AD-A100 791 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/G 9/2
DESIGN AND DEVELOPMENT OF A GRAPHICS SYSTEM FOR DISPLAY OF SOUN--ETC(U)
DEC 80 P A DUNDAS
UNCLASSIFIED AFIT/GE/EE/80D-17 NL

AFIT/GE/EE/80D-17

DESIGN AND DEVELOPMENT OF A GRAPHICS SYSTEM FOR DISPLAY OF SOUND SPECTROGRAMS

THESIS

AFIT/GE/EE/80D-17 Paul A. Dundas
1st Lt. USAF

Approved for public release; distribution unlimited.

AFIT/GE/EE/80D-17

DESIGN AND DEVELOPMENT OF A GRAPHICS SYSTEM FOR DISPLAY OF SOUND SPECTROGRAMS

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by

Paul A. Dundas, B.S.E.E.
1st Lt. USAF
Graduate Electrical Engineering

December 1980

A

Approved for public release; distribution unlimited.

Preface

The purpose of this project was to design and implement a graphics system for the display of sound spectrograms as a research aid for the digital signal processing facility located at the Air Force Institute of Technology (AFIT). A Data General ECLIPSE S/250 and NOVA 2/10 computer system serves as the foundation for this facility. Peripheral equipment includes a Cromenco S-100 bus system which is interconnected to the NOVA, and through which the spectrograms are ultimately displayed on a television monitor.

I would like to thank Captain Larry Kizer, my thesis advisor, for initially motivating this line of research and for his help and encouragement throughout the project. I would also like to thank Professor Matt Kabrisky and Major Alan Ross for their help and support.

Finally, I would like to thank my fellow students involved in research in the digital signal processing facility, Captains Dan Fredal and George Beasley, and 2nd Lieutenants Mark Felkey and Bob Taylor, for their cooperation and assistance.

Paul A. Dundas

Contents

List of Figures

List of Tables

Table Page

Abstract

A graphics system for display of sound spectrograms was designed and implemented on a Data General NOVA computer which is interconnected with a S-100 bus system. The system allows the user to interactively select a section of the spectrogram for further analysis through the use of a set of cursors.

The computer systems used in the development of this project are described. The theory involved in the generation of digital sound spectrograms is detailed along with the display methodology used. The user is provided the choice of four spectral windows and the relative merits of each window are described and compared. The six major classifications of spectrogram patterns, called sound-pattern groups, are described. Spectrograms that show the characteristic pattern for each sound-pattern group are presented along with spectrograms that show the differences between narrow and wide-band spectral analysis. The computer programs developed on both the NOVA and the S-100 system are described.

DESIGN AND DEVELOPMENT OF A GRAPHICS SYSTEM
FOR DISPLAY OF SOUND SPECTROGRAMS

I. Introduction

Background

Under sponsorship of the United States Air Force Electronic Systems Division and the United States Aerospace Medical Research Laboratory, the Air Force Institute of Technology (AFIT) Electrical Engineering (EE) department is undertaking research in digital signal processing, specifically digital speech processing and digital image processing. A Data General Corporation (DGC) NOVA 2/10 computer, a DGC ECLIPSE S/250 computer and associated peripheral equipment are being integrated to form the digital signal processing facility.

Problem Statement

A valuable aid in the analysis of speech is the sound spectrogram. The sound spectrogram represents a three-dimensional (time vs. frequency vs. intensity) display of the frequency spectrum of a speech waveform (Ref 9:57). The objective of this thesis is to develop a system for the display of sound spectrograms. The system should be intrinsically flexible, so that it may meet the needs of unforseen future research. One requirement of the system is the capability to generate cursors on the spectrogram display so as to enable the user to interactively isolate, or pick-out, a section of the spectrogram for further analysis.

Scope

An examination of this problem begins with a look at sound spectrograms in general, and digitally generated spectrograms in particular. The information that a spectrogram provides will not be discussed in detail. However, the six main clasifications of speech patterns (called sound-pattern groups) will be described to provide the reader with a basic understanding of the spectrograms shown in Chapter V. The hardware and software used in the development of this project will be briefly described with emphasis on how the spctrograms are to be displayed. The remainder of this report describes the design, development and validation of the sound spectrogram display system.

Sequence of Presentation

The introduction to this project is followed by a description of the equipment, both hardware and software, used in the development of the system. The theory involved in the generation and display of spectrograms is described next, followed by a brief look at the six sound-pattern groups. The next section provides a look at the development of and a description of the software, and is followed by a section regarding the validation of the display system. The project is then summarized, pointing out several possibilities for follow-on work.

II. Detailed Analysis

The purpose of this analysis is to describe the computer systems involved in generating the spectrograms. A detailed discussion of the theory of digital spectrogram displays follows in Chapter III. These spectrograms are produced on the NOVA 2/10 computer. This machine is interconnected with an S-100 bus system which produces the display on a dedicated television monitor. A description of the S-100 bus system follows the description of the NOVA computer.

NOVA

The DGC NOVA is a sixteen bit machine installed within the digital signal processing facility of the AFIT EE department. The NOVA shares a ten megabyte disk with an ECLIPSE S/250 computer located in the same facility. An Inter-Processor Buffer arbitrates simultaneous disk access. Each computer operates under a Real-Time Disk Operating System (RDOS), which is partially resident in core memory as well as on the disk itself (Ref 5:15).

Programs developed on the NOVA were written in DG's Fortran IV which is an implementation of the American National Standards Institute (ANSI) Fortran Standard X3.9-1966 plus extensions (Ref 4:iii). In addition, the NOVA has a fortran callable Discrete Fourier Transform (DFT) routine. The DG DFT allows for the calculation of the

Fourier coeffecients for either the forward or inverse transform of a sequence of integer powers of two up to a maximum of 1024 points (Ref 3:i).

S-100

The NOVA is interconnected with a Cromenco Z-2 S-100 bus system, which includes a Z-80 Central Processor Unit (CPU) card and a 21-card-slot shielded motherboard. The major hardware conponets of the system are listed in Table I. The programs devleloped on this system were written in Fortran IV and run under the Cromenco Disk Operating System (CDOS). Cromenco's Fortran IV is a partial implementation of the ANSI Fortran Standard X3.9-1966 (Ref 2:7).

Table I

S-100 Hardware Components

COMPANY	HARDWARE
Cromenco	CPU (model ZPU-K) floppy disk controller (model 4FDC)
Seattle Computer Products	CPU support card (model SCP300C) 8/16 RAM (model SCP-107)
Tecmar	16 channel 12 bit A/D convertor board 4 channel 12 bit D/A convertor board video digitizer board set
Siemens	floppy disk drive (model FDD100-5B)

The S-100 system is used to digitize recorded speech and the digitized speech can then be transferred to a disk file, via the NOVA, where it can be analyzed on either the NOVA or the ECLIPSE. Also, the system can receive disk files from the NOVA and reconstruct an analog speech waveform from the digitized speech for replay over a speaker system. Of more importance to this project, however, is the system's ability to display the contents of memory (each byte representing two picture elements, or pixels) on a television monitor.

The system can not only display the contents of memory but also digitize video signals for storage and/or display through the use of the Tecmar video digitizer board set. This board set consists of an A/D board, Direct Memory Access (DMA) board and D/A board. These boards are combined to provide digitization of video signals from standard (EIA or NTSC) television cameras and storage of the digital image in computer memory using the S-100 bus, to provide display of the contents of memory on a television monitor, or both. When activated by the computer, this set of boards takes over control of the S-100 bus in a DMA mode. Picture information is moved directly from the A/D board to memory, or from memory to the D/A board (Ref 10:1).

The spectrograms are displayed on the television monitor by using just the DMA and D/A boards. Together, these boards are called the monitor interface boards, a subset of the video digitizer board set. The spectrogram is displayed in sixteen gray levels (two pixels per byte) from S-100 memory via DMA (processor activity is suspended for the duration of a field of video information). The pixel pair is unpacked and each pixel, in turn, is sent to the D/A converter to provide sixteen gray levels. The converter also generates the required sync for the television monitor (Ref 10:2).

III. Theory

This chapter begins with a look at sound spectrograms and continues with a discussion of the different factors affecting the choice of the window function used in generating the spectrograms. The display method chosen is detailed next. Finally, for use in Chapter V, the six sound-pattern (or phonetic) groups are described.

Sound Spectrograms

The sound spectrogram, which represents a three-dimensional (time vs. frequency vs. intensity) display of the frequency spectrum, is an important aid in the analysis of speech. Many speech sounds can be considered to be produced by exciting a resonant cavity, the vocal tract, with either a quasi-periodic or a noiselike excitation. For these applications the speech waveform is characterized by the frequencies of the vocal tract resonances and, for the quasi-periodic excitation, the fundamental frequency of the excitation, both of which are readily apparent on a spectrogram. The spectrum resulting from a quasi-periodic excitation (for voiced sounds such as vowels) is a line spectrum with harmonics spaced in frequency by the reciprocal of the pitch period (typically about 125 Hz for a male speaker). The envelope of the line spectrum will contain peaks corresponding to the resonant frequencies of the vocal cavity. Noiselike excitation (for unvoiced, fricative sounds such as /sh/ as in should), on the other hand produces, as the name implies, a noiselike output and has no line spectrum (Ref 8:57).

One method for obtaining the sound spectrogram is the classic spectrograph analyzer. This machine processes the speech through a bank, or series, of equal bandwidth filters. The usual method consists of heterodyning (moving the signal up or down in frequency) past a single fixed filter. A wide-band filter would provide better time resolution at the expense of spectral resolution; that is, it would tend not to resolve individual pitch harmonics but it would be better able to track the rapid changes of the resonances of the vocal cavity that take place in continuous speech. If a narrow-band filter is used, the frequency resolution improves and the harmonic structure becomes more evident, but the spectral analysis loses its ability to follow the rapid changes of the vocal cavity resonances. Therefore, it is common in spectral analysis to utilize both narrow-band analysis, corresponding to good frequency resolution, and wide-band analysis, corresponding to good time resolution. For narrow-band analysis the filter bandwidths are typically 45 Hz; for wide-band analysis they are 300 Hz. The spectrograph analyzer records the spectrogram on Teledeltos paper with frequency on the vertical axis, time on the horizontal axis and intensity as the darkness of the print (Ref 8:58).

Sound spectrograms can also be generated digitally on a computer (see Oppenheim, Mermelstein, and Strong and Palmer) (Ref 7, 8 and 11). This is accomplished through the use of the Discrete Fourier Transform (DFT). The DFT provides a method for calculating the frequency spectrum for many functions of time. Any time-domain waveform that can be described as a sequence of discrete values can be transformed into the frequency domain by the computer. The rediscovery of the algorithm that exploits the various symmetries inherent in the definition of the Fourier transform was made by James W. Cooley and John W. Tukey. This algorithm made the computation of the DFT practical in that it reduced the number of computations required by a factor of $(\log_2 N)/N$ where N is the length of the DFT (see Eq. (1)) (Ref 1:48). This algorihm, and its successors, are what are known today as fast Fourier transforms (FFTs).

The DFT is defined as:

$$F(k) = \sum_{n=0}^{N-1} f(nT)e^{-j(2\pi/N)nk} \qquad (1)$$

where f(nT) corresponds to equally spaced samples of an analog time function f(t). However, the computation in Eq. (1) provides only one spectral section, that is, the transform of one section of the analog time function starting at t = 0 and ending at t = (N-1)T. To obtain a spectral analysis, we would like to preform this computation at successive instants of time. Also, since a computation of the DFT as given by Eq. (1) is necessarily restricted to a computation of a finite length of data, there is implicit in (1) a time window imposed on f(t); that is, f(t) is multiplied by a rectangular window with a width equal to NT. To determine a running spectrum, and provide flexibility in terms of the time window used (and the resulting filter shape), the expression in (1) can be modified as:

$$F_r(k) = \sum_{n=0}^{N-1} w(nT)f(nT+rMT)e^{-j(2\pi/N)nk} \qquad (2)$$

Equation (2) intoduces two changes. The first is to include a window, w(nT), to allow the filter shape to be changed from that of the filter shape imposed by the rectangular window. The second modification incorporated in Eq. (2) corresponds to implementing a spectral analysis of successive sections of the waveform. In other words, the set of numbers $F_r(k)$ represents a computation of the DFT of a section of the analog time function starting at t = rMT and ending at t = rMT+(N-1)T. Successive sections are spaced in time by MT (Ref 8:59).

Windows

The problem of spectral leakage is one reason the ability to change the window is desired. In general, the spectrum of a window function consists of a main lobe representing the middle of the spectrum and various side lobes located on either side of the main lobe. Spectral leakage occurs when a given spectral component, say at $f=f_1$, contributes output (or is observed) at another frequency, say at $f=f_2$, according to the sidelobe level (or gain) of the window centered at f_1 and measured at f_2. In other words, the spectral component at $f=f_1$ is not only detected at f_1 but also is detected at f_2. This leakage causes a bias in the amplitude and the position of the harmonic estimate. This bias is most troublesome when trying to detect small signals in the presence of nearby large signals. To reduce the effects of this bias, the window should exhibit low-amplitude sidelobes far from the central main lobe, and the transition to the low amplitude sidelobes should be as rapid as possible (Ref 6:57).

The other consideration involved in choosing a window is that the main lobe should be as narrow as possible. This is to enable the DFT to pick-out, or resolve, frequency components that are close together in frequency. If, for example, two equal amplitude frequency components, say at $f=f_1$ and $f=f_2$, are less than the 6.0-dB bandwidth of the window apart, the DFT will not resolve the two individual frequencies but will exhibit a single spectral line at $f=(f_1+f_2)/2$. Therefore, the window function should have a narrow 6.0-dB bandwidth (Ref 6:58).

This means the window selection must be measured against two criteria: (a) The main lobe should be as narrow as possible. (b) The maximum side lobe level should be as small as possible. It turns out that both of these criteria can not be simultaneously optimized, so that a compromise between these two factors must be made in the choice of a window function. As mentioned above the 6.0-dB bandwidth of the window is a measure of how narrow the main lobe is. Measures of how well a window suppresses leakage are the peak sidelobe level (relative to the main lobe) and the asymptotic rate of falloff of these sidelobes (Ref 6:57).

Table II lists these figures of merit, along with the equivalent noise bandwidth, for the four windows chosen for use in this project. The first three windows, rectangular, Hanning and Hamming, were chosen for use based on previous research (Ref 7, 8, and 11). The 4-sample Kaiser-Bessel window was chosen based on its performance and on the fact that its coefficients are easily generated (when compared to other high performance windows). The window functions are defined in Table III using normalized coordinates with T=1.0 (Ref 6:82).

Table II

Windows and Figures of Merit (Ref 6:55)

WINDOW	HIGHEST SIDE-LOBE LEVEL (dB)	SIDE-LOBE FALL-OFF (dB/OCT)	EQUIV. NOISE BW (BINS)	6.0-dB BW (BINS)
RECTANGULAR	-13	-6	1.00	1.21
HANNING	-32	-18	1.50	2.00
HAMMING	-43	-6	1.36	1.81
4-SAMPLE KAISER-BESSEL	-69	-6	1.80	2.44

Table III
Window Equations

WINDOW	EQUATIONS
RECTANGULAR	$w(n)=\begin{cases} 1.0, & n=0,1,...N-1 \\ 0, & \text{elsewhere} \end{cases}$
HANNING	$w(n)=\begin{cases} 0.5(1.0-\cos(2\pi n/(N-1))), & n=0,1,...N-1 \\ 0, & \text{elsewhere} \end{cases}$
HAMMING	$w(n)=\begin{cases} 0.54-0.46\cos(2\pi n/(N-1)), & n=0,1,...N-1 \\ 0, & \text{elsewhere} \end{cases}$
4-SAMPLE KAISER-BESSEL	$w(n)=\begin{cases} 0.40243-0.49804\cos(2\pi n/(N-1)) \\ +0.09831\cos(4\pi n/(N-1)) \\ -0.00122\cos(6\pi n/(N-1), & n=0,1,...N-1 \\ 0, & \text{elsewhere} \end{cases}$

As previously discussed the filter bandwidths used for spectral analysis are typically 45 and 300 Hz. In the digital case the frequency resolution is given as:

$$\Delta f = B(f_s/N) \qquad (3)$$

where f_s is the sampling frequency, N is the number of time samples to be transformed, and B is the coefficient reflecting the bandwidth increase (over a rectangular window with the same peak power gain that would accumulate the same noise power) due to the particular window selected. The minimum resolution of the DFT is (f_s/N), which we denote as a DFT bin. The coefficient B is usually selected to be the equivalent noise bandwith in bins as listed in Table II (Ref 6:56).

The use of a window other than the rectangular window does present one problem. The increase in performance in terms of suppressing spectral leakage is gained by tapering the windows to zero at the boundaries. If the window and the DFT are applied to nonoverlapping sections of the digitized speech, a significant part of the speech is ignored due to the window's exhibiting small values near the boundaries. To overcome this loss of data, the user has the option to apply the window and the DFT to overlapped sections of digitized speech and to pick the amount of overlap. Typically, 50 or 75 percent overlap is used so that each data sample is covered equally. Overlapping the windows does increase the number of computations that have to be made to cover a particular section of speech, but the reward of suppressing spectral leakage while still covering all the data samples equally warrants the extra effort (Ref 6:56).

The capability to choose these various parameters, the sequence length N, the window, and the amount of overlap, adds flexibility to the system. Such flexibility is ideally suited to computer implementation (Ref 8:57). The same sample of speech can be analyzed in several different ways with a minimum amount of effort. This added dimension of flexibility is one of the main advantages of digitally generated sound spectrograms over analog methods. The next section details how the spectrograms are displayed.

Display Method

Thus far successive sections of digitized speech have been multiplied by a window and transformed into the frequency domain through the use of the FFT (see FIG 1). The output of the FFT must now be prepared for display. The log magnitude of the output of the FFT is taken in order to compress its dynamic range. Each section of transformed speech is normalized so that the maximum value of each section is 0 dB. The results are linearly interpolated (if necessary) to provide the desired number of frequency components for the display (see FIG 2).

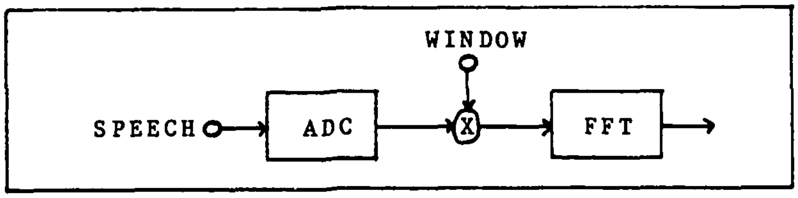

Figure 1. Partial Spectrogram Computation Block Diagram

The display system used is based on the display system used by Strong and Palmer (Ref 11). The motivation behind this method is the desire to display as much detail as possible with the available CRT brightness levels. This is accomplished through the use of simple level manipulation and the addition of spectral shaping (Ref 11:900).

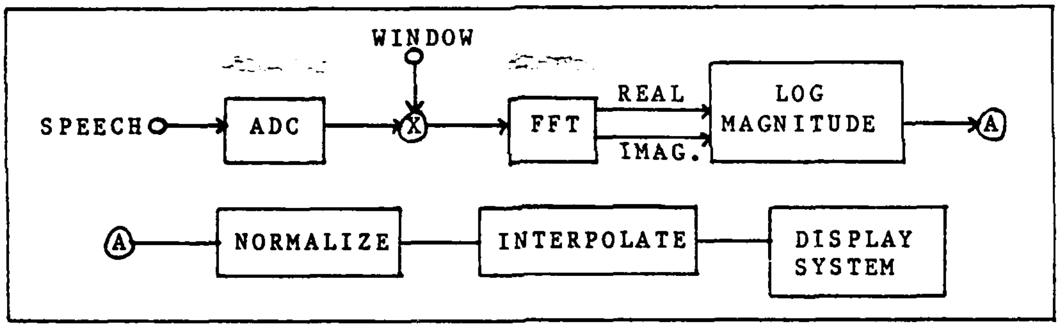

Figure 2. Spectrogram Computation Block Diagram

The first specification provided to the user of the spectrogram display system is the sound-intensity-level threshold (in decibels). This is done in order to make the best use of the available dynamic range of the display. The threshold defines the level relative to which the dynamic range begins and permits the elimination of background noise without sacrificing part of the dynamic range of the display (Ref 11:901).

The next specification provided to the user is the step size (in decibels). The user assigns the number of decibels per brightness level of the display. Strong and Palmer (Ref 11:900) provide this example: if 5 dB per brightness level is the specification, then levels up to 4 dB above threshold will be assigned brightness zero, levels from 5 dB to 9 dB above threshold will be assigned brightness one, and so on. This feature allows a greater or lesser amount of the dynamic range of the speech signal to be included in the available dynamic range of the display medium much as is done with contour plots in modern analog spectrographs (Ref 11:900).

The user is also provided with the option of inhibiting some of the lower brightness levels of the display. As an example, if the lower four brightness levels were inhibited, spectral points that would normally be assigned brightness levels one, two or three would be assigned brightness level zero. This capability provides higher contrast. This makes certain features stand out more strongly at the expense of giving up some of the subtle details (Ref 11:901).

A spectral shaping method is also provided to enhance high-frequency detail. The user inputs the amount of emphasis desired (in units of decibels per octave) and the starting frequency (in Hz). The spectral shaping takes place at the time of display and elimanates the need to pre-emphasize, or filter, the input speech to boost the high frequency components. (Ref 11:901).

The display system generates a display array with integers representing sixteen levels of intensity which is transferred into S-100 memory. The array can then be displayed on a standard television monitor through the use of the monitor interface boards. As mentioned previously, this board set consists of a DMA board and a D/A board and it can display pictures in sixteen gray levels from computer memory via DMA. Video data is brought from the memory after proper synchronization occurs. The pixel pair is unpacked and each pixel, in turn, is sent to the digital to analog converter to provide sixteen gray levels. The converter generates the required sync signals for the monitor (Ref 10:2).

This completes the discussion of the spectrogram display technique. The next section describes the six sound-pattern groups for use in understanding the spectrograms presented in Chapter V.

Sound-Pattern Groups

Since the sound spectrogram can be considered to be a phonetic translation of speech sounds, the nature of the spectrogram patterns is closely related to the manner in which the speech sounds are produced. Consequently, when sounds are classified either according to their spectrogam patterns or to the manner of their production, they fall naturally into the six groups that are used generally in phonetic classifications (Ref 9:33). A description of these clasifications follows the description of the four types of modulation in speech.

The term modulation is used to describe how speech is produced because it expresses concisely what happens to the air as it flows from the lungs. As used here, the term modulate means to alter, vary or regulate the flow of air. The first type of modulation is start-stop modulation, which may be produced by either the vocal cords or the tongue, lips, and the other articulators, and is evidenced as pauses in the flow of speech. Vocal cord modulation is produced by the vibrations of the vocal cords which periodically interrupt the flow of air. This gives the characteristics of voicing to speech. Frictional modulation is caused by forcing the air through a small opening which produces a turbulent air flow. The final type of modulation is cavity modulation which occurs in the coupled cavities of the throat, mouth and nose. This modulation can supress some of the overtones produced by both vocal cord and frictional modulation giving it the property of selective transmission (Ref 9:30-31).

Voiceless Stop Sounds. These sounds are produced primarily by a combination of stop and frictional modulation. The sounds are made by stopping the breath flow at some point in the articulatory tract, building up breath pressure, and then rapidly releasing this pressure. The p sound in up is an example (Ref 9:33).

Voiced Stop Sounds. These sounds are produced primarily by a combination of stop, vocal cord, and frictional modulation. The sounds are made by stopping the voiced breath stream at some point in the articulatory tract, building up voiced breath pressure, and suddenly releasing this pressure. The b sound in be is an example (Ref 9:34).

Voiceless Fricative Sounds. These sounds are produced primarily by frictional modulation. The sounds are made by forcing a continuous breath stream through a small or restricted opening. The restricted opening may occur at one of several places in the articulatory tract or at the vocal cords. The f sound in five is an example (Ref 9:34).

Voiced Fricative Sounds. These sounds are produced primarily by a combination of vocal cord, frictional, and cavity modulation. The sounds are made by forcing a continuous and voiced breath stream through a restricted opening which may occur at one of several places in the articulatory tract or at the vocal cords. The v sound in five is an example (Ref 9:35).

Vowel and Vowel-like Sounds. The vowel and vowel-like sounds are produced primarily by a combination of vocal cord and cavity modulation. They are produced by the passage of the voiced breath through different combinations of the coupled vocal cavities. These cavities are modified and combined, or coupled, by the action of the articulators. The vowel i as in eve is an example of these sounds (Ref 9:36).

Combinations of Sounds. Sounds within a group, or sounds from different groups often are combined in words. Such combined sounds as t (church), and dz (judge) are used together so frequently that they may be considered as separate sound units. When two vowels are combined as in aU (out) or aI (I), the combination is called a dipthong. The dipthong aI (as in I, eye, or five) is an example of a combination of two vowel sounds (Ref 9:36).

This completes the brief look at the six sound-pattern groups. The spectrogram patterns characteristic of each group will be detailed in Chapter V. The next chapter deals with the development and description of the computer programs used in the spectrogram display system.

IV. Program Development/Description

The development of the programs to generate the spectrograms started on the NOVA. As the intial program grew in size it became obvious that the program would have to be segmented. Eventually four main programs were developed: TRANSFORM, PREPARE, SHAPE and EXAMINE. These programs can be executed sequentially or separately. Interacting with these programs are the programs DISPLAY and CURSOR on the S-100 system. What follows is a description of the above programs and their development.

TRANSFORM

This program inputs digitized speech from the disk file DSPEECH. The speech is then multiplied by the chosen window. The DFT of the windowed speech is taken, and the results are stored in the disk file WINDOW. Before any of this can take place, however, several choices have to be made by the user.

The program was first developed with the following choices to be made by the user: the window, N (the number of points in the DFT), the percent overlap (of the windowed sections) and the number of input samples to be transformed. The choice of the window and N determines the filter bandwidth of the transform. Therefore the user is provided a table after the choice of the window which lists the filter bandwidth for each choice of N.

An option to pick the display size was added to allow for more flexibility. The user can pick both the vertical and horizontal resolution of the desired display. The vertical resolution choice affects the amount of interpolation required and the horizontal choice affects how much digitized speech is transformed. The display size is limited by the amount of memory available on the S-100 system. Therefore, after picking the desired vertical resolution, the user is given a list of the allowable horizontal resolutions before being asked to make a choice.

During development the capabilbity to pick the starting block of speech to be transformed was added. The digitized speech is organized on disk in blocks of 256 integers. This specification allows the user to skip the initial blocks of speech which may not contain any information of interest. The user can also use this feature to zero in on a specific section of speech.

There are more choices for the user to make before program execution begins. The choices involve noise threshold, linear interpolation and the type of display desired. However, these choices do not affect the operation of TRANSFORM. These choices will be discussed later during the program in which they come into play. Information required by later programs for operation, including the above choices, is stored in the disk file CONSTANTS.

When TRANSFORM comes to the end of its execution, all disk files are closed. PREPARE is called via the Chain command. PREPARE overwrites TRANSFORM in core memory and execution of PREPARE begins.

PREPARE

This program reads in the variables it requires for proper operation from the disk file CONSTANTS. Transformed speech is read in from the disk file WINDOW one DFT section at a time. The section is energy normalized if the resulting number of frequency points does not equal the desired vertical resolution (an N point DFT yields N/2 unique frequency points), interpolation is reqiured. The results of these operations are stored in the disk file INTRP. First, a look at the normalization routine.

The normalization routine begins by obtaining the absolute value of every complex point in the array representing the DFT section. The maximum value of this array is determined, and all points in the array are divided by this value. The resulting values in the array are converted to decibels with a maximum value of zero dB. The routine was changed when it was realized that background noise received equal emphasis with speech. To suppress the background noise the capability to enter a noise threshold was added. If the maximum value of the array was not above the threshold, the array was not divided by the maximum value of the array. Instead, the array was divided by one hundred times the noise threshold. This puts background noise at a forty decibel disadvantage compared to speech above the threshold. Considering the lowest threshold value for the spectrograms, both narrow-band and wide-band, presented in Chapter V is -20 dB, the noise which has a maximum value of -40 dB will not be displayed.

To finish preparation for the final display, each DFT section must yield the correct number of frequency points as required by the desired vertical resolution. If the number of frequency points (N/2) equals the desired resolution, no manipulation is required. When the number of frequency points is greater than the resolution, the points are combined together in a linear fashion. For example, if the number of points is 4 times greater than the desired resolution, points 1, 2, 3 and 4 are averaged together to produce resolution point 1. Likewise, frequency points 5, 6, 7 and 8 yield resolution point 2, and so on. The linear interpolation choice mentioned earlier comes into play when the number of frequency points is less than the resolution. If linear interpolation is desired, the frequency points will be linearly expanded to fill the desired resolution. The other option is to repeat the frequency points as necessary. For example, if the desired resolution is twice the number of frequency points, frequency point one will become resolution points one and two. Frequency point two will become resolution points three and four, and so on. In all the above cases, once the resolution points are determined the results are stored in the disk file INTRP.

Once all the sections have been stored in INTRP, all disk files are closed. SHAPE is called via the Chain command. SHAPE overwrites PREPARE in core memory and execution of SHAPE begins.

<u>SHAPE</u>

This program reads in the variables it requires for proper operation from the disk file CONSTANTS. The transformed and interpolated speech is read in from the disk file INTRP. The user inputs the desired threshold, step size, number of skipped levels and spectral emphasis (if desired). These values are used to set a grey level (0-15) for each spectral point. These grey levels, or pixels, are packed four to an integer. The integers that represent the display are stored in the disk file DISPLAY. When DISPLAY is filled, the file is transferred to S-100 memory for display. If the user wants to modify the display, he can enter new inputs and a new DISPLAY file is generated.

The process begins with the setting of the grey level for each spectral point. Simple level manipulation is used. A grey level of zero is assigned to spectral points with values below threshold. Values above threshold yield grey levels of up to fifteen. This produces a display with high intensity as white and the background black.

The user has previously selected (during the initial choices presented by the program TRANSFORM) either a white on black display or a black on white display. For the white on black display the display values are passed as is, via the subroutine PASCURS, to S-100 memory. To achieve a black on white display, the subroutine PASSDAT is used. PASSDAT differs from PASCURS only in that the integers are complemented prior to transfer.

The sequence of integers passed to the S-100 represent one vertical spectrogram slice after another. The first integer contains the first four display levels starting from the bottom left corner and then going up. The next integers fill a vertical line up to the top of the screen. The remaining integers fill up the rest of the screen with one vertical line, from bottom to top, after another. However, the monitor interface boards use the S-100 memory to fill the screen from left to right, starting with the upper left corner. To get around this problem, the yoke of a conventional television monitor was rotated ninety degrees. This made the upper left corner of the display appear in the lower left corner. Also, the vertical resolution switches on the monitor interface DMA board became the horizontal resolution switches, and vice versa. Thus a display with frequency as the vertical axis, time as the horizontal axis and intensity as the gray scale was presented on the television monitor.

Once an adaquate display is attained, the user can end program execution or can call the program EXAMINE via the Chain command. If EXAMINE is called, SHAPE is overwritten and execution of EXAMINE begins.

EXAMINE

This program reads in the variables it requires for proper operation from the disk file CONSTANTS. The user enters the desired cursor positions and the information is transferred to the S-100 system for display. The user can then either accept the cursor positions or enter new ones. Once the cursor positions are accepted, the user can choose from three different routines or terminate the program. The first routine expands the section of the spectrogram following the leftmost cursor. This routine provides a new display with a horizontal resolution of sixty four lines in the disk file BLOWUP. The other two routines store the digitized speech from the display between the cursors in the disk file DSPOUT. One routine obtains the digitized speech from the original speech file DSPEECH. The other routine reconstructs the digitized speech from the DFT sections stored in WINDOW. Each DFT section is inverse transformed, normalized, and if the original DFT sections were overlapped, the overlapped sections are averaged together. This routine resides in the program RECON and is called from the program EXAMINE if desired.

This ends the discussion of the four main programs on the NOVA. Two other programs deserve note, however. SENDISP can be used to transfer an already developed DISPLAY file to S-100 memory for a black on white display. Likewise, SENDWOB produces a white on black display. The programs developed on the S-100 system will be discussed next.

DISPLAY

This program accepts the packed integers from the NOVA and stores them in S-100 memory. When all the integers have been stored, the monitor interface DMA board is turned on to display the contents of memory on the television monitor. The program pauses while the monitor is on. A carriage return turns off the DMA board and thus the display, and generates another pause. At this point the program can be terminated by entering a "T." On the other hand, a carriage return allows the program to accept another set of packed integers. Once an adaquate display is obtained, DISPLAY can be terminated and the program CURSOR started.

CURSOR

This program receives the desired cursor position from the NOVA. The contents of the memory representing the four vertical lines of the display that are to be overwritten by the cursor are stored in memory that is not being displayed. The memory is then replaced by two black vertical lines followed by two white lines. This process is repeated for the second cursor. The display is then restored to the monitor. The Pause function is used again to control whether the program is terminated or for the program to return to accept new cursor positions.

One program not mentioned in the introduction to this chapter is ONE. ONE is essentially the same program as CURSOR. The only difference is that the cursors displayed are one black line followed by one white line. The user can use either program depending on his preference for cursor size.

V. Validation

The validation of this project will consist of demonstrating the six sound-pattern groups. The expected pattern will be described and the resulting spectrogram shown with the cursors (indicated by a) isolating the pattern in question. The spectrograms were reproduced by a Tektronix 4632 Video Hard Copy unit. In addition, this section will show: the effect of different windows on the same word, the effect of pitch on a vowel pattern, and the effect of narrow-band and wide-band spectrograms. The stop sounds will be described first.

The unvoiced stop sound is characterized by a blank space, or gap, representing the stoppage of the breath flow. This gap is followed by narrow irregular vertical striations, called a spike fill. This is the result of the frictional modulation produced by the sudden release of the breath pressure. Figure 3 shows the word <u>k</u>ey. The voiced stop sound has the same pattern except for a narrow bar on the baseline prior to the spike fill. This bar is called the voicing bar (Ref 9:33-34). Figure 4 shows the word <u>d</u>ay.

Both spectrograms were generated using the 4-sample Kaiser-Bessel window with N being 64. This resulted in wide-band spectrograms with a filter bandwidth of 225 Hz. The DFT sections were overlapped by 50 per cent. The threshold for both displays is -10 dB with a step size of 2 dB. These spectrograms have a spectral emphasis of 6 dB/octave starting at 500 Hz. The resolution of the display is 64 vertical pixels by 256 horizontal pixels. Unless otherwise mentioned, these specifications hold true for all the spectrograms to follow.

(NOTE: The line that appears to the right of each cursor and is visible even on the black edges of the print is not part of the spectrogram. These lines are caused by the Tektronix copier being overdriven by the white line of the cursor).

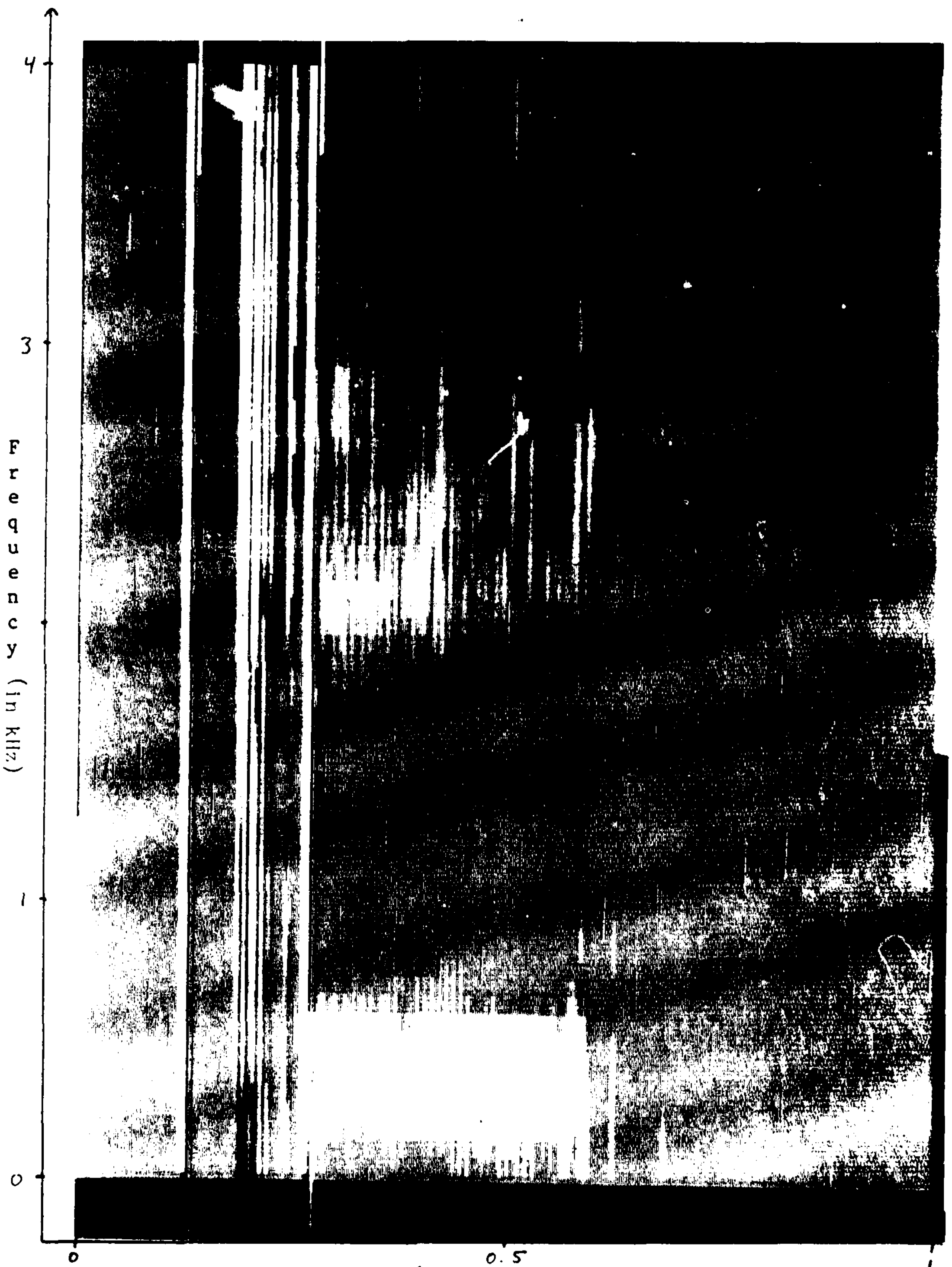

Figure 3. Unvoiced Stop Sound (key)

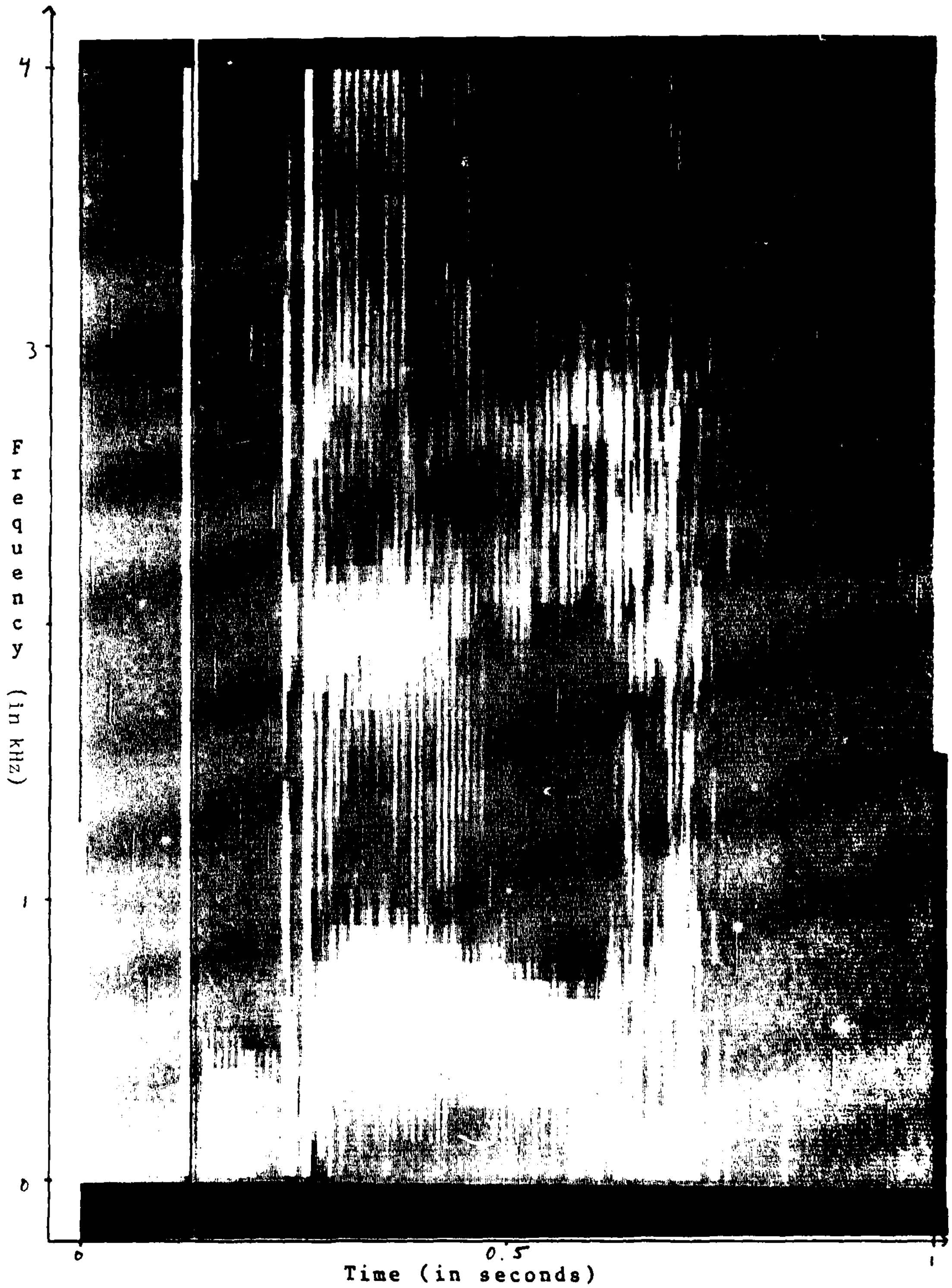

Figure 4. Voiced Stop Sound (day)

Voiceless fricative sounds produce irregular vertical striations called fills, which are generally wider than the spike fills for the stop sounds. Figure 5 shows the word <u>th</u>in. Voiced fricative sounds produce a similar pattern with the addition of a voicing bar at the baseline of the display (Ref 9:34-35). Figure 6 shows the word <u>z</u>oo.

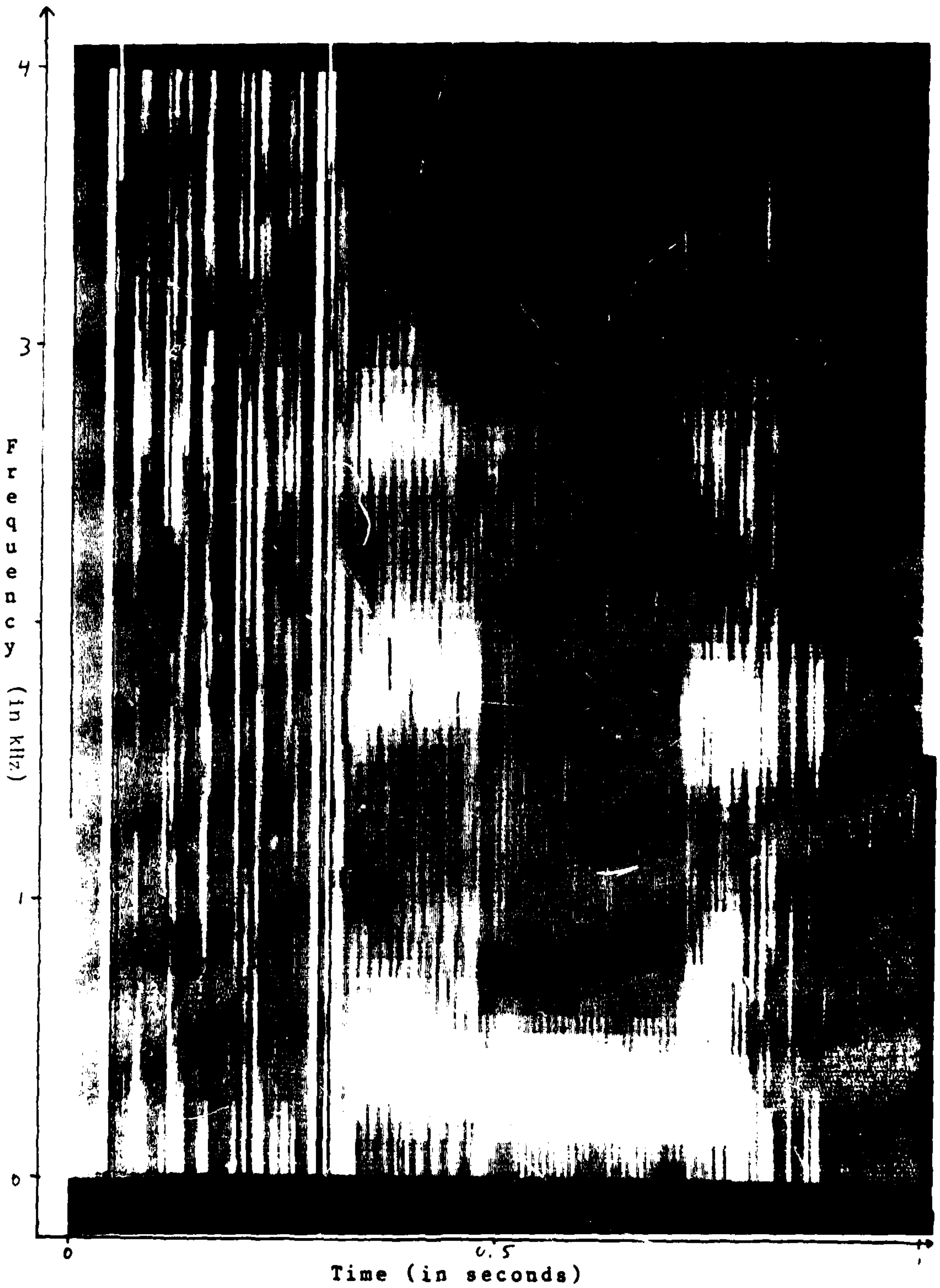

Figure 5. Unvoiced Fricative Sound (thin)

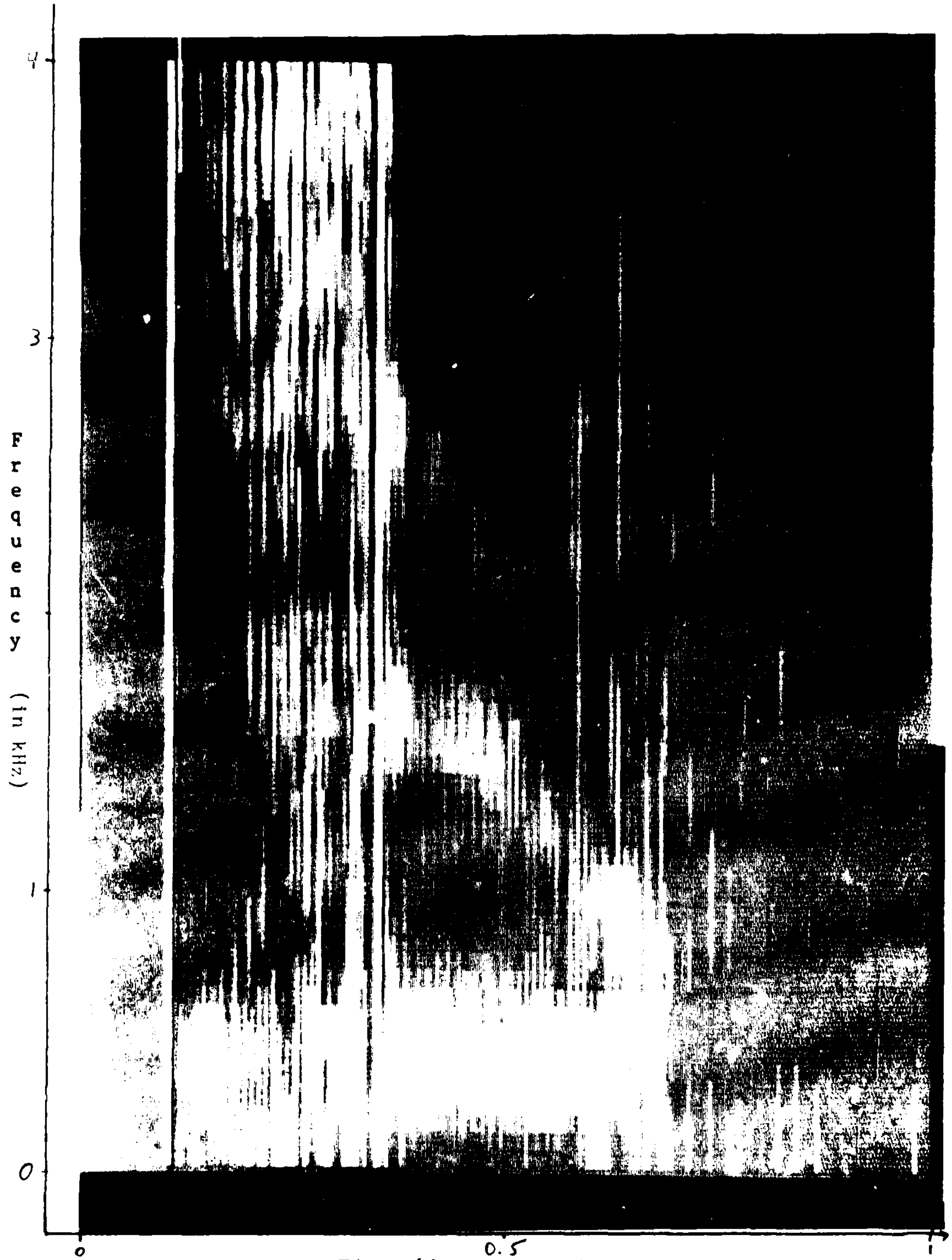

Figure 6. Voiced Fricative Sound (zoo)

The vowel and vowel-like sounds are characterized by horizontal bars called resonance bars (Ref 9:36). Figure 7 shows the word _e_ve. The only specifications that differ from the preceding spectrograms are the threshold (now -14 dB) and the step size (now 3 dB).

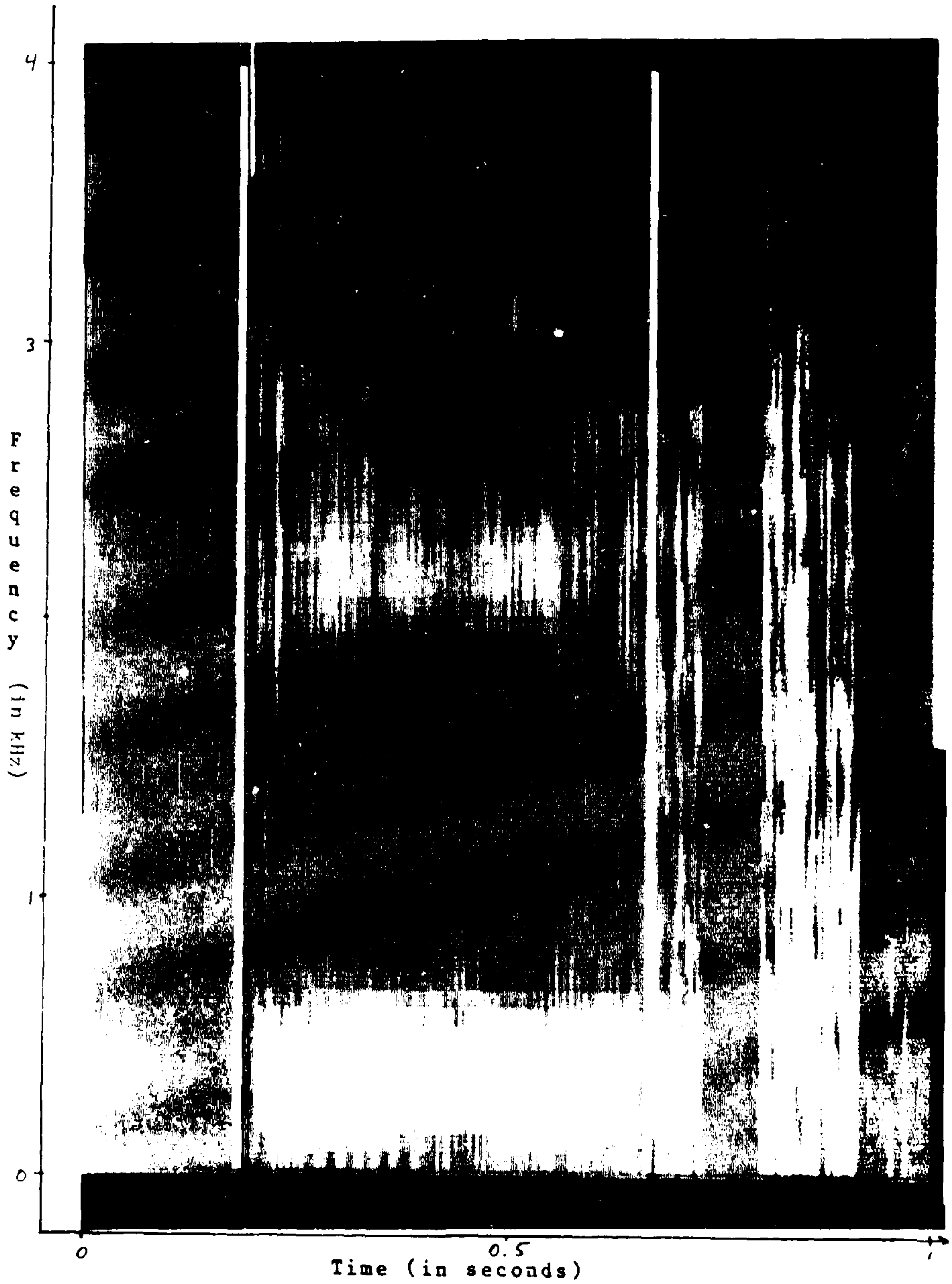

Figure 7. Vowel Sound (eve)

Combinations of sounds produce, as expected, a combination of the other five sound-pattern groups. One such combination is the t as in church. Together, the t sound is characterized by a gap followed by a spike (representing the t sound), followed by a unvoiced fill (representing the sound) (Ref 9:36-37). The word church was transformed using all four windows with a N of 64. The filter bandwidths for each window are listed below for comparison with the typical analog wide-band spectrogram filter bandwidth of 300 Hz.

Figure 8 shows the 4-sample Kaiser-Bessel window with a filter bandwidth of 225 Hz. Figure 9 shows the Hamming window with a filter bandwidth of 170 Hz. Figure 10 shows the Hanning window with a filter bandwidth of of 188 Hz. Figure 11 (note the time scale change) shows the rectangular window with a filter bandwidth of 125 Hz. The DFT sections were overlapped by 50 per cent in all cases except for the rectangular window, where no overlapping was done. The threshold for these displays was changed to -8 dB.

There appears to be little difference between the spectrograms produced by using the Kaiser-Bessel, Hamming or Hanning window. The spectrogram produced by the rectangular window does show the basic pattern but it lacks the detail of the other spectrograms.

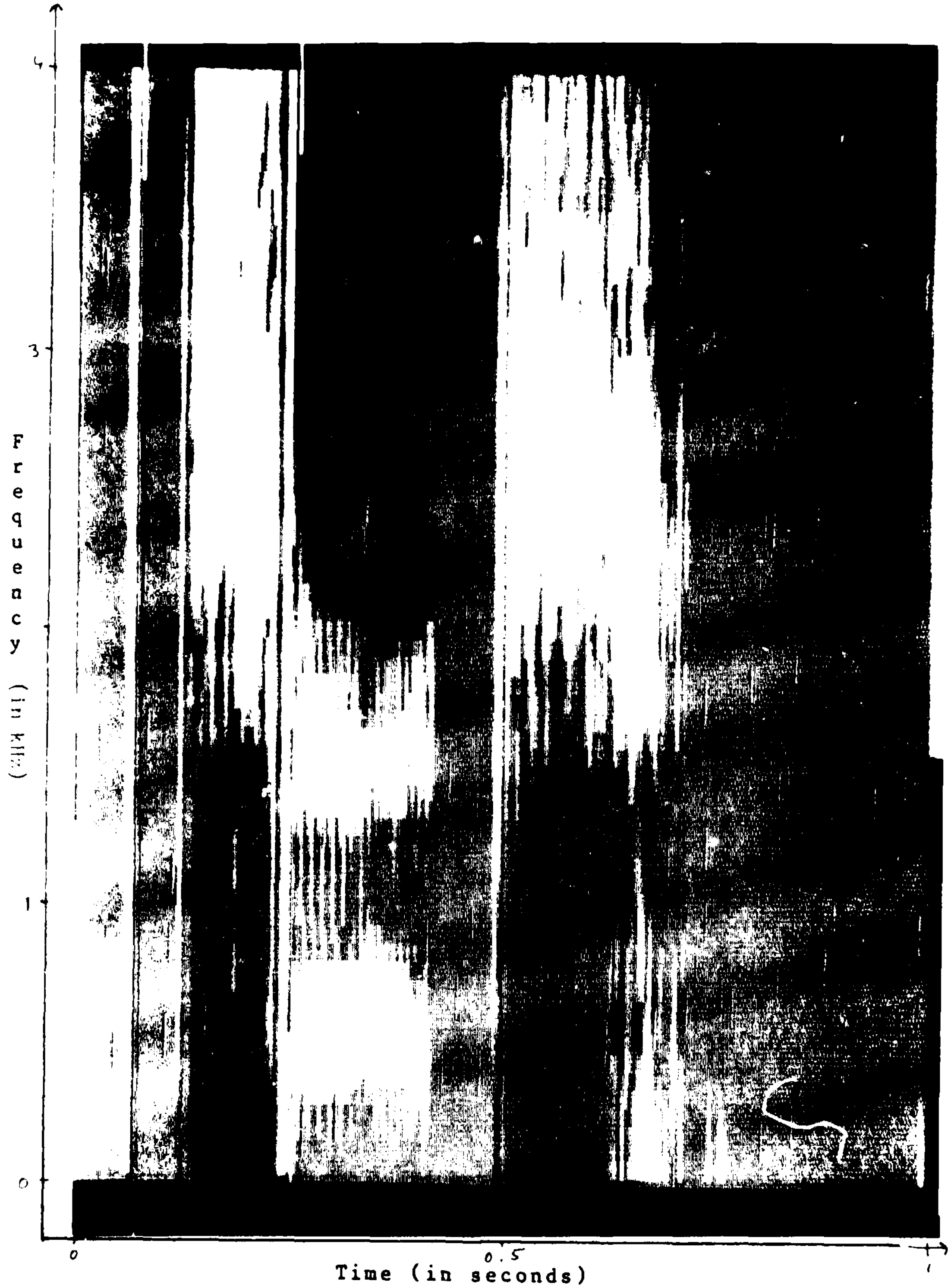

Figure 8. Kaiser-Bessel Window (<u>ch</u>ur<u>ch</u>)

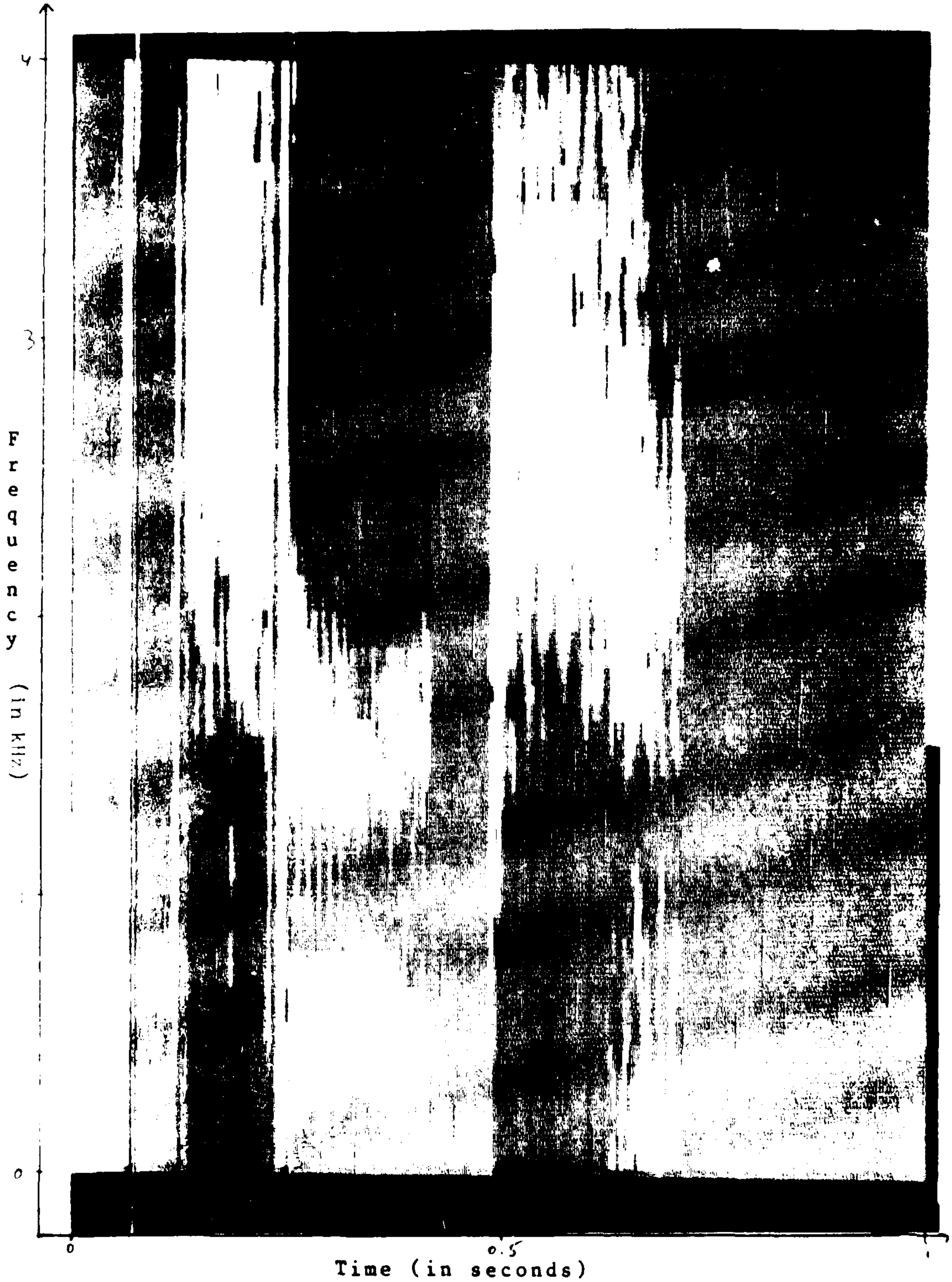

Figure 9. Hamming Window (church)

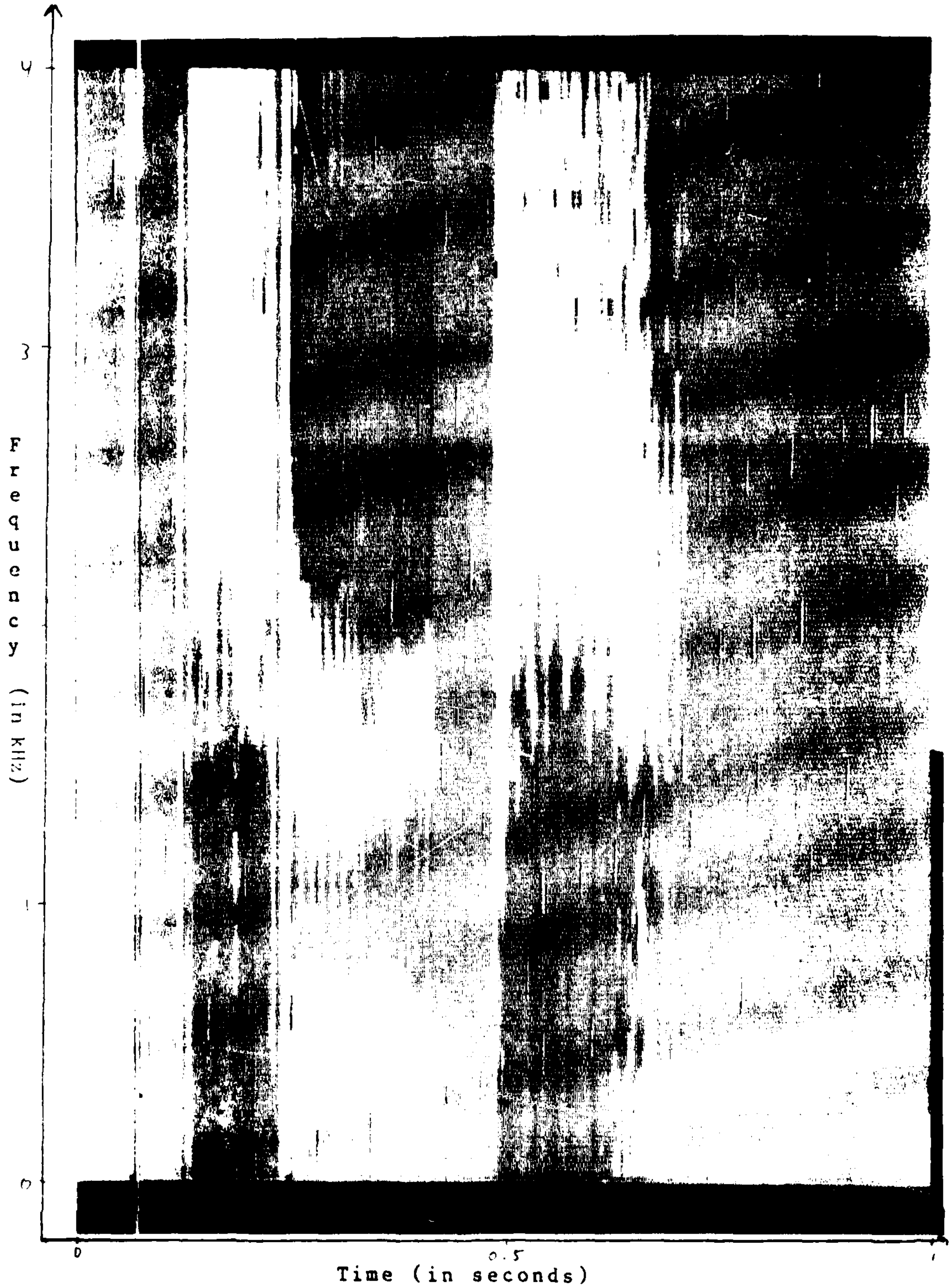

Figure 10. Hanning Window (<u>ch</u>ur<u>ch</u>)

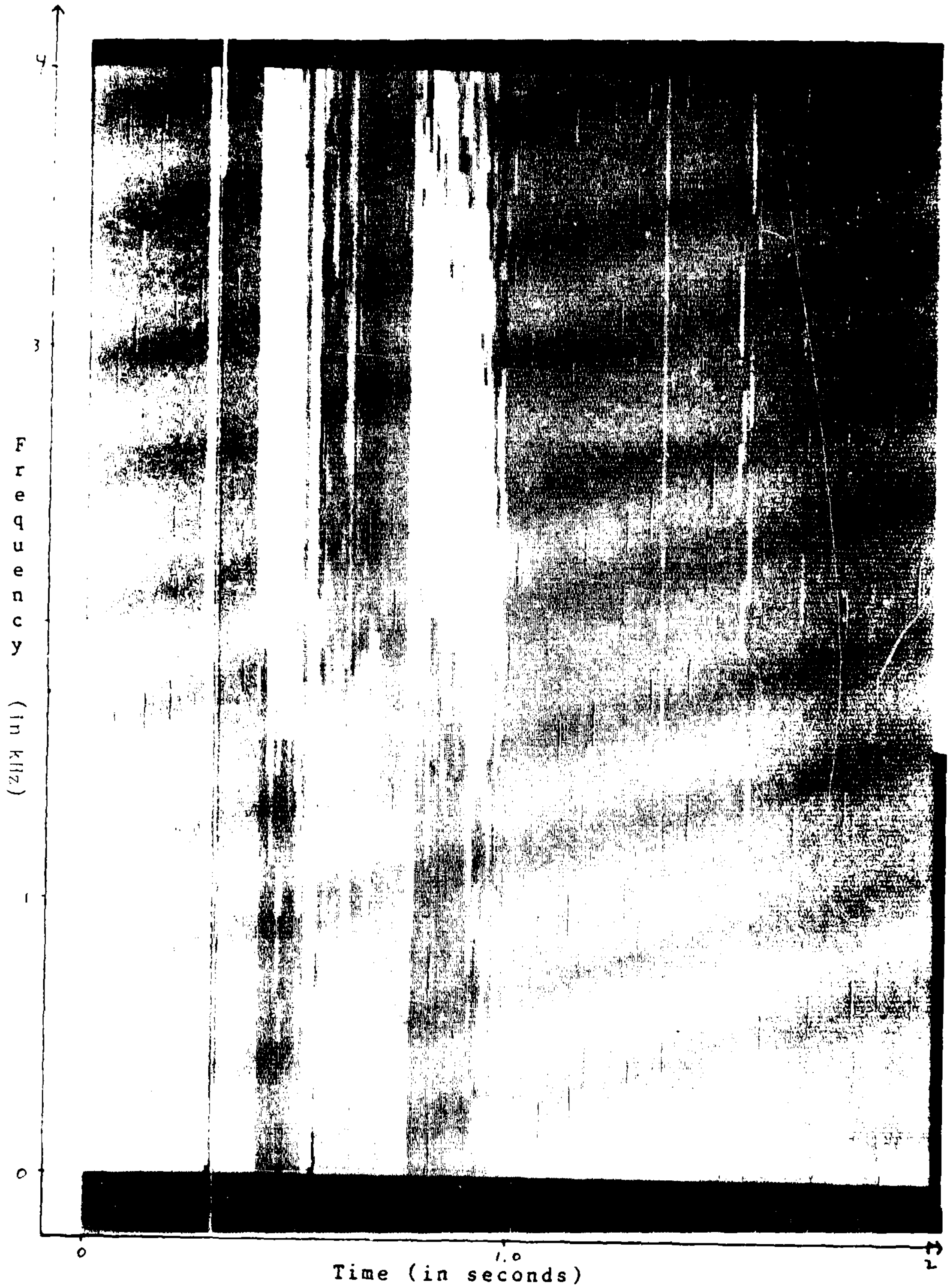

Figure 11. Rectangular Window (<u>ch</u>ur<u>ch</u>)

Figures 12 and 13 demonstrate the effect of pitch on a vowel pattern. The vowel used is the vowel I (as in eve). Figure 12 is the spectrogram of the low pitched vowel and figure 13 is the high pitch. It can be seen that the horizontal striations, corresponding to the individual pitch harmonics, are slightly lower in frequency for the low pitch spectrogram than the high pitch spectrogram.

Both spectrograms were generated using the Hamming window with N being 256. This resulted in a narrow-band spectrogram with a filter bandwidth of 43 Hz. The DFT sections were overlapped by 75 per cent. The threshold for both figures is -20 dB with a step size of 2.5 dB. The resolution of both displays is 128 vertical pixels by 64 horizontal pixels.

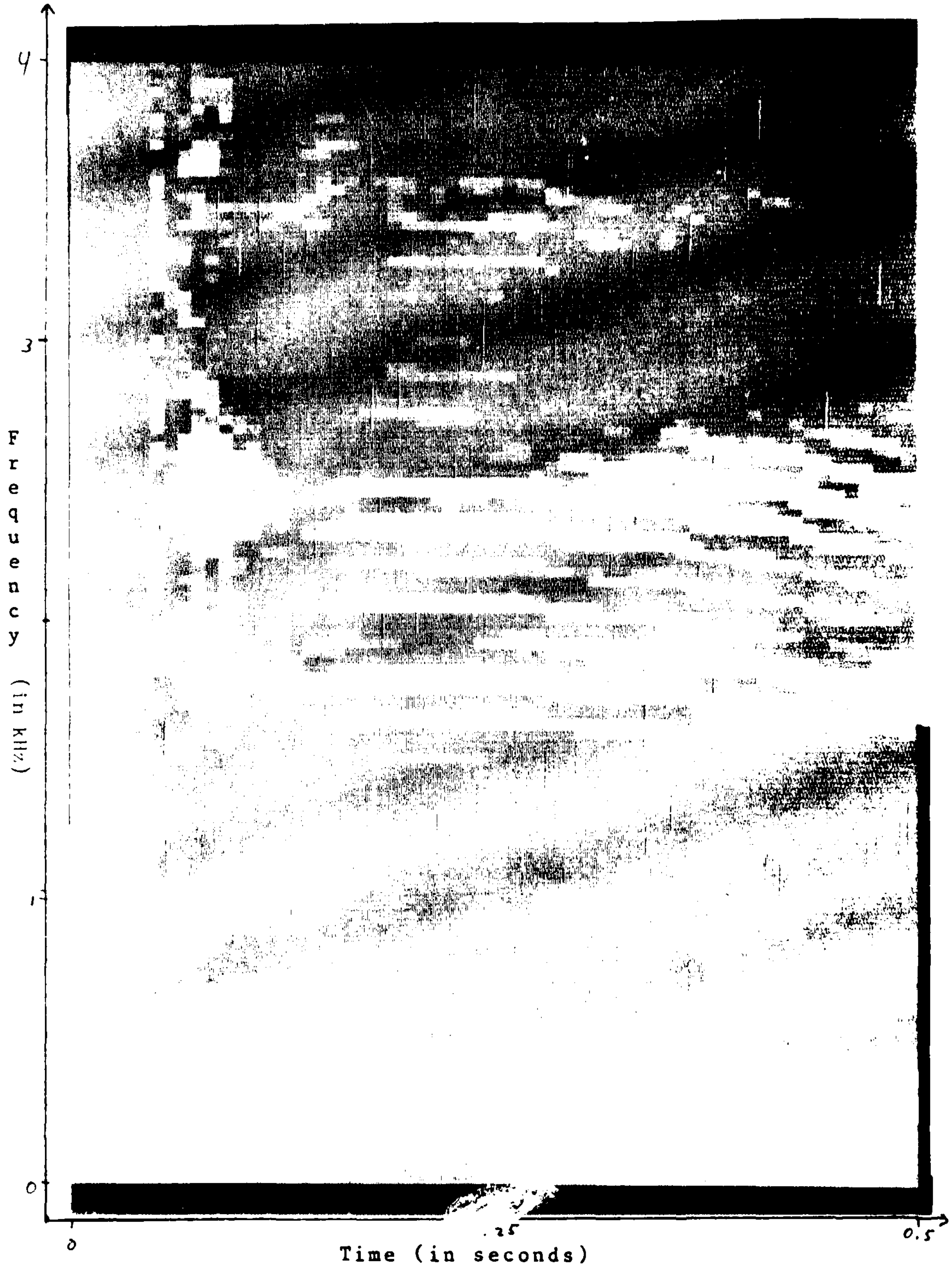

Figure 12. Low Pitch Vowel (I)

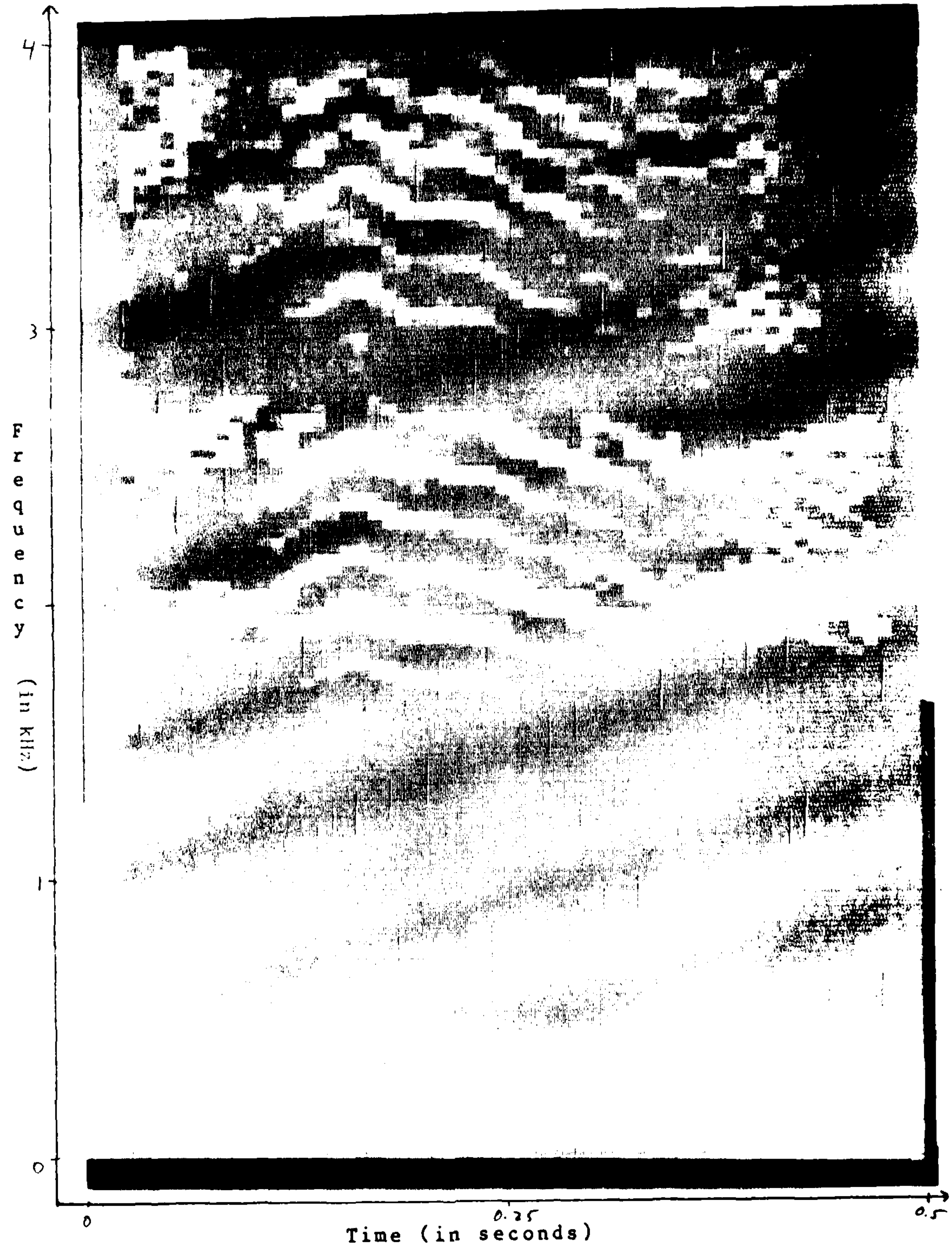

Figure 13. High Pitch Vowel (I)

Thus far most of the spectrograms shown have been wide-band. The wide-band spectrogram is characterized by vertical striations, corresponding to the individual pitch periods. This is the result of the good time resolution of the wide-band spectrogram. The narrow-band spectrogram is characterized by horizontal striations, corresponding to the individual pitch harmonics. This is the result of the good frequency resolution of the narrow-band spectrogram (Ref 8:58). Figures 14 through 17 show the results of narrow-band spectrograms with each of the four windows. The filter bandwidths for each window are listed below for comparison with the typical narrow-band spectrogram filter bandwidth of 45 Hz.

The sentence used is "Ask her to stay." Figure 14 shows the 4-sample Kaiser-Bessel window with a N of 256, and a filter bandwidth of 56 Hz. Figure 15 shows the Hamming window with a N of 256, and a filter bandwidth of 43 Hz. Figure 16 shows the Hanning window with a N of 256, and a filter bandwidth of 47 Hz. Figure 17 shows the rectangular window with a N of 128, and a filter bandwidth of 63 Hz. The DFT sections were overlapped 50 per cent in all cases except for the rectangular window, where no overlapping was done. The threshold for the first three displays is -15 dB with a step size of 2 dB. The threshold for the last display is -10 dB with a step size of 2 dB. The resolution of the displays is 128 vertical pixels by 128 horizontal pixels.

There appears to be little difference between the spectrograms produced by using the Kaiser-Bessel, Hamming or Hanning window, though the Hamming spectrogram does seem to have sharper detail. The spectrogram produced by using the rectangular window does show the basic pattern but it lacks the detail of the other spectrograms.

Figure 14. Kaiser-Bessel Window (Ask her to stay.)

Figure 15. Hamming Window (Ask her to stay.)

Figure 16. Hanning Window (Ask her to stay.)

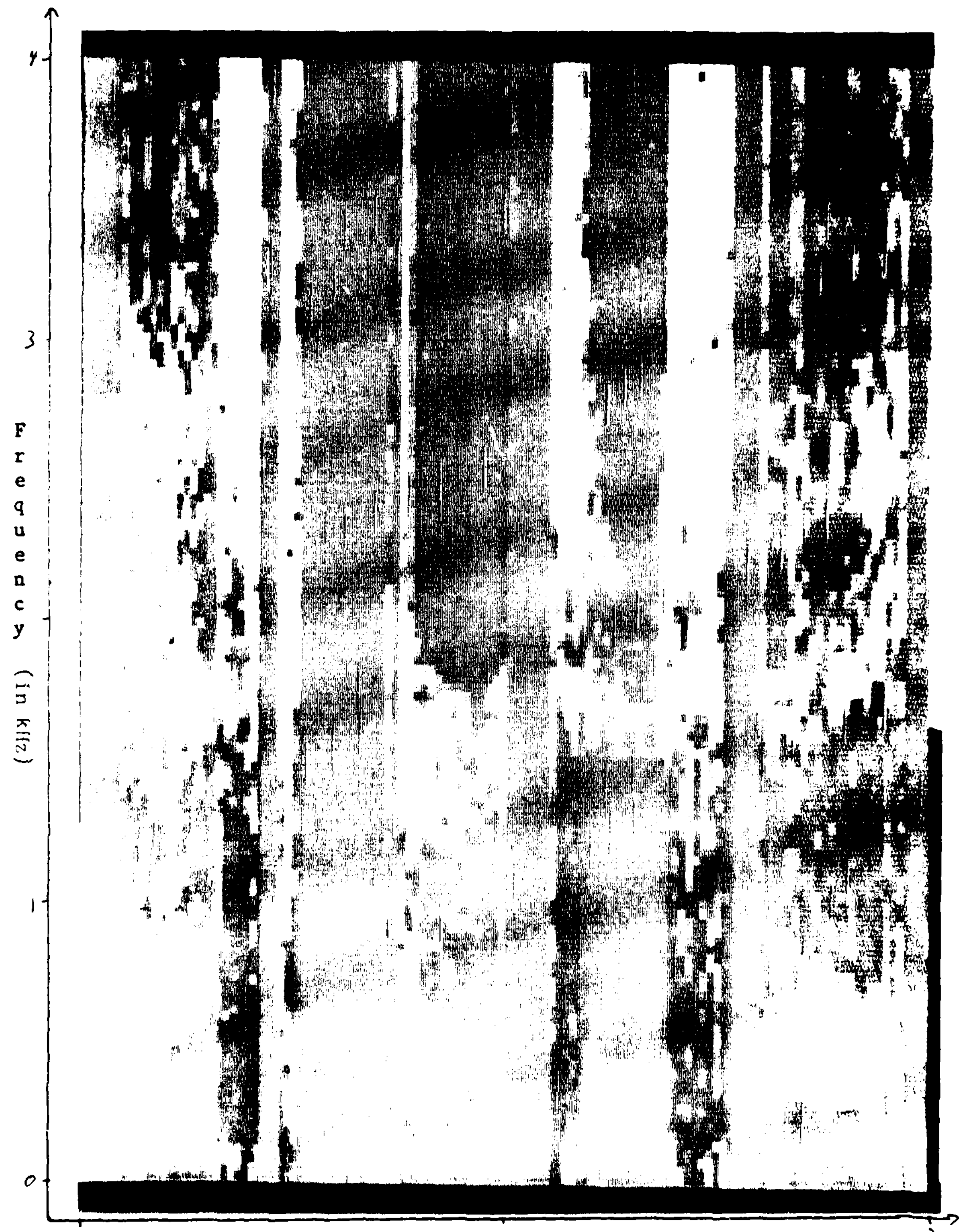

Figure 17. Rectangular Window (Ask her to stay.)

The spectrograms in this chapter have shown the six sound-pattern groups, the effects of different windows on the same section of speech, the effect of pitch on a vowel pattern and the differences between narrow and wide-band spectrograms. These spectrograms illustrate the main features of the display system in displaying both narrow and wide-band spectrograms. The capabilities of the system are summarized in the next chapter, along with some recommendations on how the system can be improved.

VI. Conclusions and Recommendations

A sound spectrogram display system has been designed, developed, and implemented. The system can display both wide-band and narrow-band spectrograms on a television monitor. Sections of the spectrogram can be isolated through the use of a pair of cursors for further analysis. The system is flexible in that it gives the user control over several options. These options include: the window function, the length (N) of the DFT, the amount of overlap of the DFT sections, the number of picture elements that make up the display, and the type of display (black on white, or white on black). The remainder of this chapter will be devoted to the conclusions and recommendations regarding this report.

Conclusions

The display system is capable of generating both wide-band and narrow-band spectrograms on a television monitor.

The time required to generate a spectrogram is dependent on several factors. The major influence is the length (N) of the DFT. The bigger N is, the longer it takes. Increasing the amount of overlap, also increases the time required. The amount of interpolation required to fill the chosen display size has an effect: the less interpolation, the better. The window function used plays a minor role in this regard. The 4-sample Kaiser-Bessel window requires the most computation time. The Hamming and Hanning windows are approximately equal in this regard, while the rectangular window requires the least computation time.

The wide-band spectrograms (with N equal to 64, 50 per cent overlap, and a display size of 64 vertical pixels by 256 horizontal pixels) shown in Chapter V, required between six and seven minutes for production depending on the window used. The narrow-band spectrograms (with N equal to 256, 50 per cent overlap, and a display size of 128 vertical pixels by 128 horizontal pixels) required between nine and eleven minutes for production depending on the window used.

Based on the above time considerations and the figures of merit for each window listed in Table II, the 4-sample Kaiser-Bessel window should be used when a wide-band spectrogram is desired and the Hamming window should be used when generating a narrow-band spectrogram. The 4-sample Kaiser-Bessel window is far superior in suppressing spectral leakage but in the case of the narrow-band spectrogram the extra time required to compute the window is not warranted. In addition, the Hamming window has a narrower main lobe resulting in better frequency resolution, which is more important in the narrow-band case than the wide-band case.

The amount of overlap used in generating the spectrograms should be 50 percent so as to adaquately cover as much digitized speech as possible. Only when transforming a small section of digitized speech should the overlap be increased to 75 percent.

Spectral emphasis should be used in generating the spectrogram display. Without the emphasis too much of the high frequency detail is lost. Lowering the threshold of the display does help bring out some of this lost detail, but at the cost of losing some of the detail at the lower frequencies due to the dynamic range of the display being exceeded.

Once a spectrogram has been generated, the display can be reshaped in approximately two minutes.

A problem exists in the Tektronics 4632 Video Hard Copy unit in that the copies it produces do not accuately depict the display on the television monitor. The unit is not capable of reproducing the sixteen shades of gray that make up the spectrogram display. This is a problem because a significant amount of detail is lost if copies are made of spectrogram displays for future analysis.

Finally, two cursors can be displayed on the spectrogram. The section of the display (64 vertical slices) following the leftmost cursor can be expanded. A new display with a horizontal resolution of 64 lines is generated and stored in the disk file BLOWUP. Also, the digitized speech that generated the display between the two cursors, can be stored in the disk file, DSPOUT, for further analysis.

Recommendations

The following recommendations are presented to indicate areas of possible improvement and/or expansion. Some of the recommendations follow directly from the conclusions stated above. Other recommendations deal with modifications that might be required due to proposed expansion and development of the speech processing lab.

The utility of the system would be greatly enhanced if the time required to generate a spectrogram were reduced. A prime area for improvement is the large number of disk acceses required. A more efficient method of handling data storage and data transfers would, most likely, lead to a significant improvement in this area.

A related matter is the arrangement of the disk file DISPLAY. The data is packed and stored one time (vertical) slice after another. The monitor interface boards, however, write the memory out one horizontal (frequency) slice after another. This problem was corrected by rotating the yoke of the monitor ninety degrees so that the monitor interface boards wrote out vertically instead of horizontally. If another television monitor is used where it is either impossible or undesirable to rotate the yoke, an efficient method must be developed to rearrange and repack the DISPLAY file.

As currently implemented, the programs (DISPLAY, CURSOR and ONE) that run on the S-100 system rely on operating through the use of a terminal. An operating system is being developed for the S-100 which will allow programs to be called from the NOVA. This will elimanate the need for connecting the S-100 to a terminal. The programs will have to be modified to accept commands from the NOVA, in place of using the Pause command to control operation of the programs.

The placement of the cursors is another area for possible improvement. Cursor placement is currently controlled by values transferred from the NOVA to the S-100. A method of controlling cursor placement through the S-100 would provide the user more direct control over the cursors. Perhaps a set of switches could be implemented to provide this function. One switch could move the cursor to the left and another move the cursor to the right. Once the cursor position is acceptable, the user could signify this by setting another switch. The position of the cursor would then be transferred to the NOVA for use in the cursor routines. Each cursor would have its own set of switches.

The primary area for improvement in the display system is in the time required to generate a spectrogram display. However, if in a particular application some of the flexibility of the system can be sacrificed, the system can be tailored to that particular application and this should result in a decrease in the display time. Therefore, the display system should be used as is until the user becomes familiar with all the capabilities of the system and then tailor the system to meet the needs of the research in progress.

Bibliography

1. Cochran, William T., et al. "What is the Fast Fourier Transform?," IEEE Transactions on Audio Electrocoustics, AU-15: 45-55 (March 1967).

2. Cromenco Fortran IV. Cromenco Inc., Mountain View, California, 1979.

3. Discrete Fourier Transform User's Manual. 093-000104-01. Data General Corporation (DGC), Westboro, Massachusetts, 1978.

4. Fortran IV User's Manual. 093-000053. Data General Corporation (DGC), Westboro, Massachusetts, 1978.

5. Griess, Wayne D. Construction of a General Purpose Command Language for Use in Computer to Computer Dialog. MS Thesis. Wright-Patterson AFB, Ohio: Air Force Institute of Technology, September 1980.

6. Harris, F.J. "On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform," Proceedings of the IEEE, 66: 51-83 (January 1978).

7. Mermelstein, Paul. "Computer-Generated Spectrograms for On-Line Speech Research," IEEE Transactions on Audio Electrocoustics, AU-15: 45-55 (March 1967).

8. Oppenheim, Alan V. "Speech Spectrograms using the Fast Fourier Transform," IEEE Spectrum, 7: 57-62 (August 1970).

9. Potter, Ralph K., et al. Visible Speech. New York: D. Van Nostrand Company, Inc., 1947.

10. Real Time Video Digitizer. Tecmar Inc., Cleveland, Ohio, 1978.

11. Strong, William J. and E. Paul Palmer. "Computer-Based Sound Spectrograph System," J. Acoust. Soc. Am., 58: 899-904 (October 1975)

Appendix A

User's Manual

Contents

I. Introduction

This user's manual is intended to provide the user with enough information and background to utilize the system effectively. In the next section the programs involved are described in general. Following that, the user is taken step-by-step through the operation of the system. Particular attention is paid to possible problem areas. The video digitizer board set is described with the emphasis on the various switch settings. This manual concludes with a discussion of the possible problem areas and how to avoid them.

II. Program Description

NOVA Programs

TRANSFORM. This program accepts your choices for the various parameters that govern the computation of the spectrogram. Digitized speech is input from the disk file DSPEECH. The chosen window is applied to the speech and the result is transformed into the frequency domain. The transformed speech is then stored in the disk file WINDOW. The program PREPARE is then called into execution.

PREPARE. This program inputs transformed speech from the disk file WINDOW, one DFT section at a time. Each section is normalized. The resulting spectral points are interpolated (if necessary) to fill the desired vertical resolution of the display. The product of this effort is stored in the disk file INTRP. When program execution has come to an end, the program SHAPE is called into execution.

SHAPE. This program accepts your choices for setting the display levels. Each spectral point is assigned a display level. These levels are packed together, four levels to an integer, and then stored in the disk file DISPLAY. When the DISPLAY file is complete, the file is transferred to the S-100 for display. At this point you have three options: the display can be reshaped, the program EXAMINE can be called into execution, or the program can be terminated.

EXAMINE. This program accepts your inputs for the desired cursor position. The inputs are transferred to the S-100 for display. You can accept the cursor position or re-enter new inputs. Once you accept the cursor positions, you have four options. The first is a new display can be generated representing the first 64 vertical lines following the leftmost cursor. The display is stored in the file BLOWUP. Second, the digitized speech representing the spectrogram between the cursors is stored in the file DSPOUT. The speech is obtained from the original speech file DSPEECH. The third option is an alternative to the second. The program RECON can be called and the transformed speech in the file WINDOW is inverse transformed and stored in DSPOUT. Finally, the user can choose to do none of the above and terminate the program.

SENDISP and SENDWOB. These programs transfer already developed display files to the S-100 for display. SENDISP transfers a black on white display, while SENDWOB transfers the opposite type of display.

S-100 Programs

DISPLAY. This program accepts the display file from the NOVA and stores it in memory. The monitor interface DMA board is turned on and thus the spectrogram is displayed. The program pauses while the display is on. A carriage return turns off the display and generates another pause. At this point the user can either terminate the program or can accept another display file from the NOVA.

CURSOR and ONE. These programs accept the desired cursor positions from the NOVA. The memory locations that will be overwritten by the cursors are stored in memory that is not being displayed. Program CURSOR displays cursors that consist of two black vertical lines followed by two white vertical lines, while program ONE generates one black line followed by one white line. The display is turned on and the program pauses. A carriage return turns off the display and causes another pause. At this point you can either terminate the program or can accept new cursor positions from the NOVA. If new cursor positions are accepted, the memory orginally overwritten by the cursors is restored. New cursors are then displayed in the same manner as described above.

This completes the description of the developed programs on both the NOVA and S-100 systems. Table IV lists the subroutines and libraries that must be loaded along with the programs if modifications are made. The next section describes the generation of the spectrograms.

Table IV

Program Loading Table

Program	System	Subroutines	Libraries
TRANSFORM	NOVA	NONE	NDFT.LB FORT.LB
PREPARE	NOVA	NONE	FORT.LB
SHAPE	NOVA	PASSDAT PASCURS	FORT.LB
EXAMINE	NOVA	PASSDAT PASCURS	FORT.LB
RECON	NOVA	NONE	NDFT.LB FORT.LB
SENDISP	NOVA	PASSDAT	FORT.LB
SENDWOB	NOVA	PASCURS	FORT.LB
DISPLAY	S-100	NONE	NONE*
CURSOR	S-100	NONE	NONE*
ONE	S-100	NONE	NONE*

* The fortran library is automatically searched during loading.

III. Generating the Spectrogram

The digitized speech to be transformed must be loaded into the disk file DSPEECH. Inputing speech through the S-100 system automatically loads the DSPEECH file. Additionally, you can generate your own DSPEECH file (perhaps a prototype phoneme set) for analysis. Either way, the spectrogram grenerating process begins with calling the program TRANSFORM. This program will present a series of choices. These choices will be looked at step-by-step. The prompt that will appear on the NOVA terminal will be presented followed by a discussion of the considerations involved and the possible problem areas involved in each choice.

```
1.  PICK A WINDOW
    ENTER 1 FOR A RECTANGULAR WINDOW
    ENTER 2 FOR A HANNING WINDOW
    ENTER 3 FOR A HAMMING WINDOW
    ENTER 4 FOR A 4-SAMPLE KAISER-BESSEL WINDOW
```

The default choice is the rectangular window. The window chosen is reflected in the next prompt. For this example a rectangular window is chosen.

2. RECTANGULAR WINDOW

N	BANDWIDTH
16	500
32	250
64	125
128	63
256	31
512	16
1024	8

PICK A VALUE FOR N

If a wide-band spectrogram is desired (for good time resolution) pick a value for N that results in a bandwidth of approximately 300 Hz. For a narrow-band spectrogram (for good frequency resolution), pick a value for N that results in a bandwidth of approximately 45 Hz. (NOTE: the bandwidths are different for each window).

If a mistake is made and the N chosen is not one of the choices on the monitor, that is, if 46 was entered instead of 64; the program should be halted by issueing a CTRL-A command and restarted. The program should also be restarted if the window choice displayed is not the one desired.

3. PICK PERCENT OVERLAP (REAL)

No overlap is necessary if a rectangular window is used. At least fifty percent overlap should be used for the other three windows. Suggested values for use are fifty percent overlap for normal situations and seventy five percent overlap when a small section of speech is to be analyzed.

If, by mistake, an overlap of greater than one hundred percent is entered, the prompt:

PERCENT OVERLAP MUST BE LESS THAN 100 PERCENT

is displayed and the program asks for another input.

4. ENTER NUMBER OF INPUT SAMPLES (23040 IF
   INPUT THRU S-100)

The number of input samples is the number of integers stored in the disk file DSPEECH. If speech is input through the S-100 system the number of integers is 23040.

5. VERTICAL RESOLUTION CHOICES: 512,256,128 OR 64
   PICK VERTICAL RESOLUTION

The best choice for the vertical resolution depends on your choice for N. Table V presents the choices that result in least interpolation for each choice of N. The less interpolation required, the faster the spectrogram will be generated. Of course, if time is not a consideration, the choice can be made on the basis of the resulting display.

Table V

Recommended Vertical Resolution Choices

N	VERTICAL RESOLUTION
16	64
32	64
64	64
128	64
256	128
512	256
1024	512

6. HORIZONTAL RESOLUTION CHOICES: 256,128 OR 64

PICK HORIZONTAL RESOLUTION

The horizontal resolution choices available depend on your choice for the vertical resolution. The display size is limited to 32768 pixels. (This number is found by multiplying the vertical by the horizontal resolution). If the vertical resolution is 256 the maximum horizontal resolution available is 128 (256 X 128 = 32768). The prompt will reflect this.

In most cases you will want to choose the maximum available resolution to observe as much speech as possible. However, you may also pick a smaller resolution if your desire is to look at a small section of speech. A smaller resolution will also require less time to generate.

7. PICK STARTING BLOCK OF INPUT SPEECH

The digitized speech in the file DSPEECH (if input through the S-100 system) is organized in blocks of 256 integers. This feature allows you to skip the initial blocks of speech. For wide-band spectrograms (small N) twenty is a good pick for the starting block. This allows the program to pass over the noise that usually precedes speech input through the S-100 system using the A2D routine. (This is due to the time delay in starting the tape recorder upon hearing the start tone). For narrow-band spectrograms zero is a good pick because the spectrograms cover more time samples (large N) and there is less risk of missing the beginning of the speech. This number can also be adjusted once a display is generated to zero in on a particular section of speech. To do this a rough estimate is needed. Each vertical line on the display represents N time samples. Estimate the number of vertical lines prior to the beginning of the speech to be analyzed. Multiply this number by N times the amount of overlap and divide by 256. This number gives you the number of additional blocks to skip. You can then retransform the speech using this new value to isolate the desired speech.

8. NOISE THRESHOLD IS 10000
   TO CHANGE THE THRESHOLD ENTER 1 -

If the maximum value of any spectral section (represented by an array containing the transformed speech) is greater than the noise threshold, the array is normalized by dividing through the array by that maximum value. If the maximum value is less than the threshold, the array is normalized by dividing through the array by 100 times the thresho This puts puts spectral sections with maximum values under the threshold at a forty decibel disadvantage. The final result on the display is a blank vertical line.

If the digitized speech was input through the S-100 system, 10000 provides adequate noise suppression. You may want to change the noise threshold if either there appears to be excessive noise on the display or sections of the spectrogram appear to be missing. In the first case you will want to increase the noise threshold. The disk file

CONSTANTS contains the maximum values for each spectral section. You can use either the TYPE or PRINT command to get a listing of the maximum values. You can use this list to obtain a new threshold. There should be an obvious break between noise values and speech values. It is best to keep the noise threshold as low as possible because it is better to include some noise on the spectrogram than to elimanate any speech.

The opposite case can occur if the digitized speech was recorded at too low a level. The display can be regenerated using a lower threshold but the best course of action is to re-record the speech and start over. This is because the speech and the background noise can not be sufficiently separated by lowering the threshold to provide a good display.

9. WHITE ON BLACK DISPLAY DESIRED? (YES=1)

A white on black display has the highest intensity level as white and the lowest intensity as black. The more common spectrogram display has the highest intensity level as black and the lowest as white. A no answer will generate this type of display. This choice is left to your personnal preference.

10. LINEAR INTERPOLATION DESIRED? (YES=1)

This prompt occurs if the number of frequency points (N/2) yielded by each DFT is less than the chosen vertical resolution. You have two choices on how to fill the desired resolution. If linear interpolation is desired, a yes answer, the frequency points will be linearly expanded to fill the desired resolution. The other option, a no answer, is to repeat the frequency points as necessary. For example, if the desired resolution is twice the number of frequency points, frequency point one will become resolution points one and two. Frequency point two will become resolution points three and four, and so on. This choice, like the one above, is left to your personal preference.

This completes the discussion of the various choices presented you by the program TRANSFORM. If a mistake is made in entering your choices, the best course of action is to stop the program (CTRL-A) and start the program over.

If no mistakes have been made, the program will then display:

INPUT START TIME = XX XX XX.

This signifies the beginning of program execution. When program execution is complete, the following prompt:

TRANSFORM STOP TIME = XX XX XX

is displayed. The program PREPARE is then called into execution via the Chain command.

The monitor will then display:

PREPARE START TIME = XX XX XX.

When the program comes to an end the prompt:

PREPARE STOP TIME = XX XX XX

occurs. These prompts are used to let you know which program is currently in execution so that if an error occurs you know which program is at fault. When the PREPARE program is finished, the program SHAPE is called into execution via the Chain command. You are then presented with a series of prompts.

1. PICK THE THRESHOLD IN DECIBELS (REAL)
2. PICK THE STEP SIZE IN DECIBELS (REAL)

Table VI lists the suggested choices for both the threshold and the step size for each value of N. The resulting display should contain a wide range of intensities. If there is too much of a concentration of high intensity values, the display can be reshaped with either a higher threshold or a larger step size, or both, to produce a wider range of intensities. If there is too much of a concentration of low intensity values, the opposite holds true. Either lowering the threshold or decreasing the step size, or both, will result in a wider range of intensities. You will have to use your own judgement in making these decisions.

Table VI

Recommended Display Level Settings

N	THRESHOLD	STEP SIZE
16	-3	1.5
32	-5	1.7
64	-10	2.0
128	-16	2.2
256	-20	2.5
512	-30	3.0
1024	-45	4.0

3. ENTER THE NUMBER OF LOWER LEVELS TO BE SUPPRESSED (INTEGER)

In most cases zero should be chosen here. This choice is provided primarily for use when the spectrogram is being reshaped to provide higher contrast. For example, if you choose four levels to be suppressed, the spectral points that would normally be assigned brightness levels one, two or three will be assigned brightness level zero. This feature can be used to make certain features stand out more strongly at the expense of giving up some of the subtle details.

4. IF SPECTRAL EMPHASIS IS DESIRED ENTER 1

Spectral emphasis should be used and a yes answer generates the following prompt.

5. SPECTRAL EMPHASIS IS THE INCREASE OF SO MANY DB'S PER OCTAVE FROM A STARTING FREQUENCY

ENTER DB'S PER OCTAVE (REAL)

ENTER STARTING FREQUENCY (REAL)

Spectral emphasis of 6 decibels per octave starting at 500 Hz should be used. The values suggested for use for the threshold and the step size were based on six decibels of spectral emphasis. You must remember to adjust the values for the threshold and the step size if you choose to use a different amount of spectral emphasis.

When the amount of spectral emphasis is chosen, the prompt:

DISPLAY START TIME = XX XX XX

is displayed. When the display file is completely generated, the prompt:

DISPLAY STOP TIME = XX XX XX

appears. When the NOVA is ready to transfer the display file the prompt:

*************READY**************

appears. To receive the display file the program DISPLAY must be brought up on the S-100 system.

Before DISPLAY will receive any data from the NOVA, the program pauses. A carriage return (on the S-100 terminal) will begin the data transfer process. While the process is in operation, the prompt:

TRANSFERRING DATA

will appear several times on the NOVA terminal. When the process is complete, the television monitor is turned on to display the spectrogram and the program DISPLAY pauses; while on the NOVA the prompt:

TO RESHAPE THE SPECTRUM ENTER 1

appears.

At this point you can either accept the spectrogram or choose to reshape it. If you choose to reshape the display, you will then enter your new choices for threshold, step size, etc. When a new display is ready to be transferred, the prompt:

*************READY**************

will appear again.

The old spectrogram is still shown on the television monitor. A carriage return will then turn off the monitor and generate another pause. As before, a carriage return will begin the transfer of data, and when the transfer is complete, the new spectrogram will be displayed on the monitor. Also, the NOVA will again display the prompt:

TO RESHAPE THE SPECTRUM ENTER 1.

This process, of reshaping and displaying spectrograms, can be repeated as often as necessary.

To terminate this process on the S-100 system, input a carriage return. This will turn off the display and result in another pause. Enter a "T" followed by a carriage return to terminate the program DISPLAY. If you by mistake, terminate the program while the monitor is still on, you will have to return to the RDOS mode. This is because while the monitor is on, you will not be able to access another program form the disk. If you try to access a program, a read error will occur because the monitor interface boards (through the use of DMA) have almost total control of the bus which prevents the proper operation of the floppy disk system. Once back in RDOS, you can turn off the DMA by entering the following command:

0 00 FF

You can now boot up CDOS again.

To terminate this process on the NOVA, enter a zero (or any other number except one) when asked to reshape the spectrum. (Table VII summarizes the spectrogram display procedure commands). The following prompt:

CALL PROGRAM EXAMINE.SV? (YES=1)

will then appear. A yes answer begins execution on EXAMINE which contains the cursor routines. A no answer terminates the program SHAPE. The cursor routines will be described following the next section on transferring already developed display files.

Table VII

Display Transfer Command Summary

NOVA	S-100
READY	
	A.DISPLAY
	PAUSE (MONITOR OFF)
	(CR)
TRANSFERRING DATA	
TO RESHAPE THE SPECTRUM ENTER...	PAUSE (MONITOR ON)
(1)	
PICK THE THRESHOLD IN...	
.	
.	
.	
READY	
	(CR)
	PAUSE (MONITOR OFF)
	(CR)
TRANSFERRING DATA	
TO RESHAPE THE SPECTRUM ENTER...	PAUSE (MONITOR ON)
(1)	
.	.
.	.
.	.
TO RESHAPE THE SPECTRUM ENTER...	
(0)	
CALL PROGRAM EXAMINE.SV...	
	(CR)
	PAUSE (MONITOR OFF)
	(T CR)
	A.

IV. Transferring Display Files

There may be times when you desire to display spectrograms that have been previously generated. The programs SENDISP and SENDWOB allow you to do this. (The only difference between the two programs is that SENDISP produces a black on white display, while SENDWOB produces a white on black display). The spectrogram to be displayed must be stored in the disk file DISPLAY.

Call the program of your choice and the following prompt:

ENTER VERTICAL RESOLUTION -

will appear. Enter the vertical resolution of the display and the following prompt:

ENTER HORIZONTAL RESOLUTION -

will appear. When the resolution is entered, the prompt:

*************READY**************

appears and the program is ready to begin the transfer. To receive the display file the program DISPLAY must be brought up on the S-100 system.

Before DISPLAY will receive any data from the NOVA, the program pauses. A carriage return (on the S-100 terminal) will begin the data transfer process. While the process is in operation, the prompt:

TRANSFERRING DATA

will appear several times on the NOVA terminal. When the process is complete, the monitor is turned on to display the spectrogram and the program DISPLAY pauses; while the program SENDISP (or SENDWOB) terminates and the following message:

STOP, **************DONE***************

appears on the NOVA terminal. This procedure is summarized in Table VIII.

Table VIII

SENDISP Command Summary

NOVA	S-100
R SENDISP (OR SENDWOB) ENTER VERTICAL RESOLUTION - ENTER HORIZONTAL RESOLUTION - ***READY*** TRANSFERRING DATA STOP, ***DONE*** R	 A.DISPLAY PAUSE (MONITOR OFF) (CR) PAUSE (MONITOR ON) . . . (CR) PAUSE (MONITOR OFF) (T CR) A.

V. Using the Cursor Routines

The cursor routines are contained in the program EXAMINE. The program can be called from the program SHAPE or started on its own. To start the program on its own, the DISPLAY file must contain the spectrogram to be examined and the file must be loaded into S-100 memory. To load a DISPLAY file into memory you can use the program SENDISP (or SENDWOB) as previously described in section IV. Once this is accomplished, the program can be called into execution.

The first prompt to appear will be:

CURSOR WIDTH (SINGLE=1,DOUBLE=2) =.

This allows the program to compensate for cursor width. Your choice of cursor width determines which cursor program you use on the S-100 system. Program ONE produces single width cursors and CURSOR produces double width cursors. When the cursor width is entered, the prompts:

```
AVAILABLE CURSOR CHOICES: 0 TO 255
CURSOR POSITION ONE =
CURSOR POSITION TWO =
```

appear. The maximum available cursor choice depends on the horizontal resolution of the display. A zero entry puts a cursor on the leftmost edge of the display, while entering the maximum value puts the cursor on the rightmost edge of the screen. Enter your choices based on these limits.

Once these values are entered into the computer, the prompt:

**********READY*********

signifies the NOVA is ready to transfer the cursor positions to the S-100. To receive the cursor positions start execution of either CURSOR, or ONE. Both programs operate in the same manner. The program initially pauses and the television monitor is turned on. A carriage return (on the S-100 terminal) will begin the transfer and turn off the monitor. When the data has been transferred the monitor is turned on to display the spectrogram and the cursors, and the program CURSOR (or ONE) pauses; while on the NOVA the prompt:

**********DONE**********

appears.

The next prompt:

ACCEPT CURSOR POSITION? (YES=1)

allows you to either accept the cursor positions and to continue on to the other routines, or to enter new choices for the cursor positions. If you choose to enter new cursor positions, the prompt:

**********READY*********

will appear again. A carriage return (on the S-100 terminal) will turn off the monitor and generate a pause. As before, a carriage return will begin the transfer of data, and when the transfer is complete, the spectrogram with the new cursor positions will be displayed on the monitor. Also, the NOVA will again display the prompt:

ACCEPT CURSOR POSITION? (YES=1).

This process (summarized in Table IX), of entering and displaying new cursor positions, can be repeated as often as necessary.

Table IX

Cursor Display Command Summary

NOVA	S-100
CURSOR WIDTH (SINGLE=1,DOUBLE=2) =	
(2)	
AVAILABLE CURSOR CHOICES: 0 TO 255	
CURSOR POSITION ONE =	
CURSOR POSITION TWO =	
READY	
	A.CURSOR (OR ONE)
	PAUSE (MONITOR ON)
	(CR)
DONE*	PAUSE (MONITOR ON)
ACCEPT CURSOR POSITION? (YES=1)	
(0)	
AVAILABLE CURSOR CHOICES: 0 TO 255	
CURSOR POSITION ONE =	
CURSOR POSITION TWO =	
READY	
	(CR)
	PAUSE (MONITOR OFF)
	(CR)
DONE*	PAUSE (MONITOR ON)
ACCEPT CURSOR POSITION? (YES=1)	.
(1)	
BLOWUP SECTION OF SPEECH...	.
	.
	(CR)
	PAUSE (MONITOR OFF)
	(T CR)
	A.

To terminate this process on the S-100 system, input a carriage return. This will turn off the monitor and result in another pause. Enter a "T" followed by a carriage return to terminate the program CURSOR (or ONE). Remember, as

AD-A100 791 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOO--ETC F/6 9/2
DESIGN AND DEVELOPMENT OF A GRAPHICS SYSTEM FOR DISPLAY OF SOUN--ETC(U)
DEC 80 P A DUNDAS
UNCLASSIFIED AFIT/GE/EE/80D-17 NL

2 OF 2
AD
A100791

END
DATE
FILMED
7-81
DTIC

mentioned previously, you do not want to terminate the program while the monitor is on.

To terminate the process on the NOVA, accept the cursor positions by entering a one. The following prompt:

```
BLOWUP SECTION OF SPEECH FOLLOWING THE
   LEFTMOST CURSOR? (YES=1)
```

will then appear. This option allows you to display an expanded spectrogram with a horizontal resolution of 64. Of course this option should not be used if the original horizontal resolution was 64. Also, the leftmost cursor must be at least 64 short of the maximum available cursor position or an error will result. If you desire to blowup the spectrogram, the horizontal resolution switches on the monitor interface DMA board must be changed. (The monitor inerface boards are described in the next section). The expanded spectrogram is stored in the disk file BLOWUP, and transferred into S100 memory in the same manner as the DISPLAY file was originally transferred. This procedure is summarized in Table X.

Table X

BLOWUP Transfer Command Summary

NOVA	S-100
BLOWUP SECTION OF SPEECH...	
LEFTMOST CURSOR (YES=1)	
(1)	
READY	
	A.DISPLAY
	PAUSE (MONITOR OFF)
	(CR)
TRANSFERRING DATA	
REPLAY ORIGINAL SPEECH? (YES=1)	PAUSE (MONITOR ON)
.	.
.	.
.	.
	(CR)
	PAUSE (MONITOR OFF)
	(T CR)
	A.

When the transfer of the BLOWUP file is complete, or if a no answer was originally given, the prompt:

REPLAY ORIGINAL SPEECH? (YES=1)

appears. A yes answer fills the disk file DSPOUT with the digitized speech from the DSPEECH file that resulted in the spectrogram section between the two cursors. When DSPOUT is filled, or if a no answer was given to the last prompt, the prompt:

RECONSTRUCT SPEECH FROM DFT SECTIONS (YES=1)

appears. A yes answer calls the program RECON into execution and the file DSPOUT is filled with digitized speech reconstructed from the file WINDOW through the use of the inverse DFT. If you have just filled DSPOUT with speech from the file DSPEECH, do NOT choose this option. The DSPOUT file is deleted and recreated for this routine. When DSPOUT is filled the program RECON terminates. If a no answer was originally given, the program EXAMINE terminates.

At this point if you created a DSPOUT file, you can convert the digitized speed to analog waveform by using the system program D2AOUT. D2AOUT is a modification of the system program D2A which inputs digitized speech from the file DSPOUT instead of the file DSPEECH. The operation of D2AOUT and D2A is identical. If you use D2AOUT you will be asked to input the number of bytes of information to be transfered. You can obtain this number by listing the DSPOUT file (LIST DSPOUT) before you start the D2AOUT program.

This concludes the description of the cursor routines. The video digitizer boards will be the subject of the next section.

VI. Video Digitizer Boards

The video digitizer board set consists of three boards: an A/D board, a DMA board and a D/A board. The A/D board is not used in this system but it can be used for other purposes. This section will deal primarily with the DMA and the D/A boards. These boards when used together are called the monitor interface boards. This section will begin with a brief look at the set up of the boards.

The boards are located in the Cromenco S-100 system. Both the A/D and the D/A boards have video cable connections. The video cable leading to the A/D board and the cable connector on the board are both tagged "11". The video cable that connects the television monitor with the D/A board and the cable connector on the board are tagged "22". The switch settings of importance will be described next.

The switches that control the vertical and horizontal resolution are located near the top of the DMA board. The four switches are labled B1 through B4, with B1 being the uppermost switch. The on position is when the switch is pushed in towards the S-100 bus connections. Tables XI and XII list the switch settings for both the vertical and horizontal resolutions. The settings differ from those in the video digitizer user's manual in that the vertical and horizontal switches have been reversed. That is, in the manual switches B1 and B2 control the horizontal resolution, while in this system switches B1 and B2 control vertical resolution. This is because the television monitor has been rotated ninety degrees.

Table XI

Vertical Resolution Switch Settings

SWITCHES B1	B2	VERTICAL RESOLUTION (LINES/FIELD)
OFF	OFF	512
ON	OFF	256
OFF	ON	128
ON	ON	64

Table XII

Horizontal Resolution Switch Settings

SWITCHES B1	B2	HORIZONTAL RESOLUTION (PIXELS/LINE)
OFF	OFF	256
ON	OFF	128
OFF	ON	85
ON	ON	64

If the display on the monitor is curled at the left edge of the screen, this can be corrected by adjusting the potentiometer located near the bottom of the DMA board.

To check the boards, the following command:

O 84 FF

can be issued when the S-100 system is in the RDOS mode. This turns the DMA board on to display the contents of memory on the television monitor. The command:

0 00 FF

will turn the DMA board off.

The video digitizer user's manual should be consulted if any problems arise. This completes the description of how to operate the spectrogram display system. This user's manual will conclude with a look at some of the problem areas to avoid when using the system.

VII. General Precautions

The most likely area for problems to arise is in making mistakes in entering the various parameters that govern the generation of the spectrogram. As previously mentioned, if you realize a mistake has been made, the best course of action is to terminate the program (CTRL-A) and start over. If a prompt calls for an integer value and you enter a real number, the number will automatically be truncated to an integer value. So you need not terminate the program for this type of error, provided the truncated value is acceptable. The same holds true if the prompt calls for a real number and you enter an integer.

The files WINDOW, INTRP, DISPLAY, and CONSTANTS, are deleted and recreated when the program TRANSFORM begins execution. Therefore, if you wish to save one of these files for later use, you must rename it prior to analyzing another sample of speech. The most likely file you will want to save is DISPLAY but there may be other times when the other files might be of some future use. The files, BLOWUP and DSPOUT, are deleted and recreated when the program EXAMINE begins execution. The same considerations as above apply here. Also, the system program A2D deletes and recreates the file DSPEECH when digitizing a new sample of speech. Care should be exercised in using these programs so that a disk file that you would like to save for future use is not deleted when analyzing a new sample of speech.

The programs on the NOVA are designed to operate in sequence. This sequence normally begins with the program TRANSFORM, however, you need not always start with TRANSFORM to use the other programs. PREPARE, SHAPE and EXAMINE can be started on their own, provided the required disk files exist. Table XIII lists the programs, the required disk files and the program called upon completion. The only program that should not be used on its own is RECON.

Table XIII

Required Disk Files

PROGRAM	REQUIRED DISK FILES	PROGRAM CALLED
TRANSFORM	DSPEECH	PREPARE
PREPARE	CONSTANTS WINDOW	SHAPE
SHAPE	CONSTANTS INTRP	EXAMINE (OPTIONAL)
EXAMINE	CONSTANTS DISPLAY DSPEECH WINDOW	RECON (OPTIONAL)

The transfer of data, both display files and cursor choices, from the NOVA to the S-100 system is also a potential problem area. If the S-100 system starts looking for the data before the NOVA is ready to transfer it, the S-100 may accept some extraneous values. When this occurs, the television monitor will be turned on (indicating it has received all the data it expects) but the program on the

NOVA will still be waiting to transfer more data. This problem can be avoided if the carriage return on the S-100 terminal that starts the transfer process is not entered until the prompt:

********** READY **********

appears on the NOVA terminal. If the problem does occur, the program on the NOVA will have to be terminated and restarted.

This concludes the discussion of the problem areas to be avoided. In summary, care should be used in entering values into the NOVA, disk files that need to be saved should be renamed, and the transfer of data from the NOVA to the S-100 should not be started until the NOVA is ready. In addition to the above problems, the NOVA system itself can produce errors when writing to, reading from, opening, or closing a disk file. Such an error will produce a prompt that will tell you what kind of error took place and the fortran error code. The error codes are listed in the appendix of the Fortran IV user's manual. Using the information listed in the manual should allow you to discover the problem and correct it. The most common error of this type occurs when a program attempts to write to a disk file and there is no room left on the disk. You will then have to clear some files from the disk and restart the program.

In spite of all the information just presented, you should have little trouble in using the system. Provided, of course, that you observe all the necessary precautions. As with most systems, the more you use it, the easier it becomes to use and understand.

Appendix B

NOVA Programs

```
C       TRANSFORM
C-----------------------------------------------------------
C
C            LIST OF VARIABLES
C
C-----------------------------------------------------------
C
C
C JR - NUMBER OF INPUT SAMPLES
C JR1 - JR MINUS ONE
C N - NUMBER OF FREQUENCY SAMPLES DESIRED
C N1 - N MINUS ONE
C N21 - N DIVIDED BY TWO, MINUS ONE
C N211 - N21 MINUS ONE
C IHOZ - HORIZONTAL RESOLUTION
C IHOZ1 - IHOZ MINUS ONE
C IVER - VERTICAL RESOLUTION
C IVER1 - IVER MINUS ONE
C ICURS1 - CURSOR POSITION ONE
C ICURS2 - CURSOR POSITION TWO
C PER - PERCENT OVERLAP DESIRED
C PC - PER TIMES N
C M1 - NUMBER OF SAMPLES OVERLAPPED (PC TRUNCATED)
C M - N MINUS M1
C R - (JR MINUS N1 PLUS M) DIVIDED BY M
C IR - NUMBER OF TIME SECTIONS (R TRUNCATED)
C IR1 - IR MINUS ONE
C IW - WINDOW CHOICE
C     1.  RECTANGULAR
C     2.  HANNING (COSINE SQUARED)
C     3.  HAMMING
C     4.  4-SAMPLE KAISER-BESSEL
C PI - 3.14.....
C ARG - (TWO TIMES PI) DIVIDED BY N1
C ARG2 - TWO TIMES ARG
C ARG3 - THREE TIMES ARG
C IRECT - DATA ARRAY (N,BANDWIDTH)
C IHANN - DATA ARRAY (N,BANDWIDTH)
C IHAMM - DATA ARRAY (N,BANDWIDTH)
C IKAIS - DATA ARRAY (N,BANDWIDTH)
C INPUT - ARRAY CONTAINING INPUT SPEECH DATA
C INPUT1 - ARRAY USED IN  INPUTING SPEECH DATA
C OUTPUT - ARRAY CONTAINING THE DATA AFTER THE FFT
C IREC - RECORD LENGTH FOR 'WINDOW' FILE
C NBLK - NUMBER OF BLOCKS OF DATA TO BE INPUT
```

```
C ISTBLK - STARTING BLOCK OF INPUT SPEECH
C THRNOISE - NOISE THRESHOLD
C IWOB - WHITE ON BLACK DISPLAY DESIRED?
C LININT - LINEAR INTERPOLATION DESIRED?
C
C-----------------------------------------------------------
C
C       DECLARATION STATEMENTS
C
C-----------------------------------------------------------
C
        DIMENSION OUTPUT(0:1023),INPUT(0:1023)
        DIMENSION INPUT1(0:1279)
        COMPLEX OUTPUT
        COMMON /HERE/IRECT(7,2),IHANN(7,2),IHAMM(7,2),
     *  IKAIS(7,2)
        DATA IRECT/16,32,64,128,256,512,1024,500,250,125,
     *  63,31,16,8/
     *  IHANN/16,32,64,128,256,512,1024,750,375,188,
     *  94,47,23,12/
     *  IHAMM/16,32,64,128,256,512,1024,680,340,170,
     *  85,43,21,11/
     *  IKAIS/16,32,64,128,256,512,1024,900,450,225,
     *  113,56,28,14/
C
C       OPEN FILES.
C
        CALL DFILW("WINDOW",IER)
        IF ((IER.NE.1).AND.(IER.NE.13)) GO TO 500
        CALL DFILW("INTRP",IER)
        IF ((IER.NE.1).AND.(IER.NE.13)) GO TO 500
        CALL DFILW("DISPLAY",IER)
        IF ((IER.NE.1).AND.(IER.NE.13)) GO TO 500
        CALL DFILW("CONSTANTS",IER)
        IF ((IER.NE.1).AND.(IER.NE.13)) GO TO 500
        CALL CFILW("INTRP",1,IER)
        IF (IER.NE.1) GO TO 520
        CALL CFILW("DISPLAY",2,IER)
        IF (IER.NE.1) GO TO 520
        CALL CFILW("CONSTANTS",1,IER)
        IF (IER.NE.1) GO TO 520
        CALL OPEN(4,"DSPEECH",2,IER,512)
        IF (IER.NE.1) GO TO 540
        CALL OPEN(5,"CONSTANTS",2,IER)
        IF (IER.NE.1) GO TO 540
        GO TO 600
C
C       PRINT ERROR MESSAGE AND GO TO STOP
C
 500    WRITE (10,510) IER
 510    FORMAT (" FILE DELETING ERROR CODE - ",I3)
        GO TO 1390
 520    WRITE (10,530) IER
 530    FORMAT (" FILE CREATING ERROR CODE - ",I3)
```

```
      GO TO 1390
 540  WRITE (10,550) IER
 550  FORMAT (" FILE OPENING ERROR CODE - ",I3)
      GO TO 1390
C
C-----------------------------------------------------
C
C     PICK A WINDOW,N AND OVERLAP ROUTINE
C
C-----------------------------------------------------
C
C     DUMMY VARIABLES:
C            I,J
C
 600  ACCEPT "PICK A WINDOW <15>","ENTER 1 FOR A",
     1 " RECTANGULAR WINDOW <15>","ENTER 2 FOR A HANNING",
     2 " WINDOW <15>","ENTER 3 FOR A HAMMING WINDOW <15>",
     3 "ENTER 4 FOR A 4-SAMPLE KAISER-BESSEL WINDOW",
     4 "<15>",IW
      IF (IW.EQ.4) GO TO 680
      IF (IW.EQ.3) GO TO 650
      IF (IW.EQ.2) GO TO 620
C
C     A RECTANGULAR WINDOW HAS BEEN CHOOSEN.  THE
C     DIFFERENT CHOICES FOR N WILL BE DISPLAYED ALONG
C     WITH THE RESULTING HALF POWER BANDWIDTHS.
C
      TYPE "<15>","     RECTANGULAR WINDOW <15>"
      TYPE "       N   BANDWIDTH <15>"
      DO 610 I=1,7
      TYPE (IRECT(I,J),J=1,2)
 610  CONTINUE
      GO TO 700
C
C     A HANNING WINDOW HAS BEEN CHOOSEN.  THE DIFFERENT
C  CHOICES FOR N WILL BE DISPLAYED ALONG WITH THE RESULTING
C  HALF POWER BANDWIDTHS.
C
 620  TYPE "<15>","     HANNING WINDOW <15>"
      TYPE "       N   BANDWIDTH <15>"
      DO 630 I=1,7
      TYPE (IHANN(I,J),J=1,2)
 630  CONTINUE
      GO TO 700
C
C     A HAMMING WINDOW HAS BEEN CHOOSEN.  THE DIFFERENT
C  CHOICES FOR N WILL BE DISPLAYED ALONG WITH THE RESULTING
C  HALF POWER BANDWIDTHS.
C
 650  TYPE "<15>","     HAMMING WINDOW <15>"
      TYPE "       N   BANDWIDTH <15>"
      DO 660 I=1,7
      TYPE (IHAMM(I,J),J=1,2)
 660  CONTINUE
```

```
      GO TO 700
C
C     A 4-SAMPLE KAISER-BESSEL WINDOW HAS BEEN CHOOSEN.
C     THE DIFFERENT CHOICES FOR N WILL BE DISPLAYED ALONG
C     WITH THE RESULTING HALF POWER BANDWIDTHS.
C
 680  TYPE "<15>","  4-SAMPLE KAISER-BESSEL WINDOW<15>"
      TYPE "      N   BANDWIDTH <15>"
      DO 690 I=1,7
      TYPE (IKAIS(I,J),J=1,2)
 690  CONTINUE
C
C     PICK VALUES FOR N, THE DESIRED AMOUNT OF OVERLAP AND
C  THE NUMBER OF INPUT SAMPLES.
C
 700  ACCEPT "<15> PICK A VALUE FOR N <15>",N
 705  ACCEPT "<15> PICK PERCENT OVERLAP (REAL) <15>",PER
      IF (PER.LT.1.0) GO TO 710
      TYPE "<15>","PERCENT OVERLAP MUST BE LESS THAN
     * 100 PER CENT"
      GO TO 705
 710  PC=PER*N
      M1=PC
      TYPE "<15>"," ENTER NUMBER OF INPUT SAMPLES
     * (23040 IF"
      ACCEPT " INPUT THRU S-100)<15>",JR
      TYPE "<15>"," VERTICAL RESOLUTION CHOICES: 512,
     * 256,128 OR 64"
      ACCEPT " PICK VERTICAL RESOLUTION<15>",IVER
      IF (IVER.LE.128) GO TO 720
      IF (IVER.EQ.256) GO TO 715
      TYPE "<15>"," HORIZONTAL RESOLUTION CHOICE: 64"
      GO TO 725
 715  TYPE "<15>"," HORIZONTAL RESOLUTION CHOICES: 128
     * OR 64"
      GO TO 725
 720  TYPE "<15>"," HORIZONTAL RESOLUTION CHOICES: 256,128
     * OR 64"
 725  ACCEPT " PICK HORIZONTAL RESOLUTION<15>",IHOZ
      ACCEPT "<15> PICK STARTING BLOCK OF INPUT SPEECH
     * <15>",ISTBLK
      THRNOISE=10000
      TYPE " NOISE THRESHOLD IS 10000"
      ACCEPT " TO CHANGE THE THRESHOLD ENTER 1 - ",I
      IF (I.EQ.1) ACCEPT " NOISE THRESHOLD (REAL) = ",
     * THRNOISE
      ACCEPT " WHITE ON BLACK DISPLAY DESIRED? (YES=1) ",
     * IWOB
      LININT=0
      IF (N.GE.(IVER*2)) GO TO 730
      ACCEPT " LINEAR INTERPOLATION DESIRED? (YES=1) ",
     * LININT
```

```
C
C-----------------------------------------------------------
C
C       INITIALIZE VARIABLES
C
C-----------------------------------------------------------
C
 730    JR=JR-(ISTBLK*256)
        JR1=JR-1
        N1=N-1
        N21=(N/2)-1
        N211=N21-1
        IHOZ1=IHOZ-1
        IVER1=IVER-1
        M=N-M1
        R=(JR-N1+M)/M
        IR=R
        IR1=IR-1
        IF (IR1.GT.IHOZ1) IR1=IHOZ1
        NBLK=(N/256)+1
        IF (N.LT.256) NBLK=2
        PI=3.1415925
        ARG=(2.0*PI)/N1
        ARG2=2.0*ARG
        ARG3=3.0*ARG
        IREC=8*N
C
C       OPEN 'WINDOW' FILE AND SAVE CONSTANTS.
C
        CALL FOPEN(1,"WINDOW",IREC)
        WRITE BINARY (5) ISTBLK,THRNOISE,N,N1,N21,N211,IR,
     *  IR1,IREC,IVER,IVER1,IHOZ,IHOZ1,M,IWOB,LININT
C
C-----------------------------------------------------------
C
C       INPUT SPEECH
C
C-----------------------------------------------------------
C
C       DUMMY VARIABLES:
C               AJ,AJ1,AL,AL1,I,J,J1,L1,L2
C
        CALL FGTIM(IHR,IMIN,ISEC)
        WRITE (10,800) (IHR,IMIN,ISEC)
 800    FORMAT(" INPUT START TIME = ",I2,2X,I2,2X,I2)
        DO 1370 J=0,IR1
C
C       AL IS THE STARTING ADDRESS FOR EACH SECTION.
C
        AL=M*J
C
C       INIALIZE J1 WHICH IDENTIFIES THE FIRST BLOCK TO BE
C  READ.
C
```

```
        J1=ISTBLK
C
C       DETERMINE THE FIRST BLOCK
C
           DO 810 I=0,N1
           AJ=256.*I
           IF (AL.GT.AJ) J1=J1+1
 810       CONTINUE
C
C       READ IN NBLK BLOCKS OF SPEECH STARTING AT THE J1
C  BLOCK.
C
        CALL RDBLK(4,J1,INPUT1,NBLK,IER)
        IF ((IER.EQ.1).OR.(IER.EQ.9)) GO TO 820
C
C       PRINT ERROR MESSAGE AND GO TO STOP.
C
        WRITE(10,815) IER
 815    FORMAT(" RDBLK ERROR CODE - ",I3)
        GO TO 1390
C
C       FILL INPUT ARRAY WITH THE PROPER VALUES FROM THE
C  INPUT1 ARRAY.  L1 IS THE SLIDING INDEX ADJUSTED BY THE
C  STARTING BLOCK.
C
 820       AJ1=(J1-(ISTBLK+1))*256.
           DO 830 I=0,N1
           AL1=AL+I-AJ1
           L1=IFIX(AL1)
           INPUT(I)=INPUT1(L1)
 830       CONTINUE
           IF (J.LT.IR1) GO TO 850
C
C       J EQUALS IR1.  CHECK IF NUMBER OF INPUT SAMPLES
C  EXCEEDS JR.  IF SO FILL INPUT WITH ZEROS.
C
           DO 840 I=0,N1
           L2=N1+M*(IR1-1)+I
           IF (L2.GT.JR1) INPUT(I)=0
 840       CONTINUE
C
C       GO TO WINDOW ROUTINE ALREADY CHOOSEN.
C
C 4-SAMPLE KAISER-BESSEL?
 850    IF (IW.EQ.4) GO TO 1300
C HAMMING?
        IF (IW.EQ.3) GO TO 1200
C HANNING?
        IF (IW.EQ.2) GO TO 1100
C RECTANGULAR ROUTINE FOLLOWS
```

```
C
C-----------------------------------------------------------------
C
C       WINDOWING AND DFT ROUTINES
C
C-----------------------------------------------------------------
C
C       DUMMY VARIABLES:
C               I,J,L
C
C
C       RECTANGULAR WINDOW
C
           DO 1050 I=0,N1
           OUTPUT(I)=INPUT(I)
 1050      CONTINUE
C
C       THE WINDOW IS NOW COMPLETE.  THE DFT IS TAKEN AND
C  THE PROGRAM JUMPS TO THE OUTPUT STAGE.
C
        CALL DFT4(OUTPUT(0),N,0)
        GO TO 1360
C
C       HANNING WINDOW
C
 1100      DO 1150 I=0,N1
           OUTPUT(I)=0.5*(1.0-COS(I*ARG))*(INPUT(I))
 1150      CONTINUE
C
C       THE WINDOW IS NOW COMPLETE.  THE DFT IS TAKEN AND
C  THE PROGRAM JUMPS TO THE OUTPUT STAGE.
C
        CALL DFT4(OUTPUT(0),N,0)
        GO TO 1360
C
C       HAMMING WINDOW
C
 1200      DO 1250 I=0,N1
           OUTPUT(I)=(0.54-0.46*COS(I*ARG))*(INPUT(I))
 1250      CONTINUE
C
C       THE WINDOW IS NOW COMPLETE.  THE DFT IS TAKEN AND
C  THE PROGRAM JUMPS TO THE OUTPUT STAGE.
C
        CALL DFT4(OUTPUT(0),N,0)
        GO TO 1360
C
C       4-SAMPLE KAISER-BESSEL
C
 1300      DO 1350 I=0,N1
           OUTPUT(I)=(0.40243-0.49804*COS(I*ARG)+0.09831
     *     *COS(I*ARG2)-0.00122*COS(I*ARG3))*(INPUT(I))
 1350      CONTINUE
C
```

```
C       THE WINDOW IS NOW COMPLETE.  THE DFT IS TAKEN AND
C  PROGRAM CONTROL GOES TO THE OUTPUT STAGE.
C
        CALL DFT4(OUTPUT(0),N,0)
C
C       STORE OUTPUT IN 'WINDOW' FILE AND RETURN FOR THE
C  NEXT INPUT SECTION.
C
 1360   WRITE BINARY (1) (OUTPUT(I),I=0,N1)
 1370   CONTINUE
        CALL FGTIM(IHR,IMIN,ISEC)
        WRITE(10,1380) IHR,IMIN,ISEC
 1380   FORMAT(" TRANSFORM FINISH TIME = ",I2,2X,I2,2X,I2)
C
C       CLOSE FILES.
C
        CALL RESET
C
C       TURN PROGRAM CONTROL OVER TO PREPARE.
C
        CALL CHAIN("PREPARE.SV",IER)
 1390   STOP
        END
```

```
C       PREPARE
C-----------------------------------------------------------
C
C            LIST OF VARIABLES
C
C-----------------------------------------------------------
C
C
C JR - NUMBER OF INPUT SAMPLES
C JR1 - JR MINUS ONE
C N - NUMBER OF FREQUENCY SAMPLES DESIRED
C N1 - N MINUS ONE
C N21 - N DIVIDED BY TWO, MINUS ONE
C N211 - N21 MINUS ONE
C IHOZ - HORIZONTAL RESOLUTION
C IHOZ1 - IHOZ MINUS ONE
C IVER - VERTICAL RESOLUTION
C IVER1 - IVER MINUS ONE
C ICURS1 - CURSOR POSITION ONE
C ICURS2 - CURSOR POSITION TWO
C PER - PERCENT OVERLAP DESIRED
C PC - PER TIMES N
C M1 - NUMBER OF SAMPLES OVERLAPPED (PC TRUNCATED)
C M - N MINUS M1
C R - (JR MINUS N1 PLUS M) DIVIDED BY M
C IR - NUMBER OF TIME SECTIONS (R TRUNCATED)
C IR1 - IR MINUS ONE
C IW - WINDOW CHOICE
C       1.  RECTANGULAR
C       2.  HANNING (COSINE SQUARED)
C       3.  HAMMING
C       4.  4-SAMPLE KAISER-BESSEL
C PI - 3.14.....
C ARG - (TWO TIMES PI) DIVIDED BY N1
C ARG2 - TWO TIMES ARG
C ARG3 - THREE TIMES ARG
C OUTPUT - ARRAY CONTAINING THE DATA AFTER THE FFT
C ANORM - ARRAY CONTAINING THE DATA FROM OUTPUT NORMALIZED
C AINTRP - ARRAY CONTAINING THE DATA INTERPOLATED FROM NORM
C AMAX - ARRAY CONTAINING NORMALIZATION CONSTANTS
C ALGMAX - COMMON LOGARITHM OF AMAX
C IREC - RECORD LENGTH FOR 'WINDOW' FILE.
C ISTBLK - STARTING BLOCK FOR SPEECH DATA
C THRNOISE - NOISE THRESHOLD
C IWOB - WHITE ON BLACK DISPLAY DESIRED?
C LININT - LINEAR INTERPOLATION DESIRED?
C
C----------------------------------------------------------
C
C       DECLARATION STATEMENTS
C
C----------------------------------------------------------
C
```

```
      DIMENSION OUTPUT(0:1023),ANORM(0:511),AINTRP(0:255)
      DIMENSION AMAX(0:255)
      COMPLEX OUTPUT
C
C     OPEN FILES.
C
      CALL OPEN(2,"INTRP",2,IER)
      IF (IER.NE.1) GO TO 1420
      CALL OPEN(5,"CONSTANTS",2,IER)
      IF (IER.NE.1) GO TO 1420
      REWIND 5
      READ BINARY (5) ISTBLK,THRNOISE,N,N1,N21,N211,IR,
     * IR1,IREC,IVER,IVER1,IHOZ,IHOZ1,M,IWOB,LININT
C
C     OPEN 'WINDOW' FILE.
C
      CALL OPEN(1,"WINDOW",2,IER,IREC)
      IF (IER.NE.1) GO TO 1420
      GO TO 1430
C
C     PRINT ERROR MESSAGE AND STOP.
C
 1420 WRITE (10,1425) IER
 1425 FORMAT (" FILE OPENING ERROR CODE - ",I3)
      GO TO 1635
C
C------------------------------------------------------------
C
C     NORMALIZATION ROUTINE
C
C------------------------------------------------------------
C
C     DUMMY VARIABLES:
C             I,J,L
C
 1430 CALL FGTIM(IHR,IMIN,ISEC)
      WRITE(10,1435) IHR,IMIN,ISEC
 1435 FORMAT (" PREPARE START TIME = ",I2,2X,I2,2X,I2)
      REWIND 1
      DO 1620 J=0,IR1
      READ BINARY (1) (OUTPUT(L),L=0,N1)
C
C     SET AMAX.
C
      AMAX(J)=CABS(OUTPUT(0))
C
C     FIND AMAX AND FILL ANORM.
C
         DO 1490 I=0,N21
         ANORM(I)=CABS(OUTPUT(I))
         IF (ANORM(I).GT.AMAX(J)) AMAX(J)=ANORM(I)
 1490    CONTINUE
C
C     USE AMAX TO NORMALIZE ANORM AND TAKE THE COMMON
```

```
C  LOGARITHM OF ANORM.
C
         IF (AMAX(J).LT.THRNOISE) AMAX(J)=THRNOISE*100.0
         IF (AMAX(J).LE.0.) AMAX(J)=1.0
         ALGMAX=ALOG10(AMAX(J))
            DO 1500 I=0,N21
            IF (ANORM(I).LE.0.0) ANORM(I)=0.1**99
            ANORM(I)=20.*(ALOG10(ANORM(I))-ALGMAX)
 1500       CONTINUE
C
C--------------------------------------------------------------
C
C        INTERPOLATION ROUTINE
C
C--------------------------------------------------------------
C
C        DUMMY VARIABLES:
C        .      I,J,I1,K,L,K1,L1,L2
C
C        CHECK N AND JUMP ACCORDINGLY.
C
         IF (N.GE.(IVER*4)) GO TO 1560
         IF (N.EQ.(IVER*2)) GO TO 1590 .
C
C        N IS LESS THAN (IVER*2) AND THEREFORE THE ARRAY MUST
C  BE EXPANDED.  L IS THE NUMBER OF TIMES THE ARRAY MUST BE
C  EXPANDED.
C
         L=IVER/(N21+1)
         IF (LININT.NE.1) GO TO 1545
C
C        EXPAND ARRAY BY LINEAR INTERPOLATION.
C
            DO 1540 I=0,N21
C
C        SET I1 AND K
C
            I1=1+I
            K=I*L
            AINTRP(K)=ANORM(I)
C
C        WHEN I EQUALS N21 THERE IS NO I1 TERM SO A
C  SPECIAL ROUTINE MUST HANDLE IT.
C
            IF (I.EQ.N21) GO TO 1520
C
C        SET K1 AND INCREMENT K.
C
            K1=1
 1510       K=K+1
C
C        IF K1 EQUALS L, RETURN TO GET ANOTHER ANORM TERM.
C
            IF (K1.EQ.L) GO TO 1540
```

```
C
C       AINTRP EQUALS THE PRESENT ANORM TERM PLUS THE
C  DIFFERENCE BETWEEN THE PRESENT TERM AND THE NEXT TERM
C  MULTIPLIED BY THE FRACTION K1 OVER L.
C
          AINTRP(K)=ANORM(I)+K1*(ANORM(I1)-ANORM(I))/L
C
C       INCREMENT K1 AND RETURN TO DETERMINE THE NEXT AINTRP
C  TERM.
C
          K1=K1+1
          GO TO 1510
C
C       THIS SECTION OF THE ROUTINE IS FOR THE SPECIAL CASE
C  WHEN I EQUALS N21.  THIS SECTION IS THE SAME AS THE
C  PREVIOUS SECTION EXCEPT THE LAST TWO TERMS OF ANORM
C  ARE USED.
C
 1520     K1=1
 1530     K=K+1
          IF (K1.EQ.L) GO TO 1540
          AINTRP(K)=ANORM(N21)+K1*(ANORM(N21)-
     *    ANORM(N211))/L
          K1=K1+1
          GO TO 1530
 1540     CONTINUE
C
C       INTERPOLATION IS NOW COMPLETE.  GO ON TO STORE
C  AINTRP.
C
        GO TO 1610
C
C       EXPAND ARRAY BY REPEATING ANORM TERMS.
C
 1545     DO 1555 I=0,N21
          K=I*L
          K1=K+L
             DO 1550 I1=K,K1
             AINTRP(I1)=ANORM(I)
 1550        CONTINUE
 1555     CONTINUE
C
C       EXPANSION IS NOW COMPLETE. GO ON TO STORE AINTRP.
C
        GO TO 1610
C
C       N IS GREATER THAN (OR EQUAL TO) IVER*4 AND THEREFORE
C  MUST BE CONTRACTED TO MEET ARRAY SIZE REQUIREMENTS
C
 1560   L=(N21+1)/IVER
          DO 1570 I=0,IVER1
          L1=I*L
          AINTRP(I)=0
             DO 1565 L2=1,L
```

```
              AINTRP(I)=AINTRP(I)+(ANORM(L1)/L)
              L1=L1+1
 1565         CONTINUE
 1570      CONTINUE
C
C       INTERPOLATION IS NOW COMPLETE. GO ON TO STORE
C  AINTRP.
C
        GO TO 1610
C
C       N EQUALS IVER*2 SO NO INTERPOLATION IS NECESSARY.
C  ANORM IS STORED DIRECTLY AS AINTRP.
C
 1590   WRITE BINARY (2) (ANORM(I),I=0,IVER1)
        GO TO 1620
 1610   WRITE BINARY (2) (AINTRP(I),I=0,IVER1)
 1620   CONTINUE
        WRITE (5,1625) (AMAX(I),I=0,IR1)
 1625   FORMAT (4(2X,E14.7))
        CALL FGTIM(IHR,IMIN,ISEC)
        WRITE(10,1630) IHR,IMIN,ISEC
 1630   FORMAT(" PREPARE STOP TIME = ",I2,2X,I2,2X,I2)
C
C       CLOSE ALL FILES.
C
        CALL RESET
C
C       TURN PROGRAM CONTROL OVER TO SHAPE.
C
        CALL CHAIN("SHAPE.SV",IER)
 1635   CALL RESET
        STOP
        END
```

```
C       SHAPE
C-------------------------------------------------------------
C
C              LIST OF VARIABLES
C
C-------------------------------------------------------------
C
C
C JR - NUMBER OF INPUT SAMPLES
C JR1 - JR MINUS ONE
C N - NUMBER OF FREQUENCY SAMPLES DESIRED
C N1 - N MINUS ONE
C N21 - N DIVIDED BY TWO, MINUS ONE
C N211 - N21 MINUS ONE
C IHOZ - HORIZONTAL RESOLUTION
C IHOZ1 - IHOZ MINUS ONE
C IVER - VERTICAL RESOLUTION
C IVER1 - IVER MINUS ONE
C PER - PERCENT OVERLAP DESIRED
C PC - PER TIMES N
C M1 - NUMBER OF SAMPLES OVERLAPPED (PC TRUNCATED)
C M - N MINUS M1
C R - (JR MINUS N1 PLUS M) DIVIDED BY M
C IR - NUMBER OF TIME SECTIONS (R TRUNCATED)
C IR1 - IR MINUS ONE
C AINTRP - ARRAY CONTAINING THE DATA INTERPOLATED FROM NORM
C IDISPLAY - ARRAY CONTAINING FINAL SPECTROGRAM
C IPACKED - ARRAY CONTAINING PACKED DISPLAY
C ISGRM - ARRAY CONTAINING 4 IPACKED ARRAYS.
C THR - THRESHOLD VALUE IN DB'S
C STP - STEP SIZE IN DB'S
C ISE - SPECTRAL EMPHASIS DESIRED?
C ISS - CHANGE SPECTRAL SHAPING?
C ISUP - NUMBER OF LEVELS TO BE SUPPRESED
C ICOMP - COMPARISON MARK FOR LEVEL SETTING
C DBOC - DB'S PER OCTAVE (SPECTRAL EMPHASIS)
C SFREQ - STARTING FREQUENCY (SPECTRAL EMPHASIS)
C NPACK - THE NUMBER OF ARRAYS THAT CAN BE PACKED INTO
C         ONE BLOCK
C NTOT - NUMBER OF INTEGERS TO BE TRANSFERED.
C ISTOP - NUMBER OF BLOCKS TO BE TRANSFERED(MINUS ONE).
C IWOB - WHITE ON BLACK DISPLAY DESIRED?
C IPROG - CALL PROGRAM EXAMINE.SV?
C
C-------------------------------------------------------------
C
C       DECLARATION STATEMENTS
C
C-------------------------------------------------------------
C
        DIMENSION AINTRP(0:255),IDISPLAY(0:255),IPASS(256)
        DIMENSION ISGRM(0:255),IPACKED(0:63)
C
```

```
C        OPEN FILES.
C
        CALL OPEN(2,"INTRP",2,IER)
        IF (IER.NE.1) GO TO 1930
        CALL OPEN(3,"DISPLAY",2,IER)
        IF (IER.NE.1) GO TO 1930
        CALL OPEN(5,"CONSTANTS",2,IER)
        IF (IER.NE.1) GO TO 1930
C
C       READ IN CONSTANTS.
C
        REWIND 5
        READ BINARY (5) ISTBLK,THRNOISE,N,N1,N21,N211,IR,
     *  IR1,IREC,IVER,IVER1,IHOZ,IHOZ1,M,IWOB
        NPACK=1024/IVER
C
C-----------------------------------------------------------
C
C       SPECTROGRAM SHAPING ROUTINE
C
C-----------------------------------------------------------
C
C       DUMMY VARIABLES:
C               I,I1,I2,I3,I4,J,J2,K,K2,L,L2,M1
C
C       PICK THE THRESHOLD.
C
        TYPE "<7>"
 1710   ACCEPT "PICK THE THRESHOLD IN DECIBELS (REAL)
     1         <15>",THR
C
C       PICK THE STEP SIZE.
C
        ACCEPT "PICK THE STEP SIZE IN DECIBELS (REAL)
     *  <15>",STP
C
C       PICK THE NUMBER OF LOWER LEVELS TO BE SUPPRESSED.
C
        ACCEPT "ENTER THE NUMBER OF LOWER LEVELS TO BE ",
     1  "SUPRESSED (INTEGER) <15>",ISUP
C
C       SPECTRAL EMPHASIS DESIRED?
C
        ACCEPT "IF SPECTRAL EMPHASIS IS DESIRED ENTER
     *  1<15>",ISE
        IF (ISE.NE.1) GO TO 1715
C
C       SPECTRAL EMPHASIS IS DESIRED.  ENTER DB'S PER
C   OCTAVE AND STARTING FREQUENCY.
C
        TYPE "SPECTRAL EMPHASIS IS THE INCREASE OF SO MANY
     *  DB'S PER "
        TYPE "OCTAVE FROM A STARTING FREQUENCY<15>"
        ACCEPT "ENTER DB'S PER OCTAVE (REAL)<15>",DBOC
```

```
      ACCEPT "ENTER STARTING FREQUENCY (REAL)<15>",SFREQ
C
C     REWIND INTERPOLATED SPECTROGRAM FILE.
C
      REWIND 2
 1715 CALL FGTIM(IHR,IMIN,ISEC)
      WRITE(10,1718) IHR,IMIN,ISEC
 1718 FORMAT(" DISPLAY START TIME = ",I2,2X,I2,2X,I2)
C
C     INITIALIZE K2 AND J2 FOR USE IN FILLING ISGRM WITH
C IPACKED.
C
      K2=0
      J2=0
      DO 1860 J=0,IR1
C
C     READ IN INTERPOLATED SPECTROGRAM DATA.
C
      READ BINARY (2) (AINTRP(I),I=0,IVER1)
C
C     INITIALIZE L,M1,IPACKED(0) AND IFLAG FOR USE IN
C FILLING IPACKED WITH IDISPLAY.
C
      L=1
      M1=0
      IPACKED(0)=0
      IFLAG=0
C
C     SET I4 FOR USE IN ADDING SPECTRAL EMPHASIS
C
      I4=IFIX((SFREQ/8000.0)*N)
C
C     ASSIGN LEVELS FOR DISPLAY.
C
        DO 1830 I=0,IVER1
        IF (I.LT.I4) GO TO 1725
        IF (ISE.NE.1) GO TO 1725
C
C     ADD SPECTRAL EMPHASIS
C
        R1=(FLOAT(I))/I4
        AINTRP(I)=AINTRP(I)+DBOC*(ALOG10(R1)/ALOG10(2.))
C
C     INITIALIZE K AND DETERMINE COMP.
C
 1725   K=0
 1730   COMP=THR+K*STP
C
C     IF AINTRP IS GREATER THAN COMP THEN JUMP TO 1750
C AND DONT ASSIGN A LEVEL TO AINTRP.
C
        IF (AINTRP(I).GT.COMP) GO TO 1750
C
C     IF K IS LESS THAN ISUP AINTRP MUST BE ASSIGNED A
```

```
C  ZERO LEVEL.
C
            IF (K.LT.ISUP) GO TO 1740
C
C       ASSIGN THE PROPER LEVEL.
C
            IDISPLAY(I)=K
C
C       PROPER LEVEL HAS BEEN ASSIGNED THEREFORE RETURN TO
C  GET ANOTHER AINTRP TERM.
C
            GO TO 1755
C
C       ASSIGN A ZERO LEVEL AND RETURN FOR NEXT AINTRP TERM.
C
 1740       IDISPLAY(I)=0
            GO TO 1755
C
C       INCREMENT K AND IF K IS LESS THAN 15 RETURN TO USE
C  NEW COMPARISON MARK TO SET THE PROPER LEVEL.
C
 1750       K=K+1
            IF (K.LT.15) GO TO 1730
C
C       K EQUALS 15.  ASSIGN A LEVEL OF 15 AND RETURN TO GET
C  NEW AINTRP TERM.
C
            IDISPLAY(I)=15
C
C       DISPLAY LEVEL HAS BEEN SET.  IF L IS GREATER THAN
C  4 THEN M IS INCREMENTED, L RESET TO 1, IPACKED CLEARED
C  AND IFLAG CLEARED.
C
 1755       IF(L.LT.5) GO TO 1785
            M1=M1+1
            IPACKED(M1)=0
            L=1
            IFLAG=0
 1785       IF (L.NE.1) GO TO 1790
C
C       L=1.  SET IFLAG IF IDISPLAY = 8.
C       IDISPLAY < 8 ... IPACKED POSITIVE
C       IDISPLAY > 8 ... IPACKED NEGATIVE
C         IDISPLAY = 8 ... IPACKED = -32768 (ONE WILL BE
C  SUBTRACTED LATER)
C
            IF (IDISPLAY(I).EQ.8) IFLAG=1
            IF (IDISPLAY(I).LT.8) IPACKED(M1)=IDISPLAY(I)*
     *      4096
            IF (IDISPLAY(I).GT.8) IPACKED(M1)=(16-
     *      IDISPLAY(I))*(-4096)
            IF (IDISPLAY(I).EQ.8) IPACKED(M1)=-32767
 1790       IF (L.NE.2) GO TO 1795
C
```

```
C       L=2.
C       ADD POSITIVE NUMBER.
C       CHECK FLAG.  IF SET SUBTRACT ONE AND CLEAR FLAG
C  EXCEPT IF IDISPLAY = 0.
C
              IPACKED(M1)=IPACKED(M1)+IDISPLAY(I)*256
              IF (IDISPLAY(I).EQ.0) GO TO 1795
              IF (IFLAG.EQ.1) IPACKED(M1)=IPACKED(M1)-1
              IFLAG=0
 1795         IF (L.NE.3) GO TO 1800
C
C       L=3.
C         ADD POSITIVE NUMBER.
C       CHECK FLAG.  IF SET SUBTRACT ONE AND CLEAR FLAG
C  EXCEPT IF IDISPLAY = 0.
C
              IPACKED(M1)=IPACKED(M1)+IDISPLAY(I)*16
              IF (IDISPLAY(I).EQ.0) GO TO 1800
              IF (IFLAG.EQ.1) IPACKED(M1)=IPACKED(M1)-1
              IFLAG=0
 1800         IF (L.NE.4) GO TO 1810
C
C       L=4.
C       ADD POSITIVE NUMBER.
C       CHECK FLAG.  IF SET SUBTRACT ONE AND CLEAR FLAG
C  EXCEPT IF IDISPLAY = 0.
C
              IPACKED(M1)=IPACKED(M1)+IDISPLAY(I)
              IF (IDISPLAY(I).EQ.0) GO TO 1805
              IF (IFLAG.EQ.1) IPACKED(M1)=IPACKED(M1)-1
              IFLAG=0
C
C       CHECK FLAG.  IF SET NUMBER TO BE STORED IS 8000H.
C  CLEAR IPACKED AND SET MSB.
C
 1805         IF (IFLAG.EQ.0) GO TO 1810
              IPACKED(M1)=0
              CALL BSET(IPACKED(M1),15)
C
C       INCREMENT L
C
 1810         L=L+1
 1830         CONTINUE
C
C       IF K2=NPACK THEN K2 IS RESET TO ZERO, THE FULL
C  ARRAY, ISGRM, IS WRITTEN OUT TO DISK AND J1 IS
C  INCREMENTED.
C
        IF(K2.LT.NPACK) GO TO 1840
        K2=0
        CALL WRBLK(3,J2,ISGRM,1,IER)
        IF (IER.NE.1) GO TO 1940
        J2=J2+1
C
```

```
C       SET INDEX I1 AND I3.
C
 1840   I1=K2*(IVER/4)
        I3=(IVER/4)-1
C
C       FILL ARRAY ISGRAM WITH IPACKED USING INDEX I2.
C
           DO 1850 L2=0,I3
           I2=I1+L2
           ISGRM(I2)=IPACKED(L2)
 1850      CONTINUE
C
C       INCREMENT K2.
C
        K2=K2+1
 1860   CONTINUE
        CALL FGTIM(IHR,IMIN,ISEC)
        WRITE(10,1863) IHR,IMIN,ISEC
 1863   FORMAT(" DISPLAY STOP TIME = "I2,2X,I2,2X,I2)
C
C       TILTING AND PACKING ROUTINE FINISHED.
C
        IF (K2.EQ.NPACK) GO TO 1870
        I1=K2*(IVER/4)
C
C       FILL LEFTOVER ISGRM WITH ZEROS USING INDEX
C  I2.  IF K2=NPACK THEN THERE IS NO LEFTOVER ISGRM.
C
           DO 1865 L2=I1,255
           ISGRM(L2)=0
 1865      CONTINUE
 1870   CALL WRBLK(3,J2,ISGRM,1,IER)
        IF (IER.NE.1) GO TO 1940
C
C       SET NTOT AND ISTOP.
C
        NTOT=IVER*(IHOZ/4)
        ISTOP=(NTOT/256)-1
C
C       IF J2 EQUALS ISTOP THEN NO MORE BLOCKS OF DATA
C  NEED TO BE STORED IN 'DISPLAY'.
C
        IF (J2.EQ.ISTOP) GO TO 1878
C
C       IF J2 IS LESS THAN ISTOP THEN BLOCKS OF ZEROS
C  NEED TO BE STORED UNTIL THERE ARE ISTOP+1 BLOCKS.
C
        J2=J2+1
        DO 1875 J=J2,ISTOP
C
C       FILL ARRAY WITH ZEROS.
C
           DO 1873 I=0,255
           ISGRM(I)=0
```

```
 1873      CONTINUE
C
C       WRITE BLOCK.
C
        CALL WRBLK(3,J,ISGRM,1,IER)
        IF (IER.NE.1) GO TO 1940
 1875   CONTINUE
C
C       READY FOR TRANSFER
C
 1878         TYPE "<7>"
        TYPE "          ************** READY **************"
        DO 1890 I=0,ISTOP
C
C       READ IN BLOCK OF INTEGERS.
C
        CALL RDBLK(3,I,IPASS,1,IER)
        IF (IER.NE.1) GO. TO 1920
        IF (IWOB.EQ.1) GO TO 1882
C
C       TRANSFER INTEGER ARRAY ONE AT A TIME VIA PASSDAT
C  FOR A BLACK ON WHITE DISPLAY.
C
           DO 1880 J=1,256
           K = IPASS(J)
           CALL PASSDAT(K)
           IF(J.EQ.1)TYPE"<33><133>A<33><133>2K<7>
     +                                   TRANSFERRING DATA"
 1880      CONTINUE
        GO TO 1890
C
C       TRANSFER INTEGER ARRAY ONE AT A TIME VIA PASCURS
C  FOR A WHITE ON BLACK DISPLAY.
C
 1882      DO 1885 J=1,256
           K=IPASS(J)
           CALL PASCURS(K)
           IF (J.EQ.1) TYPE"<33><133>A<33><133>2K<7>
     +                                   TRANSFERRING DATA"
 1885      CONTINUE
 1890   CONTINUE
C
C       8192 INTEGERS ARE EXPECTED TO BE TRANSFERED.
C  TRANSFER ZEROS UNTIL THE PROPER NUMBER HAS BEEN SENT.
C
        IF (NTOT.EQ.8192) GO TO 1910
        IF (IWOB.EQ.1) GO TO 1902
        DO 1900 I=NTOT,8191
        K=0
        CALL PASSDAT(K)
 1900   CONTINUE
        GO TO 1910
 1902   DO 1905 I=NTOT,8191
        K=0
```

```
      CALL PASCURS(K)
 1905 CONTINUE
C
C     RETURN TO RESHAPE SPECTRUM?
C
 1910 ACCEPT "TO RESHAPE THE SPECTRUM ENTER 1<15>",ISS
      IF (ISS.EQ.1) GO TO 1710
      ACCEPT "CALL PROGRAM EXAMINE.SV? (YES=1) ",IPROG
      IF (IPROG.EQ.1) GO TO 1950
      GO TO 1960
C
C     PRINT ERROR MESSAGE.
C
 1920 WRITE(10,1925) IER
 1925 FORMAT(" RDBLK ERROR CODE - ",I3)
      GO TO 1960
 1930 WRITE(10,1935) IER
 1935 FORMAT(" FILE OPENING ERROR CODE - ",I3)
      GO TO 1960
 1940 WRITE(10,1945) IER
 1945 FORMAT(" WRBLK ERROR CODE - ",I3)
      GO TO 1960
C
C     CLOSE ALL FILES.
C
 1950 CALL RESET
      CALL CHAIN("EXAMINE.SV",IER)
 1960 CALL RESET
      STOP
      END
```

```
C       EXAMINE
C-------------------------------------------------------------
C
C            LIST OF VARIABLES
C
C-------------------------------------------------------------
C
C
C JR - NUMBER OF INPUT SAMPLES
C JR1 - JR MINUS ONE
C N - NUMBER OF FREQUENCY SAMPLES DESIRED
C N1 - N MINUS ONE
C N21 - N DIVIDED BY TWO, MINUS ONE
C N211 - N21 MINUS ONE
C IHOZ - HORIZONTAL RESOLUTION
C IHOZ1 - IHOZ MINUS ONE
C IVER - VERTICAL RESOLUTION
C IVER1 - IVER MINUS ONE
C PER - PERCENT OVERLAP DESIRED
C PC - PER TIMES N
C M1 - NUMBER OF SAMPLES OVERLAPPED (PC TRUNCATED)
C M - N MINUS M1
C R - (JR MINUS N1 PLUS M) DIVIDED BY M
C IR - NUMBER OF TIME SECTIONS (R TRUNCATED)
C IR1 - IR MINUS ONE
C OUTPUT - ARRAY CONTAINING THE DATA AFTER THE FFT
C IREPLAY - ARRAY USED FOR OUTPUT AFTER INVERSE DFT
C IREC - RECORD LENGTH FOR 'WINDOW' FILE.
C ISTBLK - STARTING BLOCK FOR INPUT SPEECH
C THRNOISE - NOISE THRESHOLD
C ICURS1 - CURSOR POSITION ONE
C ICURS2 - CURSOR POSITION TWO
C IWIDTH - CURSOR WIDTH
C IPOS - CURSOR POSITION ACCEPTED?
C IBLP - BLOWUP OF DISPLAY DESIRED?
C ISKIP - NUMBER OF PIXELS TO BE SKIPPED
C IBLK - NUMBER OF BLOCKS TO BE SKIPPED
C IBLK1 - IBLK PLUS ONE
C IPOINT - NUMBER OF INTEGERS TO BE SKIPPED IN FIRST
C           BLOCK READ
C IVEC - NUMBER OF VECTORS PER BLOCK OF DISPLAY
C IREAD - NUMBER OF BLOCKS TO BE READ TO PRODUCE 64
C          VECTORS
C IREP - REPLAY ORIGINAL SPEECH?
C MSKIP - NUMBER OF INTEGERS TO BE SKIPPED
C MBLK - NUMBER OF BLOCKS TO BE SKIPPED
C MBLK1 - MBLK PLUS ONE
C MPOINT - NUMBER OF INTEGERS TO BE SKIPPED IN FIRST BLOCK
C           READ
C MREAD - NUMBER OF BLOCKS TO BE READ
C MTOT - TOTAL NUMBER OF INTEGERS TO BE READ
C MPOINT1 - NUMBER OF INTEGERS TO BE INCLUDED IN LAST BLOCK
C                 READ
```

```
C IWOB - WHITE ON BLACK DISPLAY DESIRED?
C
C-----------------------------------------------------------
C
C       DECLARATION STATEMENTS
C
C-----------------------------------------------------------
C
        DIMENSION INPUT(0:255),OUTPUT(0:1023)
        DIMENSION IREPLAY(0:1023)
        DIMENSION IDISPLAY(0:255),IBLOWUP(0:255)
        COMPLEX OUTPUT
C
C-----------------------------------------------------------
C
C       OPEN FILES AND INITIALIZE VARIABLES
C
C-----------------------------------------------------------
C
        CALL DFILW("DSPOUT",IER)
        IF ((IER.NE.1).AND.(IER.NE.13)) GO TO 900
        CALL DFILW("BLOWUP",IER)
        IF ((IER.NE.1).AND.(IER.NE.13)) GO TO 900
        CALL OPEN(2,"CONSTANTS",2,IER)
        IF (IER.NE.1) GO TO 920
        CALL FOPEN(3,"DSPOUT",512)
        CALL FOPEN(4,"BLOWUP",512)
        CALL OPEN(5,"DISPLAY",2,IER,512)
        IF (IER.NE.1) GO TO 920
        CALL OPEN(7,"DSPEECH",2,IER,512)
        IF (IER.NE.1) GO TO 920
        REWIND 2
        READ BINARY (2) ISTBLK,THRNOISE,N,N1,N21,N211,IR,
     *  IR1,IREC,IVER,IVER1,IHOZ,IHOZ1,M,IWOB
        CALL FOPEN(1,"WINDOW",IREC)
C
C-----------------------------------------------------------
C
C       CURSOR DISPLAY ROUTINE
C
C-----------------------------------------------------------
C
        ACCEPT "CURSOR WIDTH(SINGLE=1,DOUBLE=2) = ",IWIDTH
C
C       INPUT DESIRED CURSOR POSITION
C
 100    TYPE "<15>"
        WRITE (10,110) IHOZ1
 110    FORMAT (" AVAILABLE CURSOR CHOICES: 0 TO ",I3)
        ACCEPT "CURSOR POSITION ONE = ",ICURS1
        ACCEPT "CURSOR POSITION TWO = ",ICURS2
C
C       READY FOR TRANSFER
C
```

```
      TYPE "<7>","                   **********READY*********"
C
C       TRANSFER VALUES
C
        K=ICURS1
        CALL PASCURS(K)
        K=ICURS2
        CALL PASCURS(K)
        K=IVER/64
        CALL PASCURS(K)
        TYPE "<7>","                   **********DONE**********"
C
C       RETURN IF CURSOR POSITION UNACCEPTABLE
C
        ACCEPT "ACCEPT CURSOR POSITION? (YES=1) ",IPOS
        IF (IPOS.NE.1) GO TO 100
        IF (ICURS1.LT.ICURS2) GO TO 140
        I=ICURS1
        ICURS1=ICURS2
        ICURS2=I
C
C-----------------------------------------------------------
C
C       BLOWUP ROUTINE
C
C-----------------------------------------------------------
C
C       DUMMY VARIABLES:
C               I,I1,I2,J,K,L,L1,L2,L3,M1,M2
C
C       THIS ROUTINE PRODUCES A DISPLAY WITH A HORIZONTAL
C  RESOLUTION OF 64 FOLLOWING THE FIRST CURSOR.
C
 140    TYPE "BLOWUP SECTION OF SPEECH FOLLOWING THE"
        ACCEPT "   LEFTMOST CURSOR? (YES=1) ",IBLP
        IF (IBLP.NE.1) GO TO 500
C
C       INITIALIZE THE VARIABLES
C
        I1=ICURS1+1
        IF (IWIDTH.EQ.2)I1=I1+2
        I2=ICURS2-I1
        ISKIP=I1*IVER
        IBLK=ISKIP/1024
        IBLK1=IBLK+1
        IPOINT=(ISKIP-(1024*IBLK))/4
        IVEC=1024/IVER
        IREAD=64/IVEC
        L1=IBLK+IREAD
        L2=L1-1
        L3=IREAD-1
        M1=IPOINT-1
C
C       IF IPOINT EQUALS ZERO THAT MEANS THE DISPLAY CAN
```

```
C  BE TRANSFERRED DIRECTLY A BLOCK AT A TIME.
C
        IF (IPOINT.EQ.0) GO TO 310
C
C       INITIALIZE L AND K
C
        L=0
        K=0
C
C       READ IN BLOCK OF 'DISPLAY'
C
        CALL RDBLK(5,IBLK,IDISPLAY,1,IER)
        IF (IER.NE.1) GO TO 940
C
C       STORE APPROPRIATE VALUES OF IDISPLAY IN IBLOWUP
C
        DO 150 I=IPOINT,255
        IBLOWUP(L)=IDISPLAY(I)
        L=L+1
 150    CONTINUE
        DO 300 J=IBLK1,L1
C
C       READ IN NEXT BLOCK OF 'DISPLAY'
C
        CALL RDBLK(5,J,IDISPLAY,1,IER)
        IF (IER.NE.1) GO TO 940
C
C       FILL THE REST OF THE ARRAY IBLOWUP WITH THE NEW
C  IDISPLAY.
C
           DO 200 I=0,M1
           IBLOWUP(L)=IDISPLAY(I)
           L=L+1
 200       CONTINUE
C
C       WRITE OUT BLOCK OF 'BLOWUP'
C
        CALL WRBLK(4,K,IBLOWUP,1,IER)
        IF (IER.NE.1) GO TO 960
        IF (J.EQ.IREAD1) GO TO 300
C
C       CLEAR L AND INCREMENT K
C
        L=0
        K=K+1
C
C       AGAIN STORE APPROPRIATE VALUES OF IDISPLAY IN
C  IBLOWUP
C
           DO 250 I=IPOINT,255
           IBLOWUP(L)=IDISPLAY(I)
           L=L+1
 250       CONTINUE
 300    CONTINUE
```

```
      GO TO 360
C
C     CLEAR K
C
 310  K=0
      DO 350 J=IBLK,L2
C
C     READ IN BLOCK OF 'DISPLAY'
C
      CALL RDBLK(5,J,IDISPLAY,1,IER)
      IF (IER.NE.1) GO TO 940
C
C     WRITE OUT BLOCK OF 'BLOWUP'
C
      CALL WRBLK(4,K,IDISPLAY,1,IER)
      IF (IER.NE.1) GO TO 960
C
C     INCREMENT K
C
      K=K+1
 350  CONTINUE
C
C     READY FOR TRANSFER
C
 360  TYPE "<7>"
      TYPE "                  **********READY*********"
      DO 400 I=0,L3
C
C     READ IN BLOCK OF 'BLOWUP'
C
      CALL RDBLK(4,I,IBLOWUP,1,IER)
      IF (IER.NE.1) GO TO 940
      IF (IWOB.EQ.1) GO TO 380
C
C     TRANSFER BLOCK OF DATA ONE INTEGER AT A TIME
C  VIA PASSDAT FOR A BLACK ON WHITE DISPLAY.
C
         DO 370 J=0,255
         K=IBLOWUP(J)
         CALL PASSDAT(K)
         IF (J.EQ.1) TYPE "<33><133>A<33><133>2K<7>
     *                               TRANSFERRING DATA"
 370     CONTINUE
      GO TO 400
C
C     TRANSFER BLOCK OF DATA ONE INTEGER AT A TIME
C  VIA PASCURS FOR A WHITE ON BLACK DISPLAY.
C
 380     DO 390 J=0,255
         K=IBLOWUP(J)
         CALL PASCURS(K)
         IF (J.EQ.1) TYPE "<33><133>A<33><133>2K<7>
     *                                          TRANSFERRING DATA"
 390     CONTINUE
```

```
 400    CONTINUE
C
C       THE CROMENCO EXPECTS 8192 INTEGERS TO BE
C  TRANSFERRED.  TRANSFER ZEROS UNTIL 8192
C  INTEGERS HAVE BEEN PASSED.
C
        M2=8192-(IREAD*256)
        IF (IWOB.EQ.1) GO TO 460
        DO 450 I=1,M2
        K=0
        CALL PASSDAT(K)
 450    CONTINUE
        GO TO 500
 460        DO 480 I=1,M2
        K=0
        CALL PASCURS(K)
 480    CONTINUE
C
C-----------------------------------------------------------------
C
C       REPLAY ORIGINAL SPEECH ROUTINE
C
C-----------------------------------------------------------------
C
C       DUMMY VARIABLES:
C               I,I1,I2,J,K,M1,M2,M3,M4
C
 500    ACCEPT "REPLAY ORIGINAL SPEECH? (YES=1) ",IREP
        IF (IREP.NE.1) GO TO 600
C
C       INITIALIZE VARIABLES.
C
        I1=ICURS1+1
        IF (IWIDTH.EQ.2) I1=I1+2
        I2=ICURS2-I1
        SKIP=I1*M
        MBLK=SKIP/256
        MBLK1=MBLK+1
        TOT=N+((I2-1)*M)
        MPOINT=SKIP-MBLK*256
        M1=TOT-(256-MPOINT)
        MREAD=M1/256
        MPOINT1=M1-MREAD*256
        K=0
        M2=MBLK1+ISTBLK
C
C       READ IN FIRST BLOCK OF ORIGINAL SPEECH.
C
        CALL RDBLK(7,M2,INPUT,1,IER)
        IF (IER.NE.1) GO TO 940
        IF (MPOINT.EQ.0) GO TO 525
C
C       ZERO OUT SPEECH THAT OCCURS BEFORE THE FIRST
C  CURSOR.
```

```
C
        DO 520 I=0,MPOINT
        INPUT(I)=0
 520    CONTINUE
C
C       WRITE OUT INPUT TO 'DSPOUT'
C
 525    CALL WRBLK(3,K,INPUT,1,IER)
        IF (IER.NE.1) GO TO 960
C
C       INCREMENT K AND SET UP M3 AND M4
C
        K=K+1
        M3=M2+1
        M4=M3+MREAD
C
C       READ IN BLOCKS OF SPEECH FROM 'DSPEECH'
C  AND PASS THEM UNCHANGED TO 'DSPOUT'.
C
        DO 540 I=M3,M4
        CALL RDBLK(7,I,INPUT,1,IER)
        IF (IER.NE.1) GO TO 940
        CALL WRBLK(3,K,INPUT,1,IER)
        IF (IER.NE.1) GO TO 960
        K=K+1
 540    CONTINUE
        M5=M4+1
C
C       READ IN THE LAST BLOCK OF SPEECH.
C
        CALL RDBLK(7,M5,INPUT,1,IER)
        IF (IER.NE.1) GO TO 940
C
C       ZERO OUT SPEECH THAT OCCURS AFTER THE SECOND
C  CURSOR.
C
        DO 560 I=MPOINT1,255
        INPUT(I)=0
 560    CONTINUE
C
C       WRITE OUT INPUT TO 'DSPOUT'
C
        CALL WRBLK(3,K,INPUT,1,IER)
        IF (IER.NE.1) GO TO 960
C
C-----------------------------------------------------------
C
C       RECONSTRUCT SPEECH ROUTINE
C
C-----------------------------------------------------------
C
C
 600    ACCEPT "RECONSTRUCT SPEECH FROM DFT SECTIONS?
     *  (YES=1) ",IRCS
```

```
      IF (IRCS.NE.1) GO TO 1000
      WRITE BINARY (2) ICURS1,ICURS2,IWIDTH
      CALL CHAIN ("RECON.SV",IER)
      GO TO 1000
900   WRITE(10,910) IER
910   FORMAT(" FILE DELETING ERROR CODE - ",I3)
      GO TO 1000
920   WRITE(10,930) IER
930   FORMAT(" FILE OPENING ERROR CODE - ",I3)
      GO TO 1000
940   WRITE(10,950) IER
950   FORMAT(" RDBLK ERROR CODE - ",I3)
      GO TO 1000
960   WRITE(10,970) IER
970   FORMAT(" WRBLK ERROR CODE - ",I3)
1000  CALL RESET
      STOP
      END
```

```
C       RECON
C-----------------------------------------------------------
C
C            LIST OF VARIABLES
C
C-----------------------------------------------------------
C
C
C JR - NUMBER OF INPUT SAMPLES
C JR1 - JR MINUS ONE
C N - NUMBER OF FREQUENCY SAMPLES DESIRED
C N1 - N MINUS ONE
C N21 - N DIVIDED BY TWO, MINUS ONE
C N211 - N21 MINUS ONE
C IHOZ - HORIZONTAL RESOLUTION
C IHOZ1 - IHOZ MINUS ONE
C IVER - VERTICAL RESOLUTION
C IVER1 - IVER MINUS ONE
C PER - PERCENT OVERLAP DESIRED
C PC - PER TIMES N
C M1 - NUMBER OF SAMPLES OVERLAPPED (PC TRUNCATED)
C M - N MINUS M1
C R - (JR MINUS N1 PLUS M) DIVIDED BY M
C IR - NUMBER OF TIME SECTIONS (R TRUNCATED)
C IR1 - IR MINUS ONE
C OUTPUT - ARRAY CONTAINING THE DATA AFTER THE FFT
C OUTPUT2 - ARRAY CONTAINING THE DATA AFTER THE DFT
C IREPLAY - ARRAY USED FOR OUTPUT AFTER INVERSE DFT
C IREC - RECORD LENGTH FOR 'WINDOW' FILE.
C ISTBLK - STARTING BLOCK FOR INPUT SPEECH
C THRNOISE - NOISE THRESHOLD
C ICURS1 - CURSOR POSITION ONE
C ICURS2 - CURSOR POSITION TWO
C IWIDTH - CURSOR WIDTH
C IWOB - WHITE ON BLACK DISPLAY DESIRED?
C
C-----------------------------------------------------------
C
C       DECLARATION STATEMENTS
C
C-----------------------------------------------------------
C
        DIMENSION OUTPUT(0:1023),IREPLAY(0:1023)
        DIMENSION OUTPUT2(0:1023)
        COMPLEX OUTPUT,OUTPUT2
C
C-----------------------------------------------------------
C
C       OPEN FILES AND INITIALIZE VARIABLES
C
C-----------------------------------------------------------
C
        CALL DFILW("DSPOUT",IER)
```

```
      IF ((IER.NE.1).AND.(IER.NE.13)) GO TO 900
      CALL OPEN(2,"CONSTANTS",2,IER)
      IF (IER.NE.1) GO TO 920
      CALL FOPEN(3,"DSPOUT",512)
      REWIND 2
      READ BINARY (2) ISTBLK,THRNOISE,N,N1,N21,N211,IR,
     * IR1,IREC,IVER,IVER1,IHOZ,IHOZ1,M,IWOB,ICURS1,
     * ICURS2,IWIDTH
      CALL FOPEN(1,"WINDOW",IREC)
C
C-------------------------------------------------------------
C
C     RECONSTRUCT SPEECH ROUTINE
C
C-------------------------------------------------------------
C
C     DUMMY VARIABLES:
C           I,I1,I2,I3,J,K,L,M1
C
C     INITIALIZE VARIABLES
C
      I1=ICURS1+1
      IF (IWIDTH.EQ.2) I1=I1+2
      I2=ICURS2-I1
      M1=N-M
      I3=I2-2
      K=N1
      IF ((2*M).LT.N) K=(2*M)-1
C
C     REWIND 'WINDOW'
C
      REWIND 1
C
C     READ IN OUTPUT ARRAYS THAT ARE BEFORE THE FIRST
C  CURSOR AND IGNORE THEM
C
      DO 630 I=0,I1
      READ BINARY (1) (OUTPUT(0),L=0,N1)
 630  CONTINUE
C
C     READ IN FIRST SECTION OF 'WINDOW' AND TRANSFORM
C
      READ BINARY (1) (OUTPUT(0),L=0,N1)
      CALL DFT4(OUTPUT(0),N,1)
C
C     READ IN SECOND SECTION OF 'WINDOW' AND TRANSFORM
C
      READ BINARY (1) (OUTPUT2(0),L=0,N1)
      CALL DFT4(OUTPUT2(0),N,1)
C
C     IF I LESS THAN M, NORMALIZE THE FIRST SECTION
C  AND STORE REAL PART IN IREPLAY.  OTHERWISE, AVERAGE
C  THE TWO SECTIONS TOGETHER, NORNALIZE, AND STORE REAL
C  PART IN IREPLAY
```

```
C
         DO 650 I=0,K
         IF (I.GE.M) OUTPUT(I)=(OUTPUT(I)+OUTPUT2(I-M))/(2*N)
         IF (I.LT.M) OUTPUT(I)=(OUTPUT(I))/N
         IREPLAY(I)=REAL(OUTPUT(I))
 650     CONTINUE
C
C        STORE IREPLAY ON DISK.
C
         WRITE BINARY (3) (IREPLAY(L),L=0,K)
         DO 750 J=1,I3
C
C        MAKE SECOND SECTION NEW FIRST SECTION.
C
            DO 670 I=0,N1
            OUTPUT(I)=OUTPUT2(I)
 670        CONTINUE
C
C        READ IN SECOND SECTION AND TRANSFORM.
C
         READ BINARY (1) (OUTPUT2(L),L=0,N1)
         CALL DFT4(OUTPUT2(0),N,1)
         IF (K.LT.N1) GO TO 700
C
C        IN THIS CASE THE OVERLAP CHOOSEN WAS LESS
C  THAN 50 PERCENT.  THEREFORE, IF I IS LESS THAN
C  M, NORMALIZE THE FIRST SECTION AND STORE REAL
C  PART IN IREPLAY.  OTHERWISE, AVERAGE THE TWO
C  SECTIONS TOGETHER, NORMALIZE, AND STORE REAL
C  PART IN IREPLAY.
C
            DO 690 I=M1,K
            IF (I.GE.M) OUTPUT(I)=(OUTPUT(I)+OUTPUT2(I-M))/
     *      (2*N)
            IF (I.LT.M) OUTPUT(I)=(OUTPUT(I))/N
            IREPLAY(I)=REAL(OUTPUT(I))
 690        CONTINUE
C
C        STORE IREPLAY ON DISK.
C
         WRITE BINARY (3) (IREPLAY(L),L=M1,K)
         GO TO 750
C
C        OVERLAP GREATER THAN OR EQUAL TO 50 PERCENT.
C  THEREFORE, AVERAGE THE TWO SECTIONS TOGETHER,
C  NORMALIZE, AND STORE IN IREPLAY.
C
 700        DO 730 I=M,K
            OUTPUT(I)=(OUTPUT(I)+OUTPUT2(I-M))/(2*N)
            IREPLAY(I)=REAL(OUTPUT(I))
 730        CONTINUE
C
C        STORE IREPLAY ON DISK.
C
```

```
      WRITE BINARY (3) (IREPLAY(L),L=M,K)
C
C     RETURN TO READ IN NEW SECTION OF 'WINDOW.'
C
 750  CONTINUE
C
C     FINAL SECTION OF SPEECH
C
      IF (K.LT.N1) GO TO 800
C
C     OVERLAP IS LESS THAN 50 PERCENT.  THEREFORE,
C  NORMALIZE LAST SECTION OF SPEECH AND STORE REAL
C  PART IN IREPLAY.
C
      DO 770 I=M1,N1
      OUTPUT2(I)=(OUTPUT2(I))/N
      IREPLAY(I)=REAL(OUTPUT2(I))
 770  CONTINUE
C
C     STORE IREPLAY ON DISK.
C
      WRITE BINARY (3) (IREPLAY(L),L=M1,N1)
      GO TO 1000
C
C     OVERLAP GREATER THAN OR EQUAL TO 50 PERCENT.
C  THEREFORE, IF I IS LESS THAN M1, AVERAGE LAST
C  SECTION OF 'WINDOW' WITH NEXT TO LAST SECTION
C  OF 'WINDOW', NORMALIZE, AND STORE REAL PART
C  IN IREPLAY.
C
 800  DO 830 I=M,N1
      IF (I.LT.M1) OUTPUT2(I)=(OUTPUT2(I)+OUTPUT(I+M))/
     * (2*N)
      IF (I.GE.M1) OUTPUT2(I)=(OUTPUT2(I))/N
      IREPLAY(I)=REAL(OUTPUT2(I))
 830  CONTINUE
C
C     STORE IREPLAY ON DISK.
C
      WRITE BINARY (3) (IREPLAY(L),L=M,N1)
      GO TO 1000
 900  WRITE(10,910) IER
 910  FORMAT(" FILE DELETING ERROR CODE - ",I3)
      GO TO 1000
 920  WRITE(10,930) IER
 930  FORMAT(" FILE OPENING ERROR CODE - ",I3)
      GO TO 1000
 1000 CALL RESET
      STOP
      END
```

```
C       SENDISP
C-----------------------------------------------------------
C
C
        DIMENSION IPASS(256)
C
C       OPEN DISPLAY FILE..
C
        CALL OPEN(4,"DISPLAY",2,IER)
        IF (IER.EQ.1) GO TO 40
C
C       ERROR MESSAGE.
C
        TYPE "OPEN FILE ERROR CODE- ",IER
        STOP,        !!!!!!!!!!ERROR ABORT!!!!!!!!!!
C
C       READ IN DISPLAY SIZE.
C
 40     ACCEPT "ENTER VERTICAL RESOLUTION - ",IVER
        ACCEPT "ENTER HORIZONTAL RESOLUTION - ",IHOZ
C
C       READY FOR TRANSFER.
C
        TYPE "<7>"
        TYPE "             ************* READY **************"
C
C       NTOT - NUMBER OF INTEGERS TO BE TRANSFERED.
C       ISTOP - NUMBER OF BLOCKS TO BE TRANSFERED (MINUS
C  ONE).
C
        NTOT=IVER*(IHOZ/4)
        ISTOP=(NTOT/256)-1
        DO 200 I=0,ISTOP
C
C       READ IN A BLOCK OF INTEGERS.
C
        CALL RDBLK(4,I,IPASS,1,IER)
        IF (IER.EQ.1) GO TO 60
C
C       ERROR MESSAGE.
C
        TYPE "READ FILE ERROR CODE- ",IER
        STOP,    !!!!!!!!!!ERROR ABORT!!!!!!!!!!
C
C       TRANSFER INTEGER ARRAY ONE AT A TIME VIA PASSDAT.
C
 60        DO 100 J=1,256
           K = IPASS(J)
           CALL PASSDAT(K)
           IF (J.EQ.1) TYPE"<33><133>A<33><133>2K<7>
     +                                   TRANSFERRING DATA"
 100       CONTINUE
 200    CONTINUE
```

```
C
C       8192 INTEGERS ARE EXPECTED TO BE TRANSFERED.
C  TRANSFER ZEROS UNTIL THE PROPER NUMBER HAS BEEN SENT.
C
        IF (NTOT.EQ.8192) GO TO 400
        DO 300 I=NTOT,8191
        K=0
        CALL PASSDAT(K)
 300    CONTINUE
C
C       CLOSE ALL FILES.
C
 400    CALL RESET
        TYPE "<7>"
        STOP,              *************** DONE ***************
        END
```

```
C       SENDWOB
C--------------------------------------------------------------
C
C
        DIMENSION IPASS(256)
C
C       OPEN DISPLAY FILE.
C
        CALL OPEN(4,"DISPLAY",2,IER)
        IF (IER.EQ.1) GO TO 40
C
C       ERROR MESSAGE.
C
        TYPE "OPEN FILE ERROR CODE- ",IER
        STOP,        !!!!!!!!!!ERROR ABORT!!!!!!!!!!
C
C       ENTER DISPLAY SIZE.
C
 40     ACCEPT "ENTER VERTICAL RESOLUTION - ",IVER
        ACCEPT "ENTER HORIZONTAL RESOLUTION - ",IHOZ
C
C       READY FOR TRANSFER.
C
        TYPE "<7>"
        TYPE "           ************* READY **************"
C
C       NTOT - NUMBER OF INTEGERS TO BE TRANSFERED.
C       ISTOP - NUMBER OF BLOCKS TO BE TRANSFERED (MINUS
C  ONE).
C
        NTOT=IVER*(IHOZ/4)
        ISTOP=(NTOT/256)-1
        DO 200 I=0,ISTOP
C
C       READ IN A BLOCK OF INTEGERS.
C
        CALL RDBLK(4,I,IPASS,1,IER)
        IF (IER.EQ.1) GO TO 60
C
C       ERROR MESSAGE.
C
        TYPE "READ FILE ERROR CODE- ",IER
        STOP,   !!!!!!!!!!ERROR ABORT!!!!!!!!!!
C
C       TRANSFER INTEGER ARRAY ONE AT A TIME VIA PASCURS.
C
 60        DO 100 J=1,256
           K = IPASS(J)
           CALL PASCURS(K)
           IF (J.EQ.1) TYPE"<33><133>A<33><133>2K<7>
     +                                      TRANSFERRING DATA"
 100       CONTINUE
 200    CONTINUE
```

```
C
C       8192 INTEGERS ARE EXPECTED TO BE TRANSFERED.
C  TRANSFER ZEROS UNTIL THE PROPER NUMBER HAS BEEN SENT.
C
        IF (NTOT.EQ.8192) GO TO 400
        DO 300 I=NTOT,8191
        K=0
        CALL PASCURS(K)
 300    CONTINUE
C
C       CLOSE ALL FILES.
C
 400    CALL RESET
        TYPE "<7>"
        STOP,              ************** DONE ***************
        END
```

```
;
;       PASCURS
;

        .TITL   PASCURS
        .ENT    PASCURS
        .EXTD   .FARL,.FRET


        .NREL

        FS.

PASCURS:        JSR     @.FARL

        LDA     0,@TMP,3        ;GET WORD
        DOAS    0,25            ;SEND IT TO CROMEMCO AND SET
BUSY

TEST:   SKPDN   25              ;TEST FOR DONE FLAG
        JMP     TEST            ;IF FALSE, TEST AGAIN

        JSR     @.FRET          ;RETURN TO MAIN ROUTINE

        FS.=1                   ;FRAME SIZE
        TMP=-167

        .END    PASCURS
```

```
;
;       PASSDAT
;

        .TITL   PASSDAT
        .ENT    PASSDAT
        .EXTD   .FARL,.FRET


        .NREL

        FS.

PASSDAT:        JSR     @.FARL

        LDA     0,@TMP,3        ;GET WORD
        COM     0,0             ;COMPLEMENT IT
        DOAS    0,25            ;SEND IT TO CROMEMCO AND SET
BUSY

TEST:   SKPDN   25              ;TEST FOR DONE FLAG
        JMP     TEST            ;IF FALSE, TEST AGAIN

        JSR     @.FRET          ;RETURN TO MAIN ROUTINE

        FS.=1                   ;FRAME SIZE
        TMP=-167

        .END    PASSDAT
```

Appendix C

S-100 Programs

```
      PROGRAM DISPLAY
      BYTE    MEM,NOVHI,NOVLO,NOVST,SETDON,
    * ISTAT,ON,PORT,MASK,CHECK,OFF
C
C     NOVHI & NOVLO ARE INPUT PORTS FOR HIGH
C  & LOW BYTES FROM THE NOVA.
C
      NOVHI=X'A1'
      NOVLO=X'A0'
C
C     NOVST & SETDON ARE STATUS PORTS TO CONTROL
C  THE TRANSFER OF DATA.
C
      NOVST=X'D0'
      SETDON=X'B0'
C
C     PORT IS THE PORT FOR THE VIDEO SYSTEM.  ON
C  IS THE COMMAND TO TURN THE VIDEO DMA ON.  LIKEWISE,
C  OFF TURNS THE VIDEO DMA OFF.
C
      PORT=X'FF'
      ON=X'84'
      OFF=X'04'
C
C     MASK IS USED TO CHECK THE LEAST SIGNIFICANT BIT.
C
      MASK=X'01'
C
C     PAUSE - WAIT UNTIL NOVA READY FOR TRANSFER.
C
      PAUSE
C
C     STORE DATA IN MEMORY LOCATIONS 4096 - 20479
C
 40   DO 100 I=4096,20479,2
C
C     CHECK NOVST UNTIL BUSY (LSB=1) IS SET.
 50   ISTAT=INP(NOVST)
      CHECK=ISTAT.AND.MASK
      IF (CHECK.NE.1) GO TO 50
C
C     INPUT HIGH BYTE.
C
      MEM=INP(NOVHI)
C
```

```
C
C        STORE HIGH BYTE IN LOCATION SPECIFED BY I
C
         CALL POKE(I,MEM)
         J=I+1
C
C        INPUT LOW BYTE
C
         MEM=INP(NOVLO)
C
C        STORE LOW BYTE IN LOCATION SPECIFED BY J
C
         CALL POKE(J,MEM)
C
C        SET DONE FLAG IN NOVA
C
         CALL OUT(STBDON,ON)
 100     CONTINUE
C
C        TURN ON VIDEO
C
         CALL OUT(PORT,ON)
         PAUSE
C
C        TURN OFF VIDEO
C
         CALL OUT(PORT,OFF)
C
C        PAUSE - TYPE T TO TERMINATE PROGRAM OR
C     ANYTHING ELSE TO RETURN AND GET NEW DISPLAY.
C
         PAUSE
C
C        RETURN TO GET NEW DISPLAY
C
         GO TO 40
         END
```

```
      PROGRAM CURSOR
      BYTE     MEM,NOVHI,NOVLO,NOVST,SETDON,
     * ISTAT,ON,PORT,MASK,CHECK,OFF,CURS1,
     * CURS2,VER,BLACK,WHITE,MASK2,MASK3
C
C     NOVHI & NOVLO ARE INPUT PORTS FOR THE HIGH
C & LOW BYTES FROM THE NOVA
C
      NOVHI=X'A1'
      NOVLO=X'A0'
C
C     NOVST AND SETDON ARE STATUS PORTS TO CONTROL
C THE TRANSFER OF DATA.
C
      NOVST=X'D0'
      SETDON=X'E0'
C
C     PORT IS THE PORT FOR THE VIDEO SYSTEM.  ON
C IS THE COMMAND TO TURN THE VIDEO DMA ON.  LIKEWISE,
C OFF TURNS THE VIDEO DMA OFF.
C
      PORT=X'FF'
      ON=X'84'
      OFF=X'04'
C
C     BLACK & WHITE ARE BYTES USED TO MAKE THE
C CURSORS
C
      BLACK=X'00'
      WHITE=X'FF'
C
C     MASK IS USED TO CHECK THE LEAST SIGNIFICANT BIT.
C       MASK2 IS USED TO CHECK THE MOST SIGNIFICANT BIT.
C         MASK3 IS USED TO MASK OFF THE MOST
C SIGNIFICANT BIT.
C
      MASK=X'01'
      MASK2=X'80'
      MASK3=X'7F'
C
C     TURN VIDEO ON
C
      CALL OUT(PORT,ON)
      PAUSE
C
C     TURN VIDEO OFF
C
      CALL OUT(PORT,OFF)
C
C     INITIALIZE ICOUNT
C
      ICOUNT=0
C
C     STORE INPUT DATA IN MEMORY LOCATIONS
```

```
C   20600 - 20605
C
 140    DO 200 I=20600,20604,2
C
C       CHECK NOVST UNTIL BUSY (LSB=1) IS SET
C
 150    ISTAT=INP(NOVST)
        CHECK=ISTAT.AND.MASK
        IF (CHECK.NE.1) GO TO 150
C
C       INPUT HIGH BYTE
C
        MEM=INP(NOVHI)
C
C       STORE HIGH BYTE IN LOCATION SPECIFED BY I
C
        CALL POKE(I,MEM)
        J=I+1
C
C       INPUT LOW BYTE
C
        MEM=INP(NOVLO)
C
C       STORE LOW BYTE IN LOCATION SPECIFIED BY J
C
        CALL POKE(J,MEM)
C
C       SET DONE FLAG IN NOVA
C
        CALL OUT(SETDON,ON)
 200    CONTINUE
C
C       CURS1 EQUALS LOW BYTE OF FIRST WORD TRANSFERED
C
        CURS1=PEEK(20601)
C
C       CURS2 EQUALS LOW BYTE OF SECOND WORD TRANSFERED
C
        CURS2=PEEK(20603)
C
C       VER EQUALS LOW BYTE OF THIRD WORD TRANSFERED
C
        VER=PEEK(20605)
C
C       IVER IS THE VERTICAL RESOLUTION IN PIXELS
C
        IVER=VER*64
C
C       IVERTW - IVER TWICE
C
        IVERTW=IVER*2
C
C       CHECK MOST SIGNIFICANT BIT OF CURS1
C
```

```
      CHECK=MASK2.AND.CURS1
C
C     IF MSB = 0 GO TO 210
C
      IF (CHECK.EQ.0) GO TO 210
C
C     MASK OFF MSB
C
      CURS1=CURS1.AND.MASK3
C
C     SET INTEGER EQUAL TO BYTE VALUE MINUS MSB
C
      I1=CURS1
C
C     ONCE IN INTEGER FORM ADD THE VALUE OF THE MSB
C
      I1=I1+128
      GO TO 220
C
C     MSB=0, THEREFORE OK TO SET INTEGER EQUAL TO
C  BYTE VALUE DIRECTLY
C
 210  I1=CURS1
C
C     REPEAT ABOVE PROCESS FOR CURS2
C
 220  CHECK=MASK2.AND.CURS2
      IF (CHECK.EQ.0) GO TO 230
      CURS2=CURS2.AND.MASK3
      I2=CURS2
      I2=I2+128
      GO TO 240
 230  I2=CURS2
C
C     IF ICOUNT EQUALS 2 CURSORS ARE ALREADY
C  DISPLAYED.  THEREFORE DISPLAY MUST BE RESTORED
C  BEFORE NEW CURSOR POSITIONS CAN BE DISPLAYED.
C
 240  IF (ICOUNT.EQ.2) GO TO 400
C
C     STORE CURSOR POSITIONS (I1 & I2) IN ICURS1
C  AND ICURS2
C
      ICURS1=I1
      ICURS2=I2
C
C     J1 IS THE POINTER TO THE STORAGE LOCATIONS
C  FOR THE DISPLAY OVERWRITTEN BY THE CURSORS
C
      J1=21000
C
C     J2 IS THE POINTER TO THE STARTING LOCATION
C  FOR THE CURSOR (IT TAKES IVER/2 MEMORY LOCATIONS
C  TO STORE IVER PIXELS)
```

```
C
        J2=4096+I1*IVER/2
C
C       WRITE A BLACK LINE TWO HORIZ. WIDTHS WIDE
C
 250    DO 300 I=1,IVER
C
C       GET MEMORY FROM LOCATION J2 AND STORE IN
C  LOCATION J1
C
        MEM=PEEK(J2)
        CALL POKE(J1,MEM)
C
C       STORE 'BLACK' IN LOCATION J2 AND INCREMENT
C  J1 & J2
C
        CALL POKE(J2,BLACK)
        J1=J1+1
        J2=J2+1
 300    CONTINUE
C
C       REPEAT ABOVE PROCEDURE EXCEPT WRITE A WHITE
C  LINE
C
        DO 350 I=1,IVER
        MEM=PEEK(J2)
        CALL POKE(J1,MEM)
        CALL POKE(J2,WHITE)
        J1=J1+1
        J2=J2+1
 350    CONTINUE
C
C       INCREMENT ICOUNT AND IF ICOUNT EQUALS
C  2, BOTH CURSORS HAVE BEEN WRITTEN SO JUMP
C  TO THE END OF THE PROGRAM TO DISPLAY THEM
C
        ICOUNT=ICOUNT+1
        IF (ICOUNT.EQ.2) GO TO 800
C
C       SET UP J1 & J2 FOR USE IN WRITING THE
C  SECOND CURSOR
C
        J1=21000+IVERTW
        J2=4096+I2*IVER/2
        GO TO 250
C
C       THE CURSORS HAVE ALREADY BEEN DISPLAYED
C
C       IF I1 EQUALS ICURS1 THE NEW DESIRED LOCATION
C  FOR THE FIRST CURSOR IS THE SAME AS CURRENT
C  LOCATION OF THE CURSOR
C
 400    IF (I1.EQ.ICURS1) GO TO 700
C
```

```
C         J1 IS THE POINTER TO THE STORAGE LOCATIONS
C  FOR THE MEMORY THAT IS OVERWRITTEN BY THE
C CURSOR
          J1=21000
C
C         J2 IS THE POINTER TO THE STARTING POINT
C  IN MEMORY FOR THE NEW CURSOR
C
          J2=4096+I1*IVER/2
C
C         J3 IS THE POINTER TO THE STARTING POINT
C  IN MEMORY FOR THE OLD CURSOR
C
          J3=4096+ICURS1*IVER/2
C
C         STORE THE NEW CURSOR POSITION IN ICURS1
C  AND SET ICOUNT
C
          ICURS1=I1
          ICOUNT=1
C
C         RESTORE THE MEMORY OVERWRITTEN BY THE OLD
C  CURSOR
C
 450      DO 500 I=1,IVERTW
C
C         GET MEMORY FROM LOCATION J1 AND STORE IT IN
C  LOCATION J3
C
          MEM=PEEK(J1)
          CALL POKE(J3,MEM)
C
C         INCREMENT J1 AND J3
C
          J1=J1+1
          J3=J3+1
 500      CONTINUE
C
C         MEMORY IS NOW RESTORED.  RESET POINTER
C  J1
C
          J1=21000
          IF (ICOUNT.EQ.2) J1=J1+IVERTW
C
C         WRITE A BLACK LINE TWO HORIZ. WIDTHS WIDE
C
          DO 550 I=1,IVER
C
C         GET MEMORY FROM LOCATION J2 AND STORE IT IN
C  LOCATION J1
C
          MEM=PEEK(J2)
          CALL POKE(J1,MEM)
C
```

```
C        STORE 'BLACK' IN LOCATION J2 AND INCREMENT
C J1 & J2
C
         CALL POKE(J2,BLACK)
         J1=J1+1
         J2=J2+1
 550     CONTINUE
C
C        REPEAT ABOVE PROCEDURE TO WRITE A WHITE
C  LINE
C
         DO 600 I=1,IVER
         MEM=PEEK(J2)
         CALL POKE(J1,MEM)
         CALL POKE(J2,WHITE)
         J1=J1+1
         J2=J2+1
 600     CONTINUE
C
C        IF ICOUNT EQUALS 2 BOTH CURSORS ARE READY
C  TO BE DISPLAYED
C
         IF (ICOUNT.EQ.2) GO TO 800
C
C        IF I2 EQUALS ICURS2 THE NEW DESIRED LOCATION
C  FOR THE SECOND CURSOR IS THE SAME AS THE CURRENT
C  LOCATION OF THE CURSOR.  SET ICOUNT AND JUMP TO
C  DISPLAY CURSORS
C
 700     IF (I2.NE.ICURS2) GO TO 750
         ICOUNT=2
         GO TO 800
C
C        SET POINTERS J1,J2 & J3 FOR THE SECOND
C  CURSOR
C
 750     J1=21000+IVERTH
         J2=4096+I2*IVER/2
         J3=4096+ICURS2*IVER/2
C
C        STORE NEW CURSOR POSITION IN ICURS2 AND
C  SET ICOUNT
C
         ICURS2=I2
         ICOUNT=2
         GO TO 450
C
C        TURN VIDEO ON
C
 800     CALL OUT(PORT,ON)
         PAUSE
C
C        TURN VIDEO OFF
C
```

```
      CALL OUT(PORT,OFF)
      PAUSE
C
C     RETURN TO GET NEW CURSOR POSITION
C
      GO TO 140
      END
```

```
      PROGRAM ONE
      BYTE     MEM,NOVHI,NOVLO,NOVST,SETDON,
     * ISTAT,ON,PORT,MASK,CHECK,OFF,CURS1,
     * CURS2,VER,BLACK,WHITE,MASK2,MASK3
C
C     NOVHI & NOVLO ARE INPUT PORTS FOR THE HIGH
C  & LOW BYTES FROM THE NOVA
C
      NOVHI=X'A1'
      NOVLO=X'A0'
C
C     NOVST AND SETDON ARE STATUS PORTS TO CONTROL
C  THE TRANSFER OF DATA.
C
      NOVST=X'D0'
      SETDON=X'B0'
C
C     PORT IS THE PORT FOR THE VIDEO SYSTEM.  ON
C  IS THE COMMAND TO TURN THE VIDEO DMA ON.  LIKEWISE,
C  OFF TURNS THE VIDEO DMA OFF.
C
      PORT=X'FF'
      ON=X'84'
      OFF=X'04'
C
C     BLACK & WHITE ARE BYTES USED TO MAKE THE
C  CURSORS
C
      BLACK=X'00'
      WHITE=X'FF'
C
C     MASK IS USED TO CHECK THE LEAST SIGNIFIGANT BIT.
C       MASK2 IS USED TO CHECK THE MOST SIGNIFIGANT BIT.
C         MASK3 IS USED TO MASK OFF THE MOST
C  SIGNIFIGANT BIT.
C
      MASK=X'01'
      MASK2=X'80'
      MASK3=X'7F'
C
C     TURN VIDEO ON
C
      CALL OUT(PORT,ON)
      PAUSE
C
C     TURN VIDEO OFF
C
      CALL OUT(PORT,OFF)
C
C     INITIALIZE ICOUNT
C
      ICOUNT=0
C
C     STORE INPUT DATA IN MEMORY LOCATIONS
```

```
C  20600 - 20605
C
 140     DO 200 I=20600,20604,2
C
C        CHECK NOVST UNTIL BUSY (LSB=1) IS SET
C
 150     ISTAT=INP(NOVST)
         CHECK=ISTAT.AND.MASK
         IF (CHECK.NE.1) GO TO 150
C
C        INPUT HIGH BYTE
C
         MEM=INP(NOVHI)
C
C        STORE HIGH BYTE IN LOCATION SPECIFED BY I
C
         CALL POKE(I,MEM)
         J=I+1
C
C        INPUT LOW BYTE
C
         MEM=INP(NOVLO)
C
C        STORE LOW BYTE IN LOCATION SPECIFIED BY J
C
         CALL POKE(J,MEM)
C
C        SET DONE FLAG IN NOVA
C
         CALL OUT(SETDON,ON)
 200     CONTINUE
C
C        CURS1 EQUALS LOW BYTE OF FIRST WORD TRANSFERED
C
         CURS1=PEEK(20601)
C
C        CURS2 EQUALS LOW BYTE OF SECOND WORD TRANSFERED
C
         CURS2=PEEK(20603)
C
C        VER EQUALS LOW BYTE OF THIRD WORD TRANSFERED
C
         VER=PEEK(20605)
C
C        IVER IS THE VERTICAL RESOLUTION IN PIXELS
C
         IVER=VER*64
C
C        IVER2 - HALF OF IVER
C
         IVER2=IVER/2
C
C        CHECK MOST SIGNIFIGANT BIT OF CURS1
C
```

```
      CHECK=MASK2.AND.CURS1
C
C     IF MSB = 0 GO TO 210
C
      IF (CHECK.EQ.0) GO TO 210
C
C     MASK OFF MSB
C
      CURS1=CURS1.AND.MASK3
C
C     SET INTEGER EQUAL TO BYTE VALUE MINUS MSB
C
      I1=CURS1
C
C     ONCE IN INTEGER FORM ADD THE VALUE OF THE MSB
C
      I1=I1+128
      GO TO 220
C
C     MSB=0, THEREFORE OK TO SET INTEGER EQUAL TO
C  BYTE VALUE DIRECTLY
C
 210  I1=CURS1
C
C     REPEAT ABOVE PROCESS FOR CURS2
C
 220  CHECK=MASK2.AND.CURS2
      IF (CHECK.EQ.0) GO TO 230
      CURS2=CURS2.AND.MASK3
      I2=CURS2
      I2=I2+128
      GO TO 240
 230  I2=CURS2
C
C     IF ICOUNT EQUALS 2 CURSORS ARE ALREADY
C  DISPLAYED.  THEREFORE DISPLAY MUST BE RESTORED
C  BEFORE NEW CURSOR POSITIONS CAN BE DISPLAYED.
C
 240  IF (ICOUNT.EQ.2) GO TO 400
C
C     STORE CURSOR POSITIONS (I1 & I2) IN ICURS1
C  AND ICURS2
C
      ICURS1=I1
      ICURS2=I2
C
C     J1 IS THE POINTER TO THE STORAGE LOCATIONS
C  FOR THE DISPLAY OVERWRITTEN BY THE CURSORS
C
      J1=21000
C
C     J2 IS THE POINTER TO THE STARTING LOCATION
C  FOR THE CURSOR (IT TAKES IVER2 MEMORY LOCATIONS
C  TO STORE IVER PIXELS)
```

```
C
      J2=4096+I1*IVER2
C
C     WRITE A BLACK LINE ONE HORIZ. WIDTHS WIDE
C
 250  DO 300 I=1,IVER2
C
C     GET MEMORY FROM LOCATION J2 AND STORE IN
C  LOCATION J1
C
      MEM=PEEK(J2)
      CALL POKE(J1,MEM)
C
C     STORE 'BLACK' IN LOCATION J2 AND INCREMENT
C  J1 & J2
C
      CALL POKE(J2,BLACK)
      J1=J1+1
      J2=J2+1
 300  CONTINUE
C
C     REPEAT ABOVE PROCEDURE EXCEPT WRITE A WHITE
C  LINE
C
      DO 350 I=1,IVER2
      MEM=PEEK(J2)
      CALL POKE(J1,MEM)
      CALL POKE(J2,WHITE)
      J1=J1+1
      J2=J2+1
 350  CONTINUE
C
C     INCREMENT ICOUNT AND IF ICOUNT EQUALS
C  2, BOTH CURSORS HAVE BEEN WRITTEN SO JUMP
C  TO THE END OF THE PROGRAM TO DISPLAY THEM
C
      ICOUNT=ICOUNT+1
      IF (ICOUNT.EQ.2) GO TO 800
C
C     SET UP J1 & J2 FOR USE IN WRITING THE
C  SECOND CURSOR
C
      J1=21000+IVER
      J2=4096+I2*IVER2
      GO TO 250
C
C     THE CURSORS HAVE ALREADY BEEN DISPLAYED
C
C     IF I1 EQUALS ICURS1 THE NEW DESIRED LOCATION
C  FOR THE FIRST CURSOR IS THE SAME AS CURRENT
C  LOCATION OF THE CURSOR
C
 400  IF (I1.EQ.ICURS1) GO TO 700
C
```

```
C       J1 IS THE POINTER TO THE STORAGE LOCATIONS
C  FOR THE MEMORY THAT IS OVERWRITTEN BY THE
C CURSOR
        J1=21000
C
C       J2 IS THE POINTER TO THE STARTING POINT
C  IN MEMORY FOR THE NEW CURSOR
C
        J2=4096+I1*IVER2
C
C       J3 IS THE POINTER TO THE STARTING POINT
C  IN MEMORY FOR THE OLD CURSOR
C
        J3=4096+ICURS1*IVER2
C
C       STORE THE NEW CURSOR POSITION IN ICURS1
C  AND SET ICOUNT
C
        ICURS1=I1
        ICOUNT=1
C
C       RESTORE THE MEMORY OVERWRITTEN BY THE OLD
C  CURSOR
C
 450    DO 500 I=1,IVER
C
C       GET MEMORY FROM LOCATION J1 AND STORE IT IN
C  LOCATION J3
C
        MEM=PEEK(J1)
        CALL POKE(J3,MEM)
C
C       INCREMENT J1 AND J3
C
        J1=J1+1
        J3=J3+1
 500    CONTINUE
C
C       MEMORY IS NOW RESTORED.  RESET POINTER
C  J1
C
        J1=21000
        IF (ICOUNT.EQ.2) J1=J1+IVER
C
C       WRITE A BLACK LINE ONE HORIZ. WIDTHS WIDE
C
        DO 550 I=1,IVER2
C
C       GET MEMORY FROM LOCATION J2 AND STORE IT IN
C  LOCATION J1
C
        MEM=PEEK(J2)
        CALL POKE(J1,MEM)
C
```

```
C        STORE 'BLACK' IN LOCATION J2 AND INCREMENT
C J1 & J2
C
         CALL POKE(J2,BLACK)
         J1=J1+1
         J2=J2+1
 550     CONTINUE
C
C        REPEAT ABOVE PROCEDURE TO WRITE A WHITE
C  LINE
C
         DO 600 I=1,IVER2
         MEM=PEEK(J2)
         CALL POKE(J1,MEM)
         CALL POKE(J2,WHITE)
         J1=J1+1
         J2=J2+1
 600     CONTINUE
C
C        IF ICOUNT EQUALS 2 BOTH CURSORS ARE READY
C  TO BE DISPLAYED
C
         IF (ICOUNT.EQ.2) GO TO 800
C
C        IF I2 EQUALS ICURS2 THE NEW DESIRED LOCATION
C  FOR THE SECOND CURSOR IS THE SAME AS THE CURRENT
C  LOCATION OF THE CURSOR.  SET ICOUNT AND JUMP TO
C  DISPLAY CURSORS
C
 700     IF (I2.NE.ICURS2) GO TO 750
         ICOUNT=2
         GO TO 800
C
C        SET POINTERS J1,J2 & J3 FOR THE SECOND
C  CURSOR
C
 750     J1=21000+IVER
         J2=4096+I2*IVER2
         J3=4096+ICURS2*IVER2
C
C        STORE NEW CURSOR POSITION IN ICURS2 AND
C  SET ICOUNT
C
         ICURS2=I2
         ICOUNT=2
         GO TO 450
C
C        TURN VIDEO ON
C
 800     CALL OUT(PORT,ON)
         PAUSE
C
C        TURN VIDEO OFF
C
```

```
      CALL OUT(PORT,OFF)
      PAUSE
C
C     RETURN TO GET NEW CURSOR POSITION
C
      GO TO 140
      END
```

Vita

Paul Dundas was born on 22 September 1955 in Neptune, New Jersey. He graduated from high school in Neptune, New Jersey in 1973 and attended the United States Air Force Academy from which he received the degree of Bachelor of Science in Electrical Engineering in 1977. Upon graduation he was assigned to the 3246th Test Wing, Eglin AFB, Florida, where he worked as a test engineer until entering the School of Engineering, Air Force Institute of Technology, in June 1979.

Permanent Address: 7 Tucker Drive
Neptune City, N.J. 07753

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFIT/GE/EE/80D-17
2. GOVT ACCESSION NO.: AD A100791
3. RECIPIENT'S CATALOG NUMBER:
4. TITLE (and Subtitle): DESIGN AND DEVELOPMENT OF A GRAPHICS SYSTEM FOR DISPLAY OF SOUND SPECTROGRAMS
5. TYPE OF REPORT & PERIOD COVERED: MS Thesis
6. PERFORMING ORG. REPORT NUMBER:
7. AUTHOR(s): Paul A. Dundas, 1st Lt
8. CONTRACT OR GRANT NUMBER(s):
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Air Force Institute of Technology (AFIT-EN), Wright-Patterson AFB, Ohio 45433
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:
11. CONTROLLING OFFICE NAME AND ADDRESS:
12. REPORT DATE: December, 1980
13. NUMBER OF PAGES: 162
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):
15. SECURITY CLASS. (of this report):
15a. DECLASSIFICATION DOWNGRADING SCHEDULE:
16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

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

18. SUPPLEMENTARY NOTES

Approved for public release; IAW AFR 190-17
FREDERICK C. LYNCH, Major, USAF
Director of Public Affairs
16 JU...

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

Speech Analysis
Sound Spectrograms
Discrete Fourier Transform

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

A graphics system for display of sound spectrograms was designed and implemented on a Data General NOVA computer which is interconnected with a S-100 bus system. The system allows the user to interactively select a section of the spectrogram for further analysis through the use of a set of cursors.

The computer systems used in the development of this project are described. The theory involved in the generation of digital sound spectrograms is detailed along with the display methodology used. The user is provided the choice of four spectral windows and the relative merits of each window are described and compared. The six major classifications of spectrogram patterns, called sound-pattern groups, are described. Spectrograms that show the characteristic pattern for each sound-pattern group are presented along with spectrograms that show the differences between narrow and wide-band spectral analysis. The computer programs developed on both the NOVA and the S-100 system are described.

END

DATE
FILMED
7-81

DTIC