L I P P I N G    F I L T E R             *
C     *                                              *
C     ************************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE ACCEPTS BLOCK OF SPEECH SAMPLES
C              AND FILTERS THEM WITH A 9TH ORDER LINEAR PHASE
C              FINITE IMPULSE RESPONSE FILTER.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE IFILT(ISBUF,STRT,FNSH,IFSP)
C***************
C***************
C***************
      DIMENSION ISBUF(512),IFSP(512),IFO(16)
      INTEGER*4 STRT,FNSH
C***************
C***************
      COMMON /FILT/HF(5),IX(9),IFSC,NOFLT
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1         ,IPATH
C***************
C***************
   10 FORMAT(16I5)
      DO 500 I=STRT,FNSH
              IFINC=MOD(IFSC,16)+1
              IFSC=IFSC+1
              DO 100 J=2,9
                      JIJ=11-J
                      IX(JIJ)=IX(JIJ-1)
  100         CONTINUE
              IX(1)=ISBUF(I)
              FSP=0.0
              DO 200 KIK=1,4
                      FSP=FSP+HF(KIK)*FLOAT(IX(KIK)+IX(10-KIK))
  200         CONTINUE
                      FSP=FSP+HF(5)*FLOAT(IX(5))
              IF(FSP.GT.0.0)FSP=FSP+0.5
              IF(FSP.LT.0.0)FSP=FSP-0.5
```

```
              IF(FSP.GT.2047.)FSP=2047.0
              IF(FSP.LT.-2048.)FSP=-2048.0
              IFSP(I)=IFIX(FSP)
              IFO(IFINC)=IFSP(I)
              IF(ITFF.NE.1)GOTO 500
              IF(IFINC.EQ.16)WRITE(11,10)(IFO(JJJ),JJJ=1,16)
  500 CONTINUE
      RETURN
      END
C
C
C
C
C*********************************************************************
C*********************************************************************
```

```
C
C
C
C
C      ***********************************************
C      ***********************************************
C      ***                                         ***
C      ***                                         ***
C      ***           R 1 6 F . F T N               ***
C      ***                                         ***
C      ***                                         ***
C      ***********************************************
C      ***********************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE: R16F.FTN
C
C     BY: JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTIVE TITLE: 16KB/S NON-FINITE WORD LENGTH
C                        RECEIVER SIMULATION
C
C     DESCRIPTION:  THIS PROGRAM USES THE OUTPUT PARMETER FILE
C             OF P16F.FTN, RENAMED CMDPARA.DAT, TO SET ALL THE
C             OPTIONS OF THE RECEIVER.  FOR THIS REASON, THE
C             OPERATOR IS NOT PROMPTED FOR ANY INFORMATION
C             DURING A RUN. THIS PROGRAM REPRESENTS THE NON-
C             FINITE WORD LENGTH SIMULTAION OF THE RECEIVER FOR
C             THE TDHS-ARC ALGORTIHM.  THE TRANSMISSION RATE
C             IS ASSUMED TO BE 16KB/S.
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
      INTEGER HD(40),Q(256),IZHAT(16),CLPSP(512),CLIPFI(16),SENG
      INTEGER IPICH(400),SQ(16),FNAME1(16),P1,BUFSZ,WARE5
      DIMENSION YHAT(400),SHAT(256),H(256),QCOMP(512),IFSP(16),FS(512)
C****************
C****************
      COMMON /CLIPP/CLPP,ALPHA,SENG(16),MSEC,ISEC
      COMMON /PRED/ALP,ALAD,ABIAS,A(12),EV,EP,VHAT(12),G,RMSMIN
     1        ,B(12),NSPSAM,KQ,N
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /FILT/HF(5),XXX(9),IFSC,IFINC,NOFLT
C*****************************************************************
C*****************************************************************
C****************
C****************FUNCTIONS:
C****************      MODN:DIVIDES ARGUMENT BY 16 AND RETURNS
C****************                REMAINDER.
C****************      MODM:RETURNS NEXT LARGER NUMBER WHICH IS
C****************                A MULTIPLE OF 16.
C****************
      MODN(K)=K-(K-1)/16*16
      MODM(K)=((K-1)/16)*16+16
C*****************************************************************
C*****************************************************************
C****************
C****************FORMATS USED IN MAIN PROGRAM.
C****************
   10 FORMAT(16A2)
   20 FORMAT(6I5,10X,40A1)
   30 FORMAT(40A1)
   40 FORMAT(16I2)
   50 FORMAT(16I5)
   60 FORMAT(///,40A1,//)
   70 FORMAT(////6X,' SAMPLE NUMBER ',6X,' PITCH PERIOD '//)
   80 FORMAT(6X,I5,1X,'-',1X,I5,10X,I3)
C*****************************************************************
C*****************************************************************
C****************
C****************INITIALIZE PARAMETERS.
C****************
      OPEN(UNIT=4,TYPE='OLD',NAME='CMDPARA.DAT')
      READ(4,10)FNAME1
      READ(4,*)NBL
      READ(4,*)CLPP
      READ(4,*)KBLK1,ITMIN,ITMAX
C****************
C****************INITIALIZATION SUBROUTINE.
C****************
      CALL INSTRT
C*****************************************************************
```

```
C*****************************************************************
C****************
C****************OPEN FILES AND WRITE HEADERS.
C****************
          OPEN(UNIT=1,TYPE='OLD',NAME='DECO.DAT')
             READ(1,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
             NSPSAM=NSAMP+NSAMP
      IF(ICSF.NE.1)GO TO 400
          OPEN(UNIT=3,TYPE='NEW',NAME='ZHAT.DAT',CARRIAGECONTROL='LIST')
             WRITE(3,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  400 IF(IOSF.NE.1)GO TO 410
          OPEN(UNIT=7,TYPE='NEW',NAME='SHAT.DAT',CARRIAGECONTROL='LIST')
             WRITE(7,20)NSENT,IRATE,NSPSAM,IUPPR,ILOWR,NTERMS,HD
  410 IF(IRSE.NE.1)GO TO 420
          OPEN(UNIT=9,TYPE='NEW',NAME='RSENGY.DAT',CARRIAGECONTROL='LIST')
             WRITE(9,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  420 IF(IRPF.NE.1)GO TO 430
          OPEN(UNIT=2,TYPE='NEW',NAME='RPITCH.DAT',CARRIAGECONTROL='LIST')
             WRITE(2,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  430 IF(IRCS.NE.1)GO TO 440
          OPEN(UNIT=10,TYPE='NEW',NAME='RCLPSP.DAT',CARRIAGECONTROL='LIST')
             WRITE(10,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  440 IF(IRFF.NE.1)GO TO 445
          OPEN(UNIT=11,TYPE='NEW',NAME='RFILTER.DAT',CARRIAGECONTROL='LIST')
             WRITE(11,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  445 IF(IPATH.EQ.1.OR.IPATH.EQ.2)GO TO 435
             OPEN(UNIT=12,TYPE='OLD',NAME='TZHAT.DAT')
             READ(12,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  435 IF(IPATH.EQ.2.OR.IPATH.EQ.4)GO TO 425
             OPEN(UNIT=13,TYPE='OLD',NAME='TPITCH.DAT')
             READ(13,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
  425 CONTINUE
C*****************************************************************
C*****************************************************************
C****************
C****************INITIALIZE REMAINING PARMETERS.
C****************
      ISEC=0            !SETS SIGNAL ENERGY COUTNER TO ZERO
      IFSC=0            !SETS FILTERED SPEECH COUNTER TO ZERO
      JJ=0
      NLEFT=0
      INC1=0
      INC2=0
      P1=0
      NADD=0
      SENGY=0.
      K1=773
      K2=119
      DO 450 JQ=1,100
  450         CALL RANDU(K1,K2,XX)
C*****************************************************************
C*****************************************************************
```

```
C***************
C****************SET UP MAIN LOOP INITIAL SPEECH BLOCKS AND
C****************VARIABLES.
C***************
      BUFSZ=2*ITMAX
      BUFSZ=MODN(BUFSZ)
      DO 48# I=1,ITMAX
              QCOMP(I)=#.
  48# CONTINUE
      IF(IPATH.GE.3)READ(12,5#)(Q(J),J=1,BUFSZ)
      IF(IPATH.LE.2)READ(1,4#)(Q(J),J=1,BUFSZ)
      NNEED=BUFSZ
      INC3=ITMAX
      ISTRTI=1
      IFNSHI=NNEED+ITMAX
      NSTOP=BUFSZ+ITMAX+16
C*******************************************************************
C*******************************************************************
C***************
C****************BEGIN MAIN LOOP.  INVERSE ARC ENGAGED FOR EACH
C****************SAMPLE AND RESULTING BLOCKS OF SPEECH ARE
C****************EXPANDED USING TDHS.
C***************
  495 DO 5## I=1,NNEED
              IF(IPATH.LE.2)CALL ARCR(Q(I),YY)
              IF(IPATH.GE.3)YY=Q(I)
              INC2=INC2+1
              INC1=MODN(INC2)
              QCOMP(INC3+I)=YY
              IF(YY.GT.#.#)YY=YY+#.5
              IF(YY.LT.#.#)YY=YY-#.5
              IF(YY.GT.2#47.)YY=2#47.#
              IF(YY.LT.-2#48.)YY=-2#48.#
              IZHAT(INC1)=IFIX(YY)
              IF(ICSF.NE.1)GO TO 5##
                      IF(INC1.EQ.16)WRITE(3,5#)(IZHAT(K),K=1,16)
  5## CONTINUE
      IF(IPATH.EQ.1.OR.IPATH.EQ.3)GO TO 35#
C***************
C****************COMPRESSED SPEECH SI FILTERED BY SUBROUTINE.
C***************
      IF(NOFLT.EQ.1)GO TO 49#
      DO 5#5 I=ISTRTI,IFNSHI
              FS(I)=QCOMP(I)
  5#5 CONTINUE
      GO TO 485
  49# CALL IFLT(QCOMP,ISTRTI,IFNSHI,FS)
C***************
C****************CLIP THE COMPRESSED SPEECH BY SUBROUTINE.
C***************
  485 CALL CLPPP(FS,ISTRTI,IFNSHI,CLPSP,SENGY)
C*******************************************************************
```

```
C*****************************************************************
C****************
C****************PITCH DETERMINED BY AUTOCORRELATION.
C****************
  51# IBIGG=-1###
      I2SUM=-1###
      DO 52# IT=ITMIN,ITMAX+1
              ISUM=#
              IBGN=ITMAX
              ISTP=3*KBLK1
              DO 525 J=IBGN,ISTP
                      IF(((CLPSP(J-IT).LT.#).AND.(CLPSP(J).LT.#)).OR.
     1                        ((CLPSP(J-IT).GT.#).AND.(CLPSP(J).GT.#)))
     2                        ISUM=ISUM+1
                      IF(((CLPSP(J-IT).LT.#).AND.(CLPSP(J).GT.#)).OR.
     1                        ((CLPSP(J-IT).GT.#).AND.(CLPSP(J).LT.#)))
     2                        ISUM=ISUM-1
  525         CONTINUE
              IF(I2SUM.GT.IBIGG.AND.I2SUM.GT.ISUM)NP=IT-1
              IF(I2SUM.GT.IBIGG.AND.I2SUM.GT.ISUM)IBIGG=I2SUM
              I2SUM=ISUM
  52# CONTINUE
  35# JJ=JJ+1
C****************
C****************RANDOM PITCH GENERATOR.
C****************
      JINC=MODN(JJ)
      IF(IPATH.EQ.2.OR.IPATH.EQ.4)GO TO 355
              IF(JINC.EQ.1)READ(13,5#,END=365)(IPICH(J),J=1,16)
  36#         NP=IPICH(JINC)
              GO TO 515
  365 NP=ITMIN
  355 IF(NP.GT.ITMIN)GO TO 515
              CALL RANDU(K1,K2,XX)
              NP=XX*(ITMAX-ITMIN)/2+ITMIN
              WRITE(5,288)JJ,NP
  515 IPICH(JINC)=NP
  288 FORMAT(6X,'NUMP=',I4,6X,'NP IS=',I4,12X,'RANDOM PITCH!!!!!!')
      IF(IRPF.NE.1)GO TO 53#
      IF(JINC.EQ.16)WRITE(2,5#)(IPICH(K),K=1,16)
  53# NP2=NP+NP
C****************
C****************WINDOW FORMED FOR SPEECH EXPANSION FROM
C****************PITCH GENERATED.
C****************
      DO 54# I=1,NP2
              H(I)=1.-FLOAT(I-1)/FLOAT(NP2-1)
  54# CONTINUE
C*****************************************************************
C*****************************************************************
C****************
C****************SPEECH IS EXPANDED.
```

```
C***************
  55# P1=ITMAX-NP
      NP3=NP2+NP
      DO 555 I=1,NP3
              YHAT(I)=QCOMP(P1+I)
  555 CONTINUE
      DO 56# I=1,NP2
              SHAT(I)=YHAT(I)+H(I)*(YHAT(I+NP)-YHAT(I))
              IARG=NADD+I
              N1=MODN(IARG)
              XX=SHAT(I)
              IF(XX.GT.#.#)XX=XX+#.5
              IF(XX.LT.#.#)XX=XX-#.5
              IF(XX.GT.2#47.#)XX=2#47.#
              IF(XX.LT.-2#48.#)XX=-2#48.#
              SQ(N1)=IFIX(XX)
              IF(IOSF.NE.1)GO TO 56#
              IF(N1.EQ.16)WRITE(7,5#)(SQ(J),J=1,16)
  56# CONTINUE
C*****************************************************************
C*****************************************************************
C***************
C***************SHIFT ALL VECTORS APPROPRIATELY BASED ON PITCH
C***************VALUE USED TO EXPAND SPEECH.  SET UP VECTORS
C***************AND PARAMETERS FOR NEXT RUN THROUGH MAIN LOOP.
C***************
      NADD=NADD+NP+NP
      INC3=ITMAX-NP+NNEED+NLEFT
      ISTRTI=INC3+1
      NLEFT=NNEED-NP+NLEFT
      DO 565 I=1,INC3
              QCOMP(I)=QCOMP(I+NP)
              CLPSP(I)=CLPSP(I+NP)
              FS(I)=FS(I+NP)
  565 CONTINUE
      NNEED=MODM(NP)
      NTOT=INC3+NNEED
      IF(NTOT.GT.NSTOP)NNEED=NNEED-16
      IFNSHI=INC3+NNEED
      IF(NADD.GE.NSPSAM)GO TO 6##
      IF(IPATH.GE.3)READ(12,5#,END=58#)(Q(I),I=1,NNEED)
      IF(IPATH.LE.2)READ(1,4#,END=58#)(Q(I),I=1,NNEED)
      GO TO 59#
  58# DO 585 J=I,NNEED
              Q(I)=#
  585 CONTINUE
  59# GO TO 495
C***************
C***************AT END OF FILE, FINISH WRITING VALUES TO APPROPRIATE
C***************FILES.
C***************
  6## IF(IOSF.NE.1)GO TO 61#
```

```
      WRITE(7,50)(SQ(J),J=1,N1)
   510 IF(IRSE.NE.1)GO TO 620
      WRITE(9,50)(SENG(J),J=1,INC1)
   620 IF(ICSF.NE.1)GO TO 630
      WRITE(3,50)(IZHAT(J),J=1,INC1)
   630 IF(IRPF.NE.1)GO TO 640
      WRITE(2,50)(IPICH(J),J=1,JINC)
   640 IF(IRCS.NE.1)GO TO 650
      WRITE(10,50)(CLPSP(J),J=1,MSEC)
   650 IF(IRFF.NE.1)GO TO 670
      DO 660 I=1,IFINC
              F=FS(I)
              IF(F.GT.0.)F=F+0.5
              IF(F.LT.0.)F=F-0.5
              IFSP(I)=IFIX(F)
   660 CONTINUE
      WRITE(11,50)(IFSP(J),J=1,IFINC)
C*****************************************************************
C*****************************************************************
C***************
C***************CLOSE FILES.
C***************
   670 CLOSE(UNIT=1)
      CLOSE(UNIT=3)
      CLOSE(UNIT=4)
      CLOSE(UNIT=7)
      CLOSE(UNIT=2)
      CLOSE(UNIT=11)
      CLOSE(UNIT=10)
      STOP
      END
```

```
C
C
C
C
C     ************************************************************
C     *                                                          *
C     *     I N V E R S E     Q U A N T I Z E R                  *
C     *                                                          *
C     ************************************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE MODELS A FIXED INVERSE QUANTIZER.
C              THE IMPUT IS THE QUANTIZER SEQUENCE AND THE OUTPUT
C              IS THE ESTIMATED ERROR FOR THE PREDICTED VALUE OF
C              THE SPEECH ESTIMATE.
C
C*****************************************************************
C*****************************************************************
C****************
C****************
C****************
      SUBROUTINE INVQUA(QQ,EQ)
C****************
C****************
C****************
      INTEGER QQ
C****************
C****************
      COMMON /RMS/RMS
      COMMON /QUAN/T(51),OUT(51),EXPN(51),SIZE,SMIN,NQ
C****************
C****************
      ISIGN=1
      IF(MOD(QQ,2).EQ.0)ISIGN=-1
      J=(QQ+2)/2
C*****************************************************************
C*****************************************************************
C****************
C****************DETERMINATION OF ESTIMATED ERROR.
C****************
      EQ=ISIGN*OUT(J)*SIZE
C*****************************************************************
C*****************************************************************
C****************
C****************THE STANDARD DEVIATION ESTIMATE IS UPDATED.
C****************
      SIZE=EXPN(J)*SIZE
      SIZE=AMAX1(SIZE,RMS)
      RETURN
```

```
C
C
C
C
C     ********************************
C     *                              *
C     *  I N I T I A L I Z A T I O N  *
C     *                              *
C     ********************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROTINE INITIALIZES MOST OF THE PARAMETERS
C             BY READING FROM CMDPARA.DAT, THE RENAMED OUTPUT OF
C             P16F.FTN. OTHER PARAMETERS ARE ALSO INITIALIZED TO
C             THEIR APPROPRIATE STARTING VALUES.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE INSTRT
C***************
C***************
C***************
      INTEGER WARE5
C***************
C***************
      COMMON /CLIPP/CLPP,ALPHA,SENG(16),MSEC,ISEC
      COMMON /RMS/RMS
      COMMON /QUAN/T(51),OUT(51),EXPN(51),SIZE,SMIN,NQ
      COMMON /PRED/ALP,ALAD,ABIAS,A(12),EV,EP,VHAT(12),G,RMSMIN
     1        ,B(12),NSPSAM,KQ,N
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
      COMMON /FILT/HF(5),XXX(9),IFSC,IFINC,NOFLT
      COMMON /GAIN/GAIN,EQ
C***************
C***************
      READ(4,*)ALP,ALAD,ABIAS,G,N,RMSMIN,SMIN
      READ(4,*)KQ
      NQ=KQ/2
      NQQQ=NQ+1
C*****************************************************************
C*****************************************************************
C***************
C***************QUANTIZER VALUES ARE SET.
C***************
      READ(4,*)(T(I),I=1,NQQQ)
```

```
      READ(4,*)(OUT(I),I=1,NQQQ)
      READ(4,*)(EXPN(I),I=1,NQQQ)
      READ(4,*)ALPHA
C***************FILE MANAGEMENT FLAGS.
      READ(4,*)ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
      READ(4,*)NOFLT
      READ(4,*)(HF(I),I=1,5)
      READ(4,*)IPATH
      READ(4,*)(B(I),I=1,N)
      READ(4,*)GAIN
      SIZE=100.
      DO 118 I=1,12
      VHAT(I)=0.
      A(I)=B(I)
  118 CONTINUE
      DO 119 I=1,9
              XXX(I)=0
  119 CONTINUE
      DO 120 I=1,NQQQ
              T(I)=T(I)*SMIN
              OUT(I)=OUT(I)*SMIN
  120 CONTINUE
      RMS=RMSMIN
      RETURN
      END
```

```
C
C
C
C
C     ************************************
C     *                                  *
C     *    A R C   R E C E I V E R       *
C     *                                  *
C     ************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE MODELS THE ADAPTIVE RESIDUAL
C             CODER.  THE IMPUT IS A QUANTIZATION LEVEL AND
C             A SPEECH SAMPLE ESTIMATE IS RETURNED.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE ARCR(Q,VHAT1)
C***************
C***************
C***************
      INTEGER Q
C***************
C***************
      COMMON /RMS/RMS
      COMMON /PRED/ALP,ALAD,ABIAS,A(12),EV,EP,VHAT(12),G,RMSMIN
     1         ,B(12),NSPSAM,KQ,N
      COMMON /GAIN/GAIN,EQ
C*****************************************************************
C*****************************************************************
C***************
C***************PREDICTION IS COMPUTED AND AN ERROR SAMPLE
C***************GENERATED BY THE INVERSE QUANTIZER IS ADDED TO
C***************OBTAIN A SPEECH ESTIMATE.
C***************
      PRE=0.
      DO 120 I=1,N
  120 PRE=PRE+A(I)*VHAT(I)
      PRE=PRE*GAIN
      RMS=ALP*(RMS-RMSMIN)+(1.-ALP)*ABS(EQ)+RMSMIN
C***************
C***************ERROR SSAMPLE GENERATED BY SUBROUTINE.
C***************
      CALL INVQUA(Q,EQ)
C***************
C***************UPDATE PREDICTOR VALUES.
```

```
C***************
      DO 125 I=1,N
      J=N+2-I
  125 VHAT(J)=VHAT(J-1)
      VHAT(1)=PRE+EQ
      VHAT1=VHAT(1)
C*****************************************************************
C*****************************************************************
C***************
C*****************ADAPTATION OF PREDICTOR COEFFICIENTS IS ACCOMPLISHED
C***************BY A FIXED PERCENTAGE COMPUTATION.
C***************
      IF(G.EQ.0.0)GO TO 131
      DO 100 I=1,N
              A(I)=A(I)-B(I)
  100 CONTINUE
      DO 130 I=1,N
      SVL=SIGN(1.,EQ)*SIGN(1.,VHAT(I+1))
      IF((EQ.EQ.0..OR.VHAT(I+1).EQ.0.).AND.A(I).GE.0.)SVL=-1.
      IF((EQ.EQ.0..OR.VHAT(I+1).EQ.0.).AND.A(I).LT.0.)SVL=+1.
      AINT=SVL*AMAX1(ABS(A(I))*G,ABIAS)
      A(I)=ALAD*(A(I)+AINT)
  130 CONTINUE
      DO 150 I=1,N
              A(I)=A(I)+B(I)
  150 CONTINUE
  131 RETURN
      END
```

```
C
C
C
C
C     ************************************************
C     *                                              *
C     *       C L I P P I N G                        *
C     *                                              *
C     ************************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION: SUBROUTINE ACCEPTS BLOCK OF COMPRESSED
C              SPEECH SAMPLES AS INPUT AND THREE VALUE CENTER
C              CLIPS THIS SPEECH BY A FIXED PERCENTAGE OF THE
C              COMPUTED SIGNAL ENERGY.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE CLPPP(FS,STRT,FNSH,CLPSP,SENGY)
C***************
C***************
C***************
      INTEGER*2 CLPSP(512)
      DIMENSION FS(512)
      INTEGER SENG,CLIPFI(16),STRT,FNSH
C***************
C***************
      COMMON /CLIPP/CLPP,ALPHA,SENG(16),MSEC,ISEC
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
C***************
C***************
      DO 21 I=STRT,FNSH
              MSEC=MOD(ISEC,16)+1
              ISEC=ISEC+1
              SENGY=SENGY*ALPHA+(1.-ALPHA)*(ABS(FS(I)))
              IF(SENGY.LT.0.)SVECEL=SENGY-0.5
              IF(SENGY.GT.0.)SVECEL=SENGY+0.5
              IF(SENGY.GT.2047.)SVECEL=2047.0
              IF(SENGY.LT.-2048.)SVECEL=-2048.0
              SENG(MSEC)=IFIX(SVECEL)
              IF(IRSE.NE.1)GO TO 20
              IF(MSEC.EQ.16)WRITE(9,10)(SENG(JJ),JJ=1,16)
   20         CLPLVL=CLPP*SENGY
              CLM=-CLPLVL
              XFLOT=FS(I)
```

```
      CLPSP(I)=0
      IF(XFLOT.GT.CLPLVL)CLPSP(I)=+1
      IF(XFLOT.LT.CLM)CLPSP(I)=-1
      CLIPFI(MSEC)=CLPSP(I)
      IF(IRCS.NE.1)GO TO 21
      IF(MSEC.EQ.16)WRITE(10,10)(CLIPFI(JJ),JJ=1,16)
21 CONTINUE
10 FORMAT(16I5)
   RETURN
   END
```

```
C
C
C
C
C     ***************************************************
C     *                                                 *
C     *     C L I P P I N G     F I L T E R             *
C     *                                                 *
C     ***************************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     DESCRIPTION:  SUBROUTINE ACCEPTS BLOCK OF COMPRESSED
C             SPEECH SAMPLES AND FILTERS THEM WITH A 9TH
C             ORDER LINEAR PHASE FINITE IMPULSE RESPONSE
C             FILTER.
C
C*****************************************************************
C*****************************************************************
C***************
C***************
C***************
      SUBROUTINE IFLT(QCOMP,STRT,FNSH,FS)
C***************
C***************
C***************
      DIMENSION QCOMP(512),FS(512),IFO(16)
      INTEGER STRT,FNSH
C***************
C***************
      COMMON /FILT/HF(5),XXX(9),IFSC,IFINC,NOFLT
      COMMON /IFILE/ITCS,ITPF,ITSE,ITSF,IRCS,IRPF,IRSE,ICSF,IOSF,ITFF,IRFF
     1        ,IPATH
C***************
C***************
   10 FORMAT(16I5)
      DO 500 I=STRT,FNSH
              IFINC=MOD(IFSC,16)+1
              IFSC=IFSC+1
              DO 100 J=2,9
                      JIJ=11-J
                      XXX(JIJ)=XXX(JIJ-1)
  100         CONTINUE
              XXX(1)=QCOMP(I)
              FSP=0.0
              DO 200 KIK=1,4
                      FSP=FSP+HF(KIK)*(XXX(KIK)+XXX(10-KIK))
  200         CONTINUE
                      FSP=FSP+HF(5)*XXX(5)
                      FS(I)=FSP
```

```
              IF(FSP.GT.0.0)FSP=FSP+0.5
              IF(FSP.LT.0.0)FSP=FSP-0.5
              IF(FSP.GT.2047.)FSP=2047.0
              IF(FSP.LT.-2048.)FSP=-2048.0
              IFO(IFINC)=IFIX(FSP)
              IF(IRFF.NE.1)GO TO 500
                     IF(IFINC.EQ.16)WRITE(11,10)(IFO(JJJ),JJJ=1,16)
  500 CONTINUE
      RETURN
      END
C
C
C
C
C*********************************************************************
C*********************************************************************
```

APPENDIX C

SUPPORTING PROGRAMS

```
C
C
C
C
C       ********************************************
C       ********************************************
C       ***                                      ***
C       ***                                      ***
C       ***          P R E E M P . F T N         ***
C       ***                                      ***
C       ***                                      ***
C       ********************************************
C       ********************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE: PREEMP.FTN
C
C     BY:  JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTIVE TITLE: THIS PROGRAM SIMULATES A PREEMPHASIS
C             FILTER THAT MODELS A FILTER WHOSE Z-TRANSFORM
C             IS H(Z)=
C
C*****************************************************************
C*****************************************************************
C
C
      INTEGER FNAME(16),IVEC(512),IYN(16)
      REAL LFTOVR
C****************
C****************
C****************
      MODN(K)=K-(K-1)/16*16
C*****************************************************************
C*****************************************************************
C****************
C****************FORMATS USED.
C****************
   1# FORMAT(16A2)
   2# FORMAT(6I5,1#X,4#A1)
   3# FORMAT(16I5)
C****************
C****************
C****************
      READ(1,1#)(FNAME(I),I=1,16)
      OPEN(UNIT=1,TYPE='OLD',NAME='OPT1.DAT')
      OPEN(UNIT=2,TYPE='OLD',NAME=FNAME)
      READ(2,2#)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
      OPEN(UNIT=3,TYPE='NEW',NAME='PREEMP.DAT')
```

```
      WRITE(3,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
      DO 50 I=1,16
              IYN(I)=0
   50 CONTINUE
      V1=0.
      V2=0.
      NUMLP=NSAMP/512
      LFTOVR=(FLOAT(NSAMP)/512.-FLOAT(NUMLP))*16.
      IF(LFTOVR.GT.0.)LFTOVR=LFTOVR+0.5
      IF(LFTOVR.LT.0.)LFTOVR=LFTOVR-0.5
      ILFT=IFIX(LFTOVR)
      DO 700 K=1,NUMLP
      READ(2,30,END=100)(IVEC(I),I=1,512)
  100 ISTP=512
  200 DO 600 JJ=1,ISTP
      ICNT=ICNT+1
      ICT=MODN(ICNT)
      V2=V1
      V1=FLOAT(IVEC(JJ))
      YN=V2*(-.4)+V1
      IF(YN.GT.0.)YN=YN+0.5
      IF(YN.LT.0.)YN=YN-0.5
      IYN(ICT)=IFIX(YN)
      IF(ICT.EQ.16)WRITE(3,30)(IYN(KIK),KIK=1,16)
  600 CONTINUE
      IF(ISTP.EQ.ILFT)GO TO 900
  700 CONTINUE
      READ(2,30,END=800)(IVEC(I),I=1,ILFT)
  800 ISTP=ILFT
      GO TO 200
  900 CLOSE(UNIT=3)
      CLOSE(UNIT=2)
      CLOSE(UNIT=1)
      STOP
      END
C
C
C
C
C*********************************************************************
C*********************************************************************
```

```
C
C
C
C
C      ***********************************************
C      ***********************************************
C      ***                                         ***
C      ***                                         ***
C      ***            D E E M P . F T N            ***
C      ***                                         ***
C      ***                                         ***
C      ***********************************************
C      ***********************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE: DEEMP.FTN
C
C     BY: JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTION:  THIS PROGRAM SIMULATES A DEEMPHASIS
C             FILTER THAT MODELS A FILTER WHOSE Z-TRANSFORM
C             IS H(Z)=
C
C
C*****************************************************************
C*****************************************************************
C
C
      INTEGER FNAME(16),IVEC(512),IYN(16)
      REAL LFTOVR
C***************
C***************
C***************
      MODN(K)=K-(K-1)/16*16
C*****************************************************************
C*****************************************************************
C***************
C***************FORMATS USED.
C***************
   10 FORMAT(16A2)
   20 FORMAT(6I5,10X,40A1)
   30 FORMAT(16I5)
C***************
C***************SET UP FILES.
C***************
      OPEN(UNIT=2,TYPE='OLD',NAME='SHAT.DAT')
      READ(2,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
      OPEN(UNIT=3,TYPE='NEW',NAME='DEEMP.DAT')
      WRITE(3,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
```

```
C***************
C***************IMPLEMENTATION OF FILTER
C***************
      DO 50 I=1,16
              IYN(I)=0
   50 CONTINUE
      V1=0.
      V2=0.
      NUMLP=NSAMP/512
      LFTOVR=(FLOAT(NSAMP)/512.-FLOAT(NUMLP))*16.
      IF(LFTOVR.GT.0.)LFTOVR=LFTOVR+0.5
      IF(LFTOVR.LT.0.)LFTOVR=LFTOVR-0.5
      ILFT=IFIX(LFTOVR)
      DO 700 K=1,NUMLP
      READ(2,30,END=100)(IVEC(I),I=1,512)
  100 ISTP=512
  200 DO 600 JJ=1,ISTP
      ICNT=ICNT+1
      ICT=MODN(ICNT)
      V2=V1
      V1=FLOAT(IVEC(JJ))
      YN=0.4*V2+V1
      IF(YN.GT.0.)YN=YN+0.5
      IF(YN.LT.0.)YN=YN-0.5
      IYN(ICT)=IFIX(YN)
      IF(ICT.EQ.16)WRITE(3,30)(IYN(KIK),KIK=1,16)
  600 CONTINUE
      IF(ISTP.EQ.ILFT)GO TO 900
  700 CONTINUE
      READ(2,30,END=800)(IVEC(I),I=1,ILFT)
  800 ISTP=ILFT
      GO TO 200
  900 CLOSE(UNIT=3)
      CLOSE(UNIT=2)
      STOP
      END
C
C
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C      ********************************************
C      ********************************************
C      ***                                      ***
C      ***                                      ***
C      ***          S N R B E R . F T N         ***
C      ***                                      ***
C      ***                                      ***
C      ********************************************
C      ********************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE: SNRBER.FTN
C
C     BY: JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTION:  THIS PROGRAM ESTABLISHES THE SNR OF THE
C             ARC PORTION OF THE TDHS-ARC SYSTEM WHEN A NOISY
C             CHANNEL IS USED.  THE INPUT FILES ARE TZHAT.DAT
C             AND ZHAT.DAT WHICH SHOULD BE GENERATED BY THE
C             TRANSMITTER AND THE RECEIVER ALGORITHMS,
C             RESPECTIVELY.
C
C
C*****************************************************************
C*****************************************************************
C
C
      DIMENSION ITZ(256),IZ(256)
C***************
C***************FORMATS USED.
C***************
   1Ø FORMAT(6I5,1ØX,4ØA1)
   2Ø FORMAT(16I5)
   3Ø FORMAT(' SNR=',F7.3)
C***************
C***************FILE MANAGEMENT.
C***************
      OPEN(UNIT=3,TYPE='OLD',NAME='ZHAT.DAT')
      OPEN(UNIT=4,TYPE='OLD',NAME='TZHAT.DAT')
      READ(3,1Ø)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
      READ(4,1Ø)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
C***************
C***************INITIALIZATION.
C***************
      ICNT=Ø
```

```
      TZSUM=0.
      ZSUM=0.
      K=256
C****************
C****************
C****************
  100 READ(4,20,END=500)(ITZ(J),J=1,K)
  200 READ(3,20,END=600)(IZ(J),J=1,K)
  300 DO 400 I=1,K
            ICNT=ICNT+1
            TZ=FLOAT(ITZ(I))
            Z=FLOAT(IZ(I))
            TZ2=TZ**2.
            Z2=(TZ-Z)**2.
            FLAG=10.
            IF(Z2.NE.0.)FLAG=TZ2/(Z2*2)
            IF(FLAG.LE.1.)WRITE(6,*)ICNT,TZ,Z
            TZSUM=TZSUM+TZ2
            ZSUM=ZSUM+Z2
  400 CONTINUE
      IF(K.EQ.256)GO TO 100
      SNR=10.*ALOG10(TZSUM/ZSUM)
      WRITE(5,30)SNR
      GO TO 700
  500 K=J
      GO TO 200
  600 K=J
      GO TO 300
  700 CLOSE(UNIT=3)
      CLOSE(UNIT=4)
      STOP
      END
C
C
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C       ********************************************
C       ********************************************
C       ***                                      ***
C       ***                                      ***
C       ***              T O N . F T N           ***
C       ***                                      ***
C       ***                                      ***
C       ********************************************
C       ********************************************
C
C
C*******************************************************************
C*******************************************************************
C
C     TITLE: TON.FTN
C
C     BY: JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTIVE TITLE:  TONE GENERATOR OF D/A PROGRAM
C
C     DESCRIPTION: THIS PROGRAM GENERATES A TONE SIGNAL FOR
C             THE D/A PROGRAM.  IT IS DESIGNED TO BE USED TO
C             BE ONE TEST FOR THE EFFECTIVENESS OF THE TDHS-ARC
C             SYSTEM AS A WHOLE.
C
C*******************************************************************
C*******************************************************************
C
C
      DIMENSION IX(256)
      INTEGER HD(40)
C***************
C***************
C***************
      MODN(K) = K - (K-1)/256 * 256
C***************
C***************
C***************
      TYPE *,' ENTER THE HEADER FOR THE TONE FILE:'
      ACCEPT 10,HD
10    FORMAT(40A1)
C***************
C***************INITIALIZATION.
C***************
      NSENT = 1
      IRATE = 8000
      IUPPR = 4000
      ILOWR = 2600
```

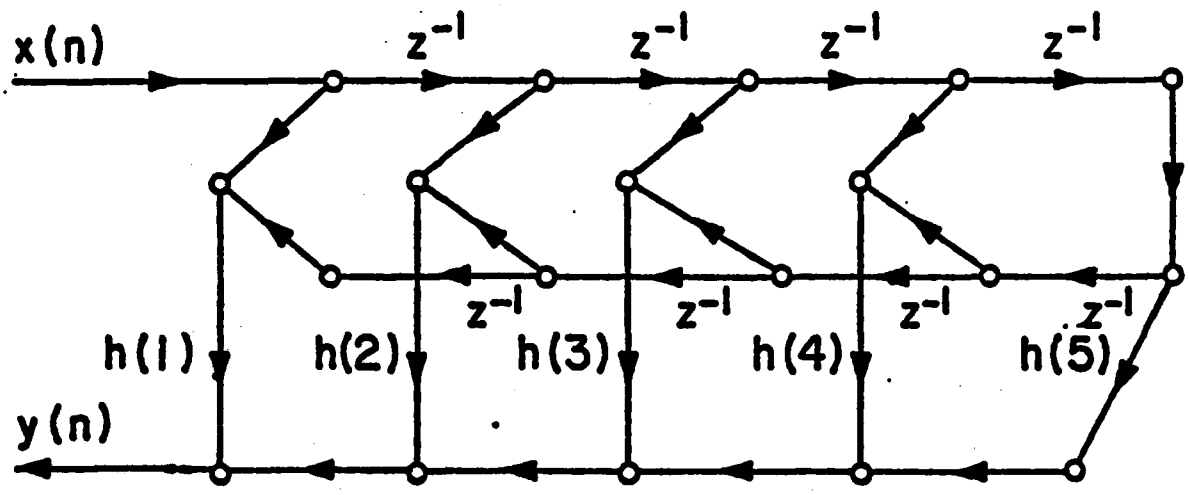
```
      NTERMS = 40
      OPEN(UNIT=1, TYPE='NEW', NAME='TONE.DAT',CARRIAGECONTROL='LIST')
C***************
C***************ENTER PARAMETERS FOR TONE.
C***************
      TYPE *,' ENTER VALUES FOR SINE FREQ,AMPLITUDE, SPEECH FREQ,'
      TYPE *,'          AND NUMBER OF SAMPLES'
      ACCEPT *,NFREQ,XMAX,NSFREQ,NSAMP
      PI = 4.*ATAN(1.)
      WRITE(1,30)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
30    FORMAT(6I5,10X,40A1)
      THETA=PI*(2. * FLOAT(NFREQ)/FLOAT(NSFREQ))
C*****************************************************************
C*****************************************************************
C***************
C***************SAMPLE GENERATION.
C***************
      DO 50    N = 0,NSAMP
               FN = FLOAT(N)
               H0= XMAX*SIN(THETA*FN)
               XX=H0
               N1 = MODN(N+1)
               IX(N1) = XX
               IF(N1.EQ.256)WRITE(1,60)IX
               IF( N.EQ.NSAMP)WRITE(1,60)(IX(J),J=1,N1)
50    CONTINUE
60    FORMAT(16I5)
      STOP
      END
C
C
C
C
C*****************************************************************
C*****************************************************************
```

```
C
C
C
C
C       ******************************************
C       ******************************************
C       ***                                    ***
C       ***                                    ***
C       ***          P I C H M A K . F T N     ***
C       ***                                    ***
C       ***                                    ***
C       ******************************************
C       ******************************************
C
C
C*****************************************************************
C*****************************************************************
C
C     TITLE: PICHMAK.FTN
C
C     BY: JAMES D. MILLS, 18 APR 1983
C
C     DESCRIPTIVE TITLE:  PITCH PULSE MAKER FOR DATAPLOT PROGRAM
C
C     DESCRIPTION:  THIS PROGRAM GENERATES SQUARE WAVE PUSES
C             WHOSE HEIGHT AND WIDTH EQUAL THE INPUT SAMPLE
C             NUMBER.  IT IS DESIGNED TO BE USED WITH A FILE
C             CONTAINING A LIST OF THE PITCH PERIODS PICKED
C             BY THE TDHS ALGORTIHM AND PLOTTED USING THE
C             DATAPLOT PROGRAM.  WHEN PLOTTED IN THIS WAY
C             IT GIVES A GRAPHICAL INDICATION OF THE EFFECTIVENESS
C             OF THE PITCH PICKING TECHNIQUE.
C
C*****************************************************************
C*****************************************************************
C
C
      INTEGER IPICH(16),ISP(16),CNT
C***************
C***************
C***************
      MODN(K)=K-(K-1)/16*16
C***************
C***************
C***************
      OPEN(UNIT=2,TYPE='NEW',NAME='PPULSE.DAT')
      OPEN(UNIT=1,TYPE='OLD',NAME='PITCH.DAT')
      READ(1,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
      WRITE(2,20)NSENT,IRATE,NSAMP,IUPPR,ILOWR,NTERMS,HD
   20 FORMAT(6I5,10X,40A1)
   10 FORMAT(16I5)
      CNT=0
```

```
      IFLAG=0
   25 READ(1,10,END=300)(IPICH(I),I=1,16)
      KK=16
C*****************************************************************
C*****************************************************************
C****************
C****************PULSE GENERATION.
C****************
   30 DO 50 I=1,KK
              MAG=IPICH(I)
C****************
C****************SET FIRST NP VALUES TO NP; WHERE NP=THE PITCH
C****************VALUE.
C****************
              DO 100 J=1,MAG
                      CNT=CNT+1
                      INC=MODN(CNT)
                      ISP(INC)=MAG
                      IF(INC.EQ.16)WRITE(2,10)(ISP(K),K=1,16)
  100         CONTINUE
C****************
C****************SET SECOND NP VALUES TO 0; WHERE NP=THE PITCH
C****************VALUE.
C****************
              DO 200 J=1,MAG
                      CNT=CNT+1
                      INC=MODN(CNT)
                      ISP(INC)=0
                      IF(INC.EQ.16)WRITE(2,10)(ISP(K),K=1,16)
  200         CONTINUE
   50 CONTINUE
      IF(IFLAG.EQ.1)GO TO 400
      GO TO 25
  300 IFLAG=1
      KK=I
      GO TO 30
  400 CONTINUE
      WRITE(2,10)(ISP(K),K=1,INC)
      CLOSE(UNIT=1)
      CLOSE(UNIT=2)
      STOP
      END
C
C
C
C
C*****************************************************************
C*****************************************************************
```

APPENDIX D

FILTER DESIGN
FINITE IMPULSE RESPONSE (FIR),
LINEAR PHASE, 9TH ORDER,
DIGITAL LOW PASS FILTER SPECIFICATIONS



IMPULSE RESPONSE

	8 kHz Wc = 1 kHz	8 kHz Wc = 500 Hz	6.4 kHz Wc = 1 kHz	6.4 kHz Wc = 500 Hz
h(1) = h(9) =	0.056	0.034	0.071	0.038
h(2) = h(8) =	0.072	0.013	0.155	0.021
h(3) = h(7) =	0.101	0.014	0.234	0.025
h(4) = h(6) =	0.122	0.015	0.319	0.028
h(5) = h(5) =	0.130	0.015	0.336	0.029
	8k//1k	8k//.5k	6.4//1k	6.4//.5k
Band 1:				
Lower Band Edge	0.0	0.0	0.0	0.0
Upper Band Edge	0.125	0.063	0.156	0.078
Band 2:				
Lower Band Edge	0.131	0.069	0.164	0.086
Upper Band Edge	0.500	0.500	0.500	0.500

APPENDIX E

FINAL OUTPUT EXAMPLES

"MOVE THE VAT OVER THE HOT FIRE"

"CATS AND DOGS EACH HATE THE OTHER"

SENTENCE: MOVE THE VAT OVER THE HOT FIRE.

ALP= 0.97 ALAD= 0.95 G=0.015 N= 4 RMSMIN= 35.0 SMIN= 0.30 ABIAS=0.01

THE FOLLOWING IS A LIST OF SCALING FACTORS,
EXPANSION FACTORS, AND THRESHOLDS FOR A 16 LEVEL
QUANTIZER(8 ON A SIDE)

LEVEL NUMBER	SCALING FACTORS	THRESHOLDS	EXPANSION FACTORS
1	0.0000	0.1639	0.5000
2	0.3277	0.5453	0.7500
3	0.7629	1.0517	1.0000
4	1.3406	1.7242	1.0000
5	2.1077	2.6169	1.0000
6	3.1262	3.8023	2.0000
7	4.4784	6.2392	3.0000
8	8.0000	0.0000	5.0000

SNR INLOOP= 26.320 H=2.91 OP 1580 1167 1207 730 753 350 316 130
37 5 32 0 8

SAMPLE NUMBER	SNR	SPER	SPERI	SNRQ
1- 64	24.77	12.11	13.64	12.66
65- 128	27.84	13.99	15.67	13.85
129- 192	28.49	14.06	15.23	14.43
193- 256	28.54	14.59	15.77	13.96
257- 320	30.74	15.04	16.08	15.69
321- 384	30.24	16.35	17.11	13.89
385- 448	28.02	14.77	17.32	13.24
449- 512	27.53	14.95	17.02	12.58
513- 576	28.71	14.73	16.54	13.98
577- 640	24.99	12.65	15.03	12.34
641- 704	25.32	11.63	12.57	13.69
705- 768	24.12	11.91	12.96	12.21
769- 832	20.96	10.10	10.45	10.86
833- 896	21.21	11.86	13.42	9.35
897- 960	26.84	9.90	9.68	16.93
961- 1024	27.98	10.58	10.96	17.40
1025- 1088	24.18	8.26	9.93	15.92
1089- 1152	23.72	9.24	10.81	14.48
1153- 1216	22.91	11.28	13.27	11.62
1217- 1280	20.30	10.17	12.24	10.12
1281- 1344	21.85	10.78	12.80	11.07
1345- 1408	20.30	9.38	10.29	10.92

1409- 1472	27.30	10.21	10.63	17.09
1473- 1536	27.91	10.85	11.83	17.05
1537- 1600	26.40	10.62	11.47	15.77
1601- 1664	28.73	11.03	11.65	17.70
1665- 1728	25.91	9.19	10.25	16.72
1729- 1792	25.82	9.65	10.25	16.17
1793- 1856	25.93	9.42	9.85	16.51
1857- 1920	26.12	10.22	10.86	15.90
1921- 1984	25.77	10.29	11.14	15.48
1985- 2048	27.24	11.02	11.36	16.22
2049- 2112	28.30	13.83	14.60	14.47
2113- 2176	17.26	7.65	11.43	9.61
2177- 2240	13.20	7.25	12.68	5.94
2241- 2304	8.76	2.63	8.36	6.13
2305- 2368	18.10	9.04	9.16	9.05
2369- 2432	28.93	13.00	14.08	15.93
2433- 2496	29.95	13.28	13.69	16.67
2497- 2560	30.53	14.76	15.97	15.77
2561- 2624	29.43	14.36	15.20	15.07
2625- 2688	29.72	14.74	15.09	14.98
2689- 2752	29.93	15.12	16.46	14.80
2753- 2816	29.16	15.00	16.41	14.16
2817- 2880	28.66	14.69	16.98	13.97
2881- 2944	27.47	14.96	16.81	12.51
2945- 3008	27.47	14.02	15.77	13.45
3009- 3072	28.96	15.23	16.43	13.73
3073- 3136	30.35	14.48	15.52	15.87
3137- 3200	29.91	13.87	15.04	16.04
3201- 3264	28.10	13.83	15.18	14.27
3265- 3328	28.97	14.35	15.44	14.62
3329- 3392	18.37	8.79	12.98	9.58
3393- 3456	17.90	11.91	18.18	5.99
3457- 3520	11.28	4.39	7.54	6.90
3521- 3584	30.52	13.37	14.09	17.15
3585- 3648	24.83	9.42	11.51	15.41
3649- 3712	26.41	10.90	12.14	15.51
3713- 3776	22.85	9.29	10.53	13.56
3777- 3840	21.20	5.92	6.75	15.28
3841- 3904	21.62	4.86	5.82	16.76
3905- 3968	18.42	2.92	3.55	15.50
3969- 4032	24.75	7.48	8.10	17.27
4033- 4096	24.01	11.01	12.78	12.99
4097- 4160	21.76	6.77	8.70	14.99
4161- 4224	24.66	7.49	8.90	17.17
4225- 4288	25.03	8.38	8.99	16.65
4289- 4352	26.75	9.49	11.07	17.26
4353- 4416	27.71	11.60	11.38	16.11
4417- 4480	17.37	9.35	10.84	8.02
4481- 4544	5.34	1.53	5.04	3.81
4545- 4608	0.18	-1.71	1.09	1.90
4609- 4672	5.87	-1.69	-0.02	7.56
4673- 4736	12.51	-0.01	1.79	12.52

4737- 4800	15.86	1.96	3.01	13.90
4801- 4864	17.25	2.67	3,30	14.58
4865- 4928	17.49	4.67	5.54	12.82
4929- 4992	17.96	4.82	5.37	13.14
4993- 5056	15.60	2.51	2.84	13.10
5057- 5120	28.46	11.83	12.40	16.62
5121- 5184	27.89	11.16	12.35	16.72
5185- 5248	27.12	10.03	10.70	17.09
5249- 5312	28.09	10.40	10.85	17.69
5313- 5376	23.33	7.45	7.57	15.88
5377- 5440	22.43	7.99	8.32	14.44
5441- 5504	21.61	7.07	8.45	14.54
5505- 5568	26.13	9.36	9.75	16.77
5569- 5632	24.08	11.01	12.83	13.07
5633- 5696	25.46	8.83	8.52	16.62
5697- 5760	20.72	5.73	7.78	14.99
5761- 5824	21.75	8.13	9.50	13.61
5825- 5888	25.61	9.70	10.92	15.91
5889- 5952	20.86	9.48	11.68	11.38
5953- 6016	26.28	9.92	9.73	16.36
6017- 6080	23.28	8.55	9.88	14.73
6081- 6144	16.69	6.03	8.24	10.66
6145- 6208	15.26	5.34	8.15	9.93
6209- 6272	14.43	5.89	8.97	8.54
6273- 6336	5.88	0.69	5.78	5.19
6337- 6400	0.00	0.00	5.08	0.00
6401- 6464	0.00	0.00	0.00	0.00

PREDICTOR PERFORMANCE = 10.47
PREDICTOR IDEAL PERFORMANCE= 11.32
SIGNAL TO NOISE RATIO= 15.85

LEVEL NUMBER	PROBABILITY	FREQUENCY
1	0.2444	1580
2	0.1805	1167
3	0.1867	1207
4	0.1129	730
5	0.1165	753
6	0.0541	350
7	0.0489	316
8	0.0201	130
9	0.0184	119
10	0.0046	30
11	0.0057	37
12	0.0008	5
13	0.0050	32
14	0.0000	0
15	0.0012	8
16	0.0000	0

SENTENCE: CATS AND DOGS EACH HATE THE OTHER.

ALP= 0.97 ALAD= 0.95 G=0.015 N= 4 RMSMIN= 35.0 SMIN= 0.30 ABIAS=0.01

THE FOLLOWING IS A LIST OF SCALING FACTORS,
EXPANSION FACTORS, AND THRESHOLDS FOR A 15 LEVEL
QUANTIZER(8 ON A SIDE)

LEVEL NUMBER	SCALING FACTORS	THRESHOLDS	EXPANSION FACTORS
1	0.0000	0.1639	0.5000
2	0.3277	0.5453	0.7500
3	0.7629	1.0517	1.0000
4	1.3406	1.7242	1.0000
5	2.1077	2.6169	1.0000
6	3.1262	3.8023	2.0000
7	4.4784	6.2392	3.0000
8	8.0000	0.0000	5.0000

SNR INLOOP= 23.152 H=3.17 OP 1973 1082 1137 865 843 618 604 302
120 42 40 1 9

SAMPLE NUMBER	SNR	SPER	SPERI	SNRQ
1- 64	17.67	0.18	-0.07	17.50
65- 128	11.62	-4.33	-5.60	15.95
129- 192	20.89	3.76	4.63	17.13
193- 256	24.73	4.66	4.38	20.07
257- 320	23.77	7.46	7.86	16.31
321- 384	25.45	7.96	8.02	17.50
385- 448	18.74	2.91	4.08	15.83
449- 512	21.11	3.59	2.80	17.52
513- 576	19.44	4.22	3.15	15.21
577- 640	19.27	2.39	3.27	16.89
641- 704	23.32	5.22	4.96	18.10
705- 768	15.78	9.90	13.78	5.88
769- 832	4.25	-0.21	4.41	4.46
833- 896	8.43	-3.53	-3.98	11.96
897- 960	3.77	-5.04	-4.88	8.81
961- 1024	7.69	-4.67	-5.65	12.36
1025- 1088	8.23	-4.26	-5.94	12.49
1089- 1152	12.51	-3.18	-4.88	15.69
1153- 1216	10.45	-5.57	-6.77	16.03
1217- 1280	9.61	-5.00	-6.09	14.62
1281- 1344	24.41	8.37	9.18	16.04
1345- 1408	24.19	7.23	7.15	16.97

1409- 1472	21.69	5.95	6.30	15.74
1473- 1536	23.77	8.68	8.88	15.09
1537- 1600	23.42	8.30	9.18	15.12
1601- 1664	25.58	10.28	10.09	15.30
1665- 1728	25.65	10.22	9.59	15.43
1729- 1792	25.58	9.70	9.85	15.88
1793- 1856	25.55	9.80	9.98	15.75
1857- 1920	24.87	9.12	8.86	15.75
1921- 1984	25.25	14.20	15.71	11.06
1985- 2048	25.23	16.08	18.72	9.14
2049- 2112	22.11	13.23	16.47	8.88
2113- 2176	17.72	1.19	-0.18	16.53
2177- 2240	24.05	8.47	7.73	15.58
2241- 2304	26.55	9.93	10.91	16.62
2305- 2368	28.87	13.78	14.40	15.09
2369- 2432	26.99	12.89	14.30	14.10
2433- 2496	29.23	13.64	14.59	15.59
2497- 2560	28.65	13.68	14.24	14.97
2561- 2624	27.35	11.82	12.71	15.53
2625- 2688	28.60	12.74	12.69	15.85
2689- 2752	26.73	10.51	12.49	16.22
2753- 2816	26.26	9.59	10.51	16.67
2817- 2880	28.33	13.18	14.01	15.15
2881- 2944	27.57	11.78	12.78	15.79
2945- 3008	29.09	13.09	13.30	16.00
3009- 3072	27.57	10.98	11.48	16.59
3073- 3136	25.89	12.28	12.75	13.61
3137- 3200	22.26	10.91	12.26	11.35
3201- 3264	26.17	13.47	14.72	12.70
3265- 3328	25.50	11.82	13.27	13.68
3329- 3392	20.52	11.23	13.61	9.30
3393- 3456	19.89	5.66	7.00	14.23
3457- 3520	17.65	1.98	1.50	15.67
3521- 3584	11.95	-4.52	-4.86	16.47
3585- 3648	10.17	-4.68	-5.09	14.85
3649- 3712	8.57	-7.89	-8.25	16.46
3713- 3776	10.65	-4.80	-5.06	15.45
3777- 3840	8.46	-7.36	-8.61	15.82
3841- 3904	10.61	-7.08	-8.01	17.69
3905- 3968	6.90	-8.91	-9.23	15.80
3969- 4032	6.60	-9.54	-8.86	16.14
4033- 4096	9.86	-5.57	-6.15	15.43
4097- 4160	8.30	-6.02	-6.82	14.32
4161- 4224	3.57	-5.34	-2.89	8.91
4225- 4288	-0.31	-0.31	-0.59	0.00
4289- 4352	0.00	0.00	-1.91	0.00
4353- 4416	0.00	0.00	-2.03	0.00
4417- 4480	0.00	0.00	-3.81	0.00
4481- 4544	0.00	0.00	-3.32	0.00
4545- 4608	0.00	0.00	-6.06	0.00
4609- 4672	9.59	6.04	9.20	3.54
4673- 4736	17.52	3.35	3.33	14.17

4737- 4800	18.90	2.11	2.01	16.79
4801- 4864	18.50	1.88	0.82	16.62
4865- 4928	16.72	-0.21	0.26	16.93
4929- 4992	17.68	0.04	0.20	17.64
4993- 5056	13.27	-3.60	-3.15	16.86
5057- 5120	16.76	-0.14	1.24	16.89
5121- 5184	13.11	5.36	7.01	7.76
5185- 5248	0.00	0.00	-8.01	0.00
5249- 5312	9.45	-7.51	-7.10	16.96
5313- 5376	8.75	-9.29	-9.83	18.04
5377- 5440	7.08	-10.27	-10.57	17.36
5441- 5504	8.26	-8.55	-9.92	16.81
5505- 5568	9.90	-5.69	-6.34	15.59
5569- 5632	15.18	-1.80	-1.68	16.97
5633- 5696	26.46	6.41	7.52	20.05
5697- 5760	19.01	2.09	2.85	16.92
5761- 5824	19.50	1.58	0.91	17.92
5825- 5888	15.01	0.01	0.01	15.01
5889- 5952	15.90	-0.86	-0.68	16.76
5953- 6016	15.18	-1.32	-2.26	16.49
6017- 6080	20.87	7.73	6.92	13.14
6081- 6144	5.49	0.39	3.47	5.10
6145- 6208	0.00	0.00	1.39	0.00
6209- 6272	0.00	0.00	-4.81	0.00
6273- 6336	0.00	0.00	-5.79	0.00
6337- 6400	17.40	2.81	1.24	14.59
6401- 6464	17.27	4.31	4.05	12.97
6465- 6528	15.98	-0.06	0.26	16.04
6529- 6592	14.78	-0.60	-2.06	15.38
6593- 6656	21.70	5.04	3.92	16.65
6657- 6720	16.21	0.30	1.35	15.92
6721- 6784	14.87	0.27	0.72	14.59
6785- 6848	14.55	-3.11	-2.44	17.66
6849- 6912	24.98	8.10	7.99	16.88
6913- 6976	24.33	8.63	10.02	15.70
6977- 7040	27.63	12.15	11.95	15.48
7041- 7104	30.29	12.95	13.76	17.35
7105- 7168	27.67	11.17	10.68	16.50
7169- 7232	28.08	9.92	10.08	18.16
7233- 7296	26.55	11.57	11.69	14.98
7297- 7360	26.88	15.23	16.02	11.65
7361- 7424	24.05	11.42	11.29	12.63
7425- 7488	21.89	6.45	7.46	15.44
7489- 7552	23.00	8.36	8.06	14.64
7553- 7616	24.40	6.21	6.68	18.19
7617- 7680	23.04	8.81	8.93	14.23
7681- 7744	22.38	6.47	6.57	15.91
7745- 7808	21.08	11.14	12.93	9.94
7809- 7872	14.81	6.48	8.39	8.33
7873- 7936	16.15	3.50	4.86	12.65
7937- 8000	3.83	-1.69	7.43	5.52
8001- 8064	0.00	0.00	0.00	0.00

PREDICTOR PERFORMANCE = 6.66
PREDICTOR IDEAL PERFORMANCE= 6.81
SIGNAL TO NOISE RATIO= 16.49

LEVEL NUMBER	PROBABILITY	FREQUENCY
1	0.2438	1973
2	0.1337	1082
3	0.1405	1137
4	0.1069	865
5	0.1042	843
6	0.0764	618
7	0.0746	604
8	0.0373	302
9	0.0410	332
10	0.0153	124
11	0.0148	120
12	0.0052	42
13	0.0049	40
14	0.0001	1
15	0.0011	9
16	0.0000	0
17	0.0000	0
18	0.0000	0
19	0.0000	0
20	0.0000	0
21	0.0000	0
22	0.0000	0
23	0.0000	0
24	0.0000	0
25	0.0000	0
26	0.0000	0
27	0.0000	0
28	0.0000	0
29	0.0000	0
30	0.0000	0
31	0.0000	0
32	0.0000	0

REFERENCES

1. Arora, A.S., Melsa, J.L., Mills, J.D., "Real Time Implementation of a Speech Coding Algorithm Using Time-Domain Harmonic Scaling and Adaptive Residual Coder", Proc. 1983 IEEE Int. Conf. Acous., Speech, Signal Processing, April 1983.

2. Cohn, D.L., Melsa, J.L., "The Relationship Between an Adaptive Quantizer and a Variance Estimator", IEEE Trans. Inform. Theory, Vol. IT-21, Nov. 1975.

3. Cohn, D.L., Melsa, J.L., The Residual Encoder and Improved ADPCM System for Speech Digitization", IEEE Trans. Commun., Vol. COM-23, September 1975.

4. Cox, R.V. Crochiere, R.E., Malah, D., "Performance of Transform and Subband Coding Systems Combined with Harmonic Scaling of Speech", IEEE Trans. on Acous. Speech and Signal Processing, Vol. ASSP-29, No. 2, April 1981.

5. Dubnowski, J.J., Rabiner, L.R., Schafer, R.W., "Real-Time Digital Hardware Pitch Detector", IEEE Trans. on Acous. Speech and Signal Processing, Vol. ASSP-24, No. 1, February 1976.

6. Luenberger, D.G., Introduction to Linear and Nonlinear Programming, Wesley Publishing Company, Massachusetts, 1965.

7. Defense Communication Agency, "Statement of Work: Contract Number DCA R610-82-029", December 1981.

8. Malah, D., "Combined Time-Domain Harmonic Compression and DVSD for 7.2 kbit/s Transmission of Speech Signals", Proc. 1980 IEEE Int. Conf. Acous. Speech, Signal Processing, April 1980.

9. Malah, D., "Time-Domain Algorithms for Harmonic Bandwidth Reduction and Time Scaling of Speech Signals", IEEE Trans. on ASSP, Vol. ASSP-27, No. 2, April 1979.

10. McDonald, R.A., "Signal-to-Noise and Idle Channel Performance of Differential Pulse Code Modulation Systems - Particular Applications to Voice Signals", Bell Syst. Tech. J., April 1966.

11. Melsa, James L., "Unsolicited Proposal to the Defense Communications Agency for support of a work entitled Real-Time Implementation of a Speech Digitization Algorithm Combining Time-Domain Harmonic Scaling and Adaptive Residual Coding", August 1981.

12. Melsa, J.L., Cohn, D.L., Kresse, J., Arora, A., Pande, A.,"A Waveform Encoder for Speech Transmission Over 9.6 kb/s Noisy Channels", NTC Conference Record, Vol. 3, No. 50.3, December 1980.

13. Melsa, J.L., Pande, A.K., "Mediumband Speech Encoding Using Time Domain Harmonic Scaling and Adaptive Residual Coding", Proc. 1981 Int. Conf. ASSP, Atlanta, GA, April 1981.

14. Melsa, J.L., Pande, A.K. "A Study of a New Approach to Digital Speech Digitization Combining Time-Domain Harmonic Scaling and Adaptive Residual Coding", Final Report to the Defense Communication Agency, August 1981.

15. Rabiner, L.R., Sambur, M.R. Schmidt, C.E., "Applications of a Nonlinear Smoothing Algorithm to Speech Processing", IEEE Trans. on Acoustics, Speech and Signal Processing, Vol. ASSP-23, No. 6, December 1975.

16. Schafer, R.W., Oppenheim, A.V., Digital Signal Processing, Prentice-Hall, New Jersey, 1975.

END