A digital filter using the 309 Intel 2920 signal processor

by

Rodney E. Lee

A Thesis Submitted to the

Graduate Faculty in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Major: Biomedical Engineering

Iowa State University
Ames, Iowa

1982

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

OBJECTIVE

This research was a preliminary study to develop a microcomputer instrument to noninvasively monitor the depth of anesthesia. It was planned that a number of methods of determining the depth of anesthesia would be incorporated into the microcomputer instrument. One method, the onset latency of a visual evoked response potential, has been demonstrated by individual researchers to be a credible indice of anesthesia depth (Uhl et al., 1980). Since, a microcomputer operates only on digital data, the visual evoked response potential must be processed by an analog-to-digital converter. Before the analog-to-digital conversion takes place, it is necessary the visual evoked response potential enter a low-pass filter to prevent aliasing effects that can occur during the analog-to-digital conversion process (Oppenheim and Schafer, 1975).

Therefore, the main focus of this research was to study the feasibility of using the Intel Corporation 2920 signal processing integrated circuit to filter digitally visual evoked response potentials. The 2920 processor combines an analog-to-digital converter, an arithmetic processor, and a digital-to-analog converter on a single integrated circuit. An additional purpose of this research was to evaluate the 2920 software support package needed in developing visual evoked filter programs for use on the 2920 signal processor.

This study will not try to correlate the onset latency of an evoked response potential with depth of anesthesia. The correlation process will be done on the soon to be developed microcomputer instrument.

LITERATURE REVIEW

Visual Evoked Response

Electrical recordings from the exposed surface of the brain or from the scalp reveal a continuous oscillating activity. A record of the electrical potentials is called an electroencephalogram (EEG). The EEG has been found useful in areas of clinical diagnosis of brain pathology and as a measure of the level of consciousness, but is inadequate in understanding brain function in vision (Perry and Childers, 1969). The visual evoked response (VER) is a waveform immersed in the EEG which is generated by an external photic stimulation. Due to random noise and the very small magnitude of the EEG (5 to 10 microvolts), a repetitive stimulus and the technique of signal averaging must be used to separate the VER from the EEG (Perry and Childers, 1969).

The EEG is a complex representation of all electrical activity that goes on in the brain, and therefore lacks specificity. It has been found that the VER might be a better index of anesthetic effect than the EEG (Uhl et al., 1980). Uhl et al. (1980) research data indicated that the recorded visual evoked responses during surgical anesthesia with halothane are qualitatively very similar to those recorded in the awake condition, although they found the latency of the VER progressively increased with increasing alveolar anesthetic concentration. They then concluded that latency measurements of the VER could be used as a possible tool in monitoring anesthesia depth.

As pointed out by Ackmann et al. (1979), disturbances of intracranial blood flow are associated with amplitude reductions of the evoked potential, and volume-occupying lesions are associated with waveform changes. They claim that clinically, the most important features of evoked potentials are general waveshape, latency, and amplitude of the various peaks and troughs. However, the main objective of their study was to determine a suitable upper frequency cutoff of their digital filter. They concluded that the frequency content of the records analyzed were largely contained in a band from from 0 to 200 Hz. They also felt that zero phase shift is necessary to prevent waveform distortion, namely latency and morphology of the evoked potential. Based on a frequency analysis using the Fast Fourier Transform, a ten-pole, low-pass, Butterworth filter was chosen.

Desmedt et al. (1974) claim that a system bandpass of 0 to 3 kHz is a requirement when studying the morphology and onset latency of the evoked potential. They claim that other researchers prefer a smooth waveform in which the higher frequency components have been drastically filtered out. Therefore, the interpretation of their results is based on incorrect data. Bennett et al. (1971) used a system bandpass of 3 Hz to 1 kHz in defining a set of statistical models that would provide information concerning the range of the normal visual evoked response parameters.

A review of a number of similar articles revealed that there is no agreed upon system bandpass range to filter evoked potentials correctly. However, there is a general concensus that the major frequencies of interest lie in the 1 to 50 Hz range. Based on this, the system bandpass in this research was chosen to be 1 to 100 Hz. The high frequency components of a visual evoked potential are not important in determining onset latency changes.

Microprocessor-Based Digital Filter

Conventional analog filter circuits use components such as operational amplifiers, transistors, precision resistors, capacitors, and diodes. The value of these components is specified within tolerances. Therefore, replacing defective parts may change the original specification of the circuit. Also, in a production environment, no two circuits will be exactly the same. Troubleshooting an analog filter circuit can be a time consuming task, since there is more than one component making up the analog circuit.

Large Scale Integration (LSI) techniques have produced a special computer on a single integrated circuit called the microprocessor. Digital filtering is a computational process in which a sequence of input numbers are converted into a sequence of output numbers by way of addition, multiplication by constants, and delays. A microprocessor can be programmed to perform this type of processing.

Agarwal and Priemer (1979) used the Intel 8080A microprocessor to implement their digital filter algorithm. The signal processing system consisted of three printed circuit boards. One board was a complete microcomputer which contained the 8080A processor, memory, and the necessary interface circuitry. The other two boards contained the data acquisition system and a hardware multiplier.

Although the 8080A has the necessary instructions to perform the required multiplications, the researchers decided to implement the multiplication process with a single integrated circuit multiplier in order to speed up processing. The effective on-line data processing rate was 2 kHz per s-order section (s is complex parameter). For a two-pole network, the sampling rate is 1 kHz, and for a four-pole network, the sampling rate is 500 Hz.

Walstrom (1979) also used the Intel 8080A microprocessor to implement his fourth-order bandpass digital filter. The digital signal processing system was designed to sample an electrocardiograph signal. The hardware consisted of a microprocessor with the necessary interface circuity, the data acquisition components, and a hardware multiplier. A sampling rate of 360 Hz was used. The author concluded that the 8080A digital bandpass filter system was a workable system, but lacked many refinements which would enhance the overall operation. An improvement suggested by the author was to implement the filter on the new generation of 16-bit microprocessors such as the Intel 8086, the Zilog Z-8000, or the Motorola 68000. These microprocessors offer greater program efficiency, faster instruction cycle time, and true 16-bit operations versus the pseudo 16-bit operations performed by the Intel 8080A and the Motorola 6809.

Typical microprocessors in use today are the Motorola 6800/6809 or the Intel 8080/8085. These microprocessors, as well as most digital microprocessors, are designed for general data processing and not for high-speed complex signal analysis. A microprocessor-based digital signal processing system can operate as a signal processor at frequencies to only a few hundred hertz.

In contrast, most general signal processing frequencies are in the kilohertz range. Signals such as speech, electrocardiogram, electroretinogram, and electroencephalogram are complex in nature, and in some cases, multiple signals must be processed in parallel. Therefore, due to the high sampling rates and the complexity of the signal, a general purpose microprocessor is not well suited for signal processing applications. A totally different microprocessor architecture is required to implement signal processing algorithms.

BACKGROUND

Digital Filters

A digital filter implemented in software processes a sequence of input numbers to produce a predefined sequence of output numbers. The central element in the concept of the digital filter is the linear difference equation as defined by Oppenheim and Schafer (1975).

$$y(n) + b_1 y(n-1) + b_2 y(n-2) + \ldots + b_k y(n-k) = a_0 x(n) + a_1 x(n-1) + a_2 x(n-2) + \ldots + a_k x(n-k)$$

where,

$x(n)$	= input sequence samples
$y(n)$	= output sequence samples
a_k and b_k	= filter weights
k	= order of the difference equation

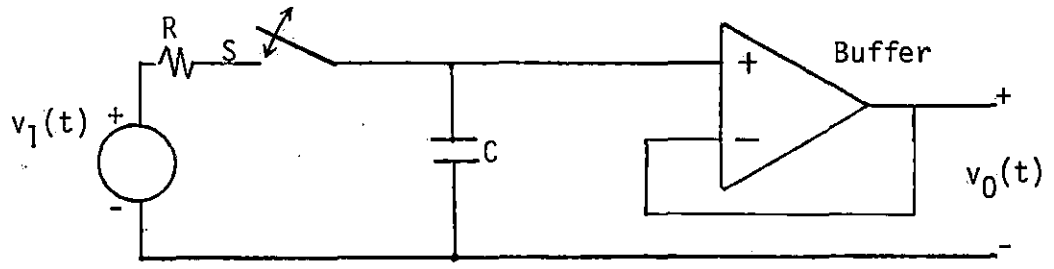
One way of deriving the linear difference equation is to start with classical analog-filter theory. Once a set of frequency characteristics has been defined, the classical theory is used to derive the analog transfer function. Corresponding to this transfer function is a differential equation. From the definition of the derivative, as the independent variable (sampling period in this case) approaches zero, the differential equation can be approximated by taking the difference between adjacent samples and dividing the difference by the independent variable. Therefore, as the sampling interval approaches zero, the approximation improves. This procedure of deriving a linear difference equation from a differential equation is simple. However, an alternative procedure proves to be more advantageous. This procedure is called the z-transform.

The first step in developing the z-transform is to note that signal samples of zero duration do not exist in any physical system. Any sampling operation implemented in hardware is necessarily associated with a holding operation producing discrete samples of nonzero duration. Consequently, a sample-and-hold operation will transform a continuous time signal into a stair-step output signal. As the sampling frequency increases, the stair-step will approach zero; hence, the sampled waveform will look more like a continuous signal in time.

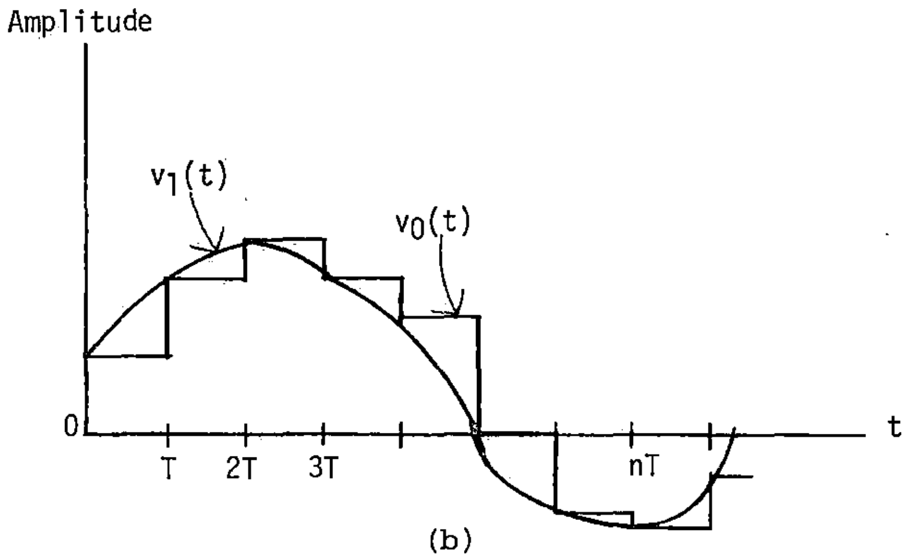
The most widely used sample-and-hold circuit has the form shown in Figure 1(a). When the input signal is continuous in time, the circuit produces a stair-step output waveform as shown in Figure 1(b).

A great deal of valuable information about the sampling process and the stair-step output waveform can be obtained from the Laplace transform of the stair-step signal. This transform is

$$V_0(s)= \int_0^\infty v_0(t)\exp(-st)\ dt.$$



(a)



(b)

FIGURE 1. Sample-and-hold circuit. (a) Circuit. (b) Waveform.

where,

$V_0(s)$ = sampled Laplace signal

$v_0(t)$ = sampled signal in time

By taking the Laplace transform of the stair-step signal, and from the definition of the integral, the integral can be expressed as the sum of infinitely many integrals, each spanning the time interval of one step in the waveform (Oppenheim and Schafer, 1975). The evaluation of these integrals yields:

$$V_0(s) = \frac{1 - \exp(-sT)}{s} \quad \Sigma_{n=0}^{\infty} v_1(nT)\exp(-nsT).$$

Implicit in this equation is the fact that during each step in the waveform, $v_0(t)$ is constant and equal to the value of $v_1(t)$ at the beginning of the step.

The factor preceding the summation is the result of the nonzero duration of the sample-and-hold operation. Oppenheim and Schafer (1975) has shown that this factor is the (sin x)/x function that amplitude modulates the magnitude of the filter function. The (sin x)/x often appears in the literature as

$$\frac{\sin(\pi * w/w_0)}{(\pi * w/w_0)}.$$

where

w = radian frequency

w_0 = radian sampling frequency

The sampling theorem indicates that the sampling rate must be greater than twice the highest frequency component of the input signal. If this condition is satisfied, the samples will contain all the significant information of the original signal. Therefore, if the sampling frequency is much greater than the highest frequency component, the (sin x)/x function will approach one, and consequently, amplitude distortion will be negligible.

The summation contains the values of all the samples of the input signal $v_1(t)$, and it also contains the instant of time $t=nT$ at which each sample occurs. Thus, it contains complete information about the sequence of samples $v_1(nT)$. It is, therefore, common practice to associate the summation with the process of sampling the input signal.

By defining a new symbol

$$z= \exp(sT),$$

the z-transform of the sequence of sample values becomes

$$V_1(z)= \Sigma_{n=0}^{\infty} v_1(nT)z^{-n}.$$

Referring to Figure 1(b), the stair-step signal is simply the unit step function, $u(t-nT)$, multiplied by the input signal. Therefore, the z^{-n} symbol represents a time delay, where n is the nth delay.

The important point is, given an arbitrary sequence of uniformly spaced sample values, the z-transform of the sequence can be written by inspection. Then, the Laplace transform of the associated stair-step wave can be written, and from the Laplace transform the frequency spectrum of the waveform can be calculated.

The z-transform, as developed, is a special form of the Laplace transform with a new variable, $z= \exp(sT)$. Also, it has been shown that the $(\sin x)/x$ weighting function is introduced when nonzero duration sampling occurs.

It will now be shown how an analog transfer function in the s-domain is transformed into a transfer function in the z-domain, and finally, how the linear difference equation is derived from the z-transfer function.

The appropriate transform for a difference equation is the z-transform, which performs the same role with difference equations as the Laplace transform carries out for differential equations. The use of the z-transform permits the specification of a digital filter from the s-transfer function directly in terms of delays, multiplications, and additions which is the correct form for hardware or software implementation.

From the linear difference equation, two types of filters may be realized. The Finite Impulse Response (FIR), or non-recursive filter, represents the summation of a limited number of input terms and thus has a finite memory. It has excellent phase characteristics but requires a large number of terms to obtain a relatively sharp attenuation characteristic. The Infinite Impulse Response (IIR) filter represents the summation of both input and output terms so that it can be considered as having an infinite memory. The IIR filter requires relatively few terms for similar attenuation characteristics, but possesses relatively poor phase response. Since phase response is not important and memory storage is limited in this application, the IIR filter will be developed.

The difference equation for an IIR filter is

$$y(n) = \Sigma_{k=1}^{N} a_k y(n-k) + \Sigma_{k=0}^{N} b_k x(n-k).$$

Taking the z-transform of the difference equation yields:

$$H(z) = \frac{\Sigma_{k=0}^{N} b_k z^{-k}}{1 - \Sigma_{k=1}^{N} a_k z^{-k}}.$$

Stanley (1975) suggests that since the design of an IIR filter is stable and causal, it is convenient to assume the coefficient $a_0 = 1$. The coefficients in the above equation are the same coefficients in the difference equation. So, by inspection of H(z), the difference equation for y(n) can be directly derived, hence, the digital filter is ready to be programmed into the computer. The question now becomes, how does one derive the z-transfer function H(z) from the given analog s-transfer function H(s)?

The bilinear transform is a method of transforming a function in the s-domain to a function in the z-domain. The standard form given by Stanley (1975) is

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

where

T = sampling period in seconds.

However, the bilinear transformation causes shifting of frequencies between the analog and digital transfer functions. A correction factor suggested by Stanley (1975) is

$$s = \frac{(z-1)}{(z+1)}$$

and

$$w_1 = \tan(w_2 T/2)$$

where

w_1 = analog frequency in radians/sec

w_2 = digital frequency in radians/sec

T = sampling period in seconds

2920 Signal Processor

With the great advances in Very Large Scale Integrated circuit technology (VLSI), it is now possible to integrate all the necessary real-time analog processing functions onto a single integrated circuit (2920 integrated package size is approximately 3.5 cm by 1.5 cm). VLSI advantages include small size, high reliability, relatively low cost, and low-power consumption. Basically, signal processing includes the generation, filtering, detection, and modulation of signals. Algorithms for signal processing repeatedly use multiplications and additions.

The Intel Corporation 2920 signal processor has been designed specifically to replace analog systems in real-time applications. An overview of a sampled-data system that uses the 2920 signal processor is shown in Figure 2. First, the analog input signal encounters an anti-aliasing filter and is sampled by a sample-and-hold circuit. Next, the just acquired analog sample is then converted to a digital signal and sent to the digital processor, where a signal analysis algorithm is performed. The result from the algorithm moves to a digital-to-analog converter, where it is converted back to an analog signal. The signal sample then moves to another sample-and-hold circuit and finally the sample is sent to a reconstruction filter. It is here the sample is smoothed to recover a continuous analog output signal.

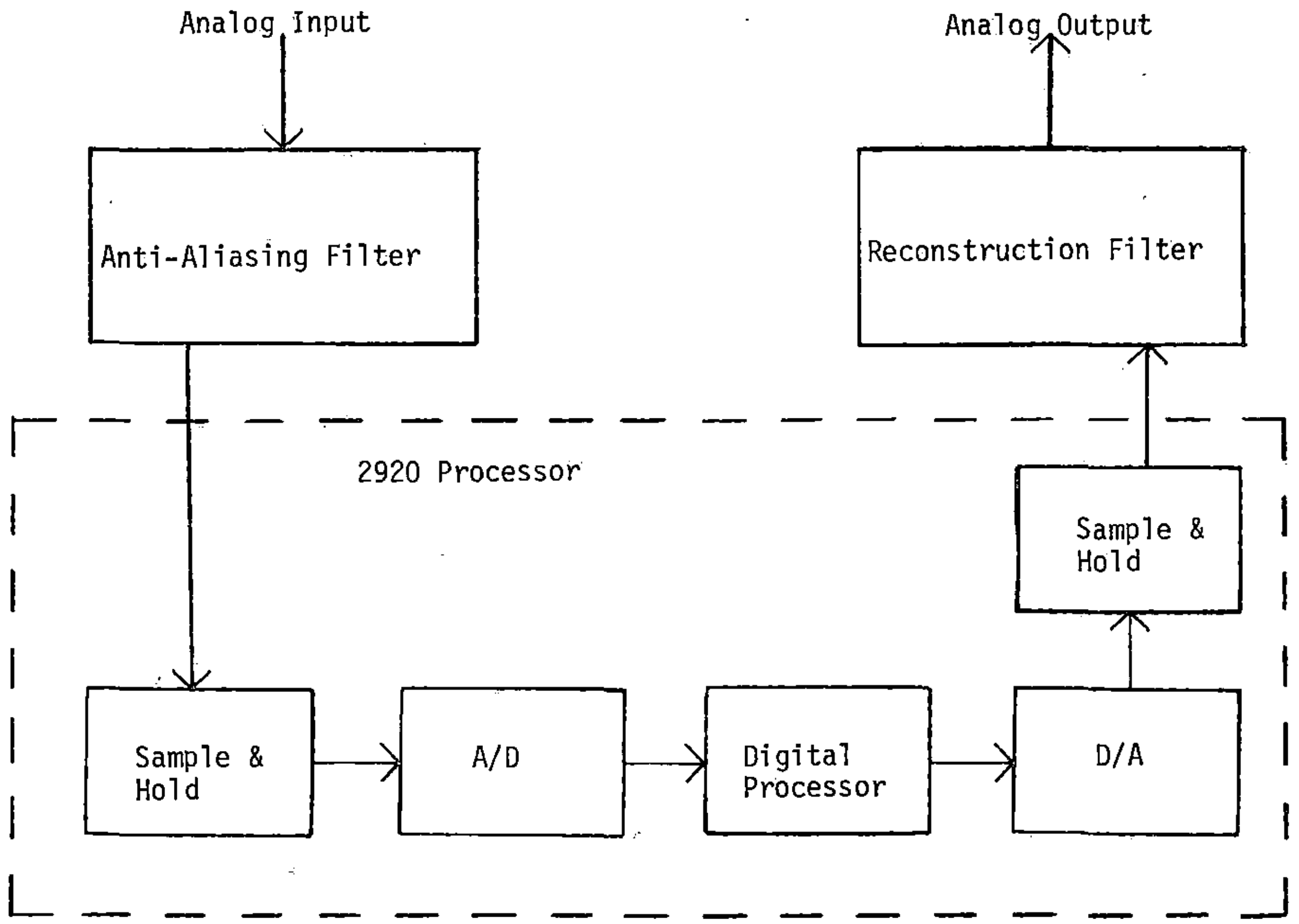


FIGURE 2. 2920 signal processor in a sampled data system

Figure 3 shows a block diagram of the integrated circuit. Referring to Figure 3, the 2920 signal processor is divided into three major sections:

1. a program storage area implemented with Electrically Programmable Read Only Memory (EPROM),
2. the arithmetic and logic unit (ALU) with data memory, and
3. the analog input/output (I/O) section.

A program within the EPROM controls all the functions of the 2920 processor. The size of the EPROM is 192 words of 24 bits each. Each word corresponds to one 2920 instruction. The 24 bits are split into the format as shown in Table 1. Each "field" within the format controls a specific aspect of the 2920 integrated circuit (Intel Corporation, 1981a).

The analog section performs analog-to-digital (A/D) and digital-to-analog (D/A) conversions. Included in the analog section are the following:

1. an input multiplexer (4 inputs: SIGIN0...SIGIN3),
2. an input sample-and-hold circuit,
3. a digital-to-analog converter (D/A),
4. a comparator, and
5. an output multiplexer with 8 output sample-and-hold circuits (SIGOUT0...SIGOUT7).

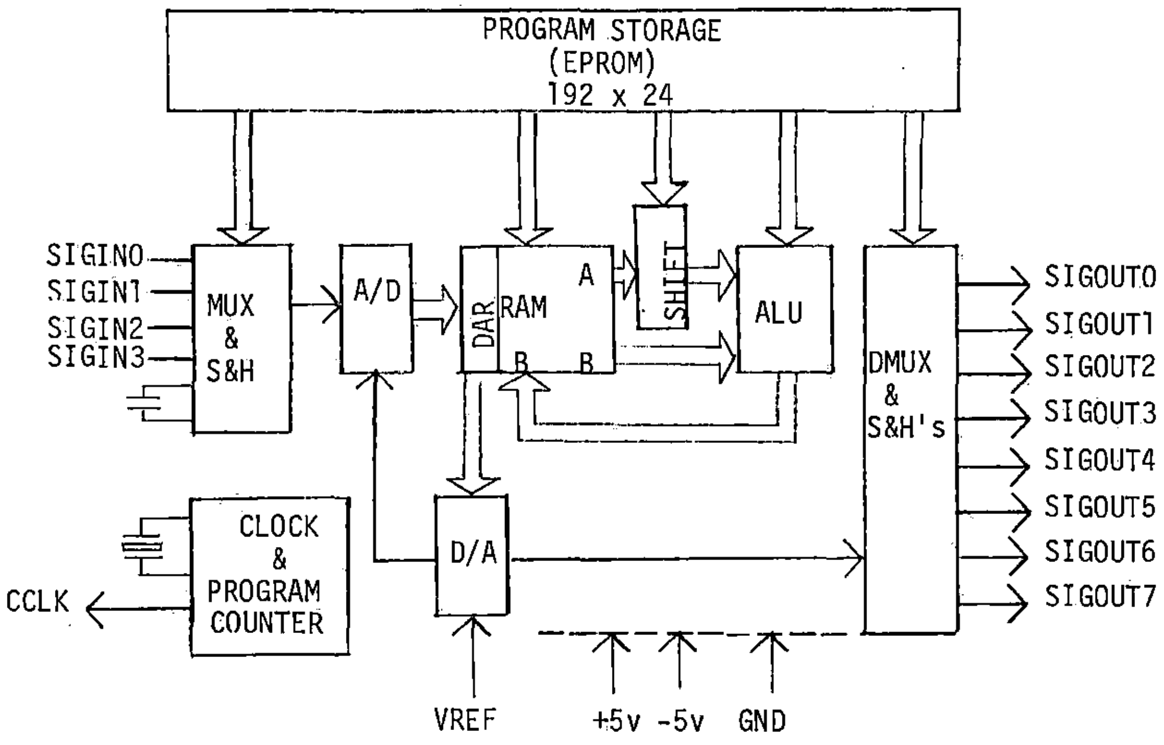


FIGURE 3. 2920 signal processor block diagram

TABLE 1. 2920 instruction format

ALU Instruction (3 bits)	B Address (6 bits)	A Address (6 bits)	Shift Code (4 bits)	Analog Instruction (5 bits)

A special register called the DAR (digital/analog register) acts as an interface between the digital and analog sections. It is 9 bits wide (one sign bit and eight amplitude bits), occupying the nine most significant positions of a word whose other bits are set to ones (Intel Corporation, 1981a). The DAR output is also tied directly to the D/A inputs. The DAR is used as a successive approximation register for analog-to-digital conversion, under control of the analog function instruction field.

In some applications, the output signal need only be a logic level of one or zero. SIGOUT pins can be selected to be either analog output or digital output (Intel Corporation, 1981a). The analog mode allows the full 9-bit D/A output to be present. The digital mode requires a LIM instruction to yield a zero or one decision (Intel Corporation, 1979).

The arithmetic section includes a 40-word by 25-bit random access memory (RAM), a scaler, and an arithmetic and logic unit (ALU). Data within this subsystem are processed using 25-bit two's complement arithmetic. The normal range of any variable "x" is considered to be

$$-1 < x < 1$$

and the smallest resolvable change "y", in any variable is given by (Intel Corporation, 1981a),

$$y = 2^{-24} = 5.96 \times 10^{-8}.$$

A positive voltage reference supply (VREF) of one volt is provided to the D/A converter to establish its voltage range. If digital output is required, VREF > 1.5 volts is necessary (Intel Corporation, 1981a).

The EPROM word addresses are numbered from 0 to 191 (decimal notation). During normal operation, the program counter begins at word address zero and the program sequentially executes through word address 191, or when an End of Program (EOP) instruction is encountered (Intel Corporation, 1979). The cycle then begins again at word address zero.

Program length determines the sampling frequency of the analog signal. If an input is sampled once per program pass, the sampling frequency is

$$1/(NT)$$

where

N = number of instructions

T = 2920 instruction cycle time.

Example: if N = 192 and T = 800 nanoseconds, then the sampling rate is approximately 6510 Hz.

The EPROM fetch/execute cycle is "pipelined" four deep, meaning the next four instructions are being fetched while the previously fetched instructions are being executed. The term "pipeline" refers to the fact that the processor has several hardware components which perform different operations simultaneously and pass data from one component to the next as through a pipe.

Limitations of the 2920 Signal Processor

The 2920 signal processor limitations are established by the size of the memory, the speed and capability of the processor, and the resolution of the A/D and D/A converters. Currently, there are three different 2920 signal processors being produced by the Intel Corporation. They are the 400 ns, 600 ns and 800 ns processors (Intel Corporation, 1981a). The different processor speeds refer to one instruction cycle time. For full 192 instructions, the sampling rates are approximately 13020 Hz, 8680 Hz, and 6510 Hz respectively. Faster sampling rates are achieved with smaller size programs.

A typical digital filter requires at least one RAM word per pole or two per complex conjugate pole pair. The size of the RAM limits the number of poles to less than 40, or less than 20 complex conjugate pairs. The number of EPROM words needed to realize a complex conjugate pole pair is approximately 10 to 11 (Intel Corporation, 1980a).

When an analog signal is digitized, it is "quantized." Theoretically, an infinite number of bits are required to represent each analog sample exactly (Oppenheim and Schafer, 1975). Of course, physical limitations preclude sampling with infinite precision. Therefore, each sample must be either truncated or rounded to fit a finite-length register.

Oppenheim and Schafer (1975) point out that the signal-tonoise ratio (SNR) in decibels (dB) can be approximated by the following equation:

$$SNR = 6N$$

where

N = number of bits in the A/D register (DAR).

The 2920 signal processor has a programmable A/D conversion of 9 bits resolution, giving the device 54 dB of instantaneous dynamic range. With a maximum analog voltage input of ±1 volt, the 2920 signal processor has a voltage resolution of 3.91 millivolts.

DESIGN PROCESS

Filter Specification

Since the major frequencies of interest in an evoked potential lie in the 1 to 50 Hz range, a 0 to 100 Hz passband was chosen. The required Nyquist sampling frequency must then be greater than 200 Hz. The derived filter programs required less than the maximum 192 instructions, but the EPROM was programmed for the full 192 instructions to give a sampling rate of 6510 Hz. This was done for the sake of convenience.

The type of filter chosen was a Butterworth filter, which is characterized by a maximally flat passband (Budak, 1974). A Chebyshev, Elliptic, or Bessel filter would have worked just as well if their passbands were optimized.

Fifth-Order Filter Design

For a 3 dB passband, the squared magnitude function for a Butterworth filter is given by Budak (1974) as

$$|G(s)|^2 = H(s) = \frac{1}{1 + (w/w_3)^{2n}}$$

where

n = order

w_3 = cutoff radian frequency.

As described by Budak (1974), a fifth-order, 3 dB passband, 1 rad/s cutoff transfer function in cascaded form is

$$H(s) = H_1(s)H_2(s)H_3(s)$$

where

$$H_1(s) = \frac{1}{(s + 1)}$$

$$H_2(s) = \frac{1}{s^2 + 0.618s + 1}$$

$$H_3(s) = \frac{1}{s^2 + 1.618s + 1}.$$

The cascaded form is a series representation where the output of an individual section becomes the input to the next section. The significance of the preceding result is all of the decomposition can take place on the analog function, for which tabulated information is given by Stanley (1975), and for which the manipulative steps of decomposition are usually easier.

As shown by Stanley (1975), the standard form of the individual analog section is

$$H(s) = \frac{B_0 + B_1 s + B_2 s^2 + \ldots + B_k s^k}{A_0 + A_1 s + A_2 s^2 + \ldots + A_k s^k}.$$

The bilinear transformation is then applied to the individual section and the discrete transfer function of the digital filter will be of the form (Oppenheim and Schafer, 1975)

$$H(z) = \frac{\Sigma_{k=0}^{N} b_k z^{-k}}{1 - \Sigma_{k=1}^{N} a_k z^{-k}}.$$

The H(z) transfer function is a bilinear transformation of H(s) with the correction factor taken into account. All the coefficients in H(z) were calculated by the formulas given by Stanley (1975). Consequently, a fifth-order Butterworth z-transform filter is

$$H(z) = H_1(z) H_2(z) H_3(z)$$

where

$$H_1(z) = \frac{b_{10} + b_{11} z^{-1}}{1 - a_{11} z^{-1}}$$

$$H_2(z) = \frac{b_{20} + b_{21} z^{-1} + b_{22} z^{-2}}{1 - (a_{21} z^{-1} + a_{22} z^{-2})}$$

$$H_3(z) = \frac{b_{30} + b_{31} z^{-1} + b_{32} z^{-2}}{1 - (a_{31} z^{-1} + a_{32} z^{-2})}$$

where

$a_{11} = 0.9079$	$b_{10} = 4.6068\times10^{-2}$	$b_{30} = 2.1585\times10^{-3}$
$a_{21} = 1.9331$	$b_{11} = b_{10}$	$b_{31} = 2b_{30}$
$a_{22} = -0.9422$	$b_{20} = 2.2594\times10^{-3}$	$b_{32} = b_{30}$
$a_{31} = 1.8467$	$b_{21} = 2b_{20}$	
$a_{32} = -0.8554$	$b_{22} = b_{20}$	

Then, by inspection the difference equations can be determined for each function. The output of section one will become the input to section two and the output of section two will become the input to section three.

<u>First Section</u>

$$y_1(n) = a_{11}y(n-1) + b_{10}x_1(n) + b_{11}x_1(n-1)$$

where

$$x_2(n) = y_1(n)$$

<u>Second Section</u>

$$y_2(n) = a_{21}y_2(n-1) + a_{22}y_2(n-2) + b_{20}x_2(n) + b_{21}x_2(n-1) + b_{22}x_2(n-2)$$

where

$$x_3(n) = y_2(n)$$

<u>Third Section</u>

$$y_3(n) = a_{31}y_3(n-1) + a_{32}y_3(n-2) + b_{30}x_3(n) + b_{31}x_3(n-1) + b_{32}x_3(n-2)$$

where

$$y_3(n) = \text{Filtered sample}$$

Tenth-Order Filter Design

The derivation of the tenth-order digital filter follows the same procedure as the fifth-order filter. As given by Budak (1974), a tenth-order 3 dB passband, 1 rad/s cutoff transfer function in a cascade form is

$$H(s)= H_1(s)H_2(s)H_3(s)H_4(s)H_5(s)$$

where

$$H_1(s) = \frac{1}{s^2 + 0.3128s + 1}$$

$$H_2(s) = \frac{1}{s^2 + 0.9080s + 1}$$

$$H_3(s) = \frac{1}{s^2 + 1.4142s + 1}$$

$$H_4(s) = \frac{1}{s^2 + 1.7820s + 1}$$

$$H_5(s) = \frac{1}{s^2 + 1.9754s + 1}.$$

The tenth-order Butterworth z-transform filter is

$$H(z)= H_1(z)H_2(z)H_3(z)H_4(z)H_5(z)$$

where

$$H_n(z) = \frac{b_{n0} + b_{n1}z^{-1} + b_{n2}z^{-2}}{1 - (a_{n1}z^{-1} + a_{n2}z^{-2})}$$

and

$$n = 1,2,3,4,5$$

and

$a_{11} = 1.9611$	$b_{10} = 2.2922\text{x}10^{-3}$	$b_{50} = 2.1245\text{x}10^{-3}$
$a_{12} = -0.9703$	$b_{11} = 2b_{10}$	$b_{51} = 2b_{50}$
$a_{21} = 1.9073$	$b_{12} = b_{10}$	$b_{52} = b_{50}$
$a_{22} = -0.9162$	$b_{20} = 2.2292\text{x}10^{-3}$	
$a_{31} = 1.8637$	$b_{21} = 2b_{20}$	
$a_{32} = -0.8724$	$b_{22} = b_{20}$	
$a_{41} = 1.8333$	$b_{30} = 2.1783\text{x}10^{-3}$	
$a_{42} = -0.8419$	$b_{31} = 2b_{30}$	
$a_{51} = 1.8177$	$b_{32} = b_{30}$	
$a_{52} = -0.8262$	$b_{40} = 2.1428\text{x}10^{-3}$	
	$b_{41} = 2b_{40}$	
	$b_{42} = b_{40}$	

The difference equations are:

<u>First Section</u>

$$y_1(n) = a_{11}y_1(n-1) + a_{12}y_1(n-2) + b_{10}x_1(n) + b_{11}x_1(n-1) + b_{12}x_1(n-2)$$

where

$$x_2(n) = y_1(n)$$

<u>Second Section</u>

$$y_2(n) = a_{21}y_2(n-1) + a_{22}y_2(n-2) + b_{20}x_2(n) + b_{21}x_2(n-1) + b_{22}x_2(n-2)$$

where

$$x_3(n) = y_2(n)$$

<u>Third Section</u>

$$y_3(n) = a_{31}y_3(n-1) + a_{32}y_3(n-2) + b_{30}x_3(n) + b_{31}x_3(n-1) + b_{32}x_3(n-2)$$

where

$$x_4(n) = y_3(n)$$

<u>Fourth Section</u>

$$y_4(n) = a_{41}y_4(n-1) + a_{42}y_4(n-2) + b_{40}x_4(n) + b_{41}x_4(n-1) + b_{42}x_4(n-2)$$

where

$$x_5(n) = y_4(n)$$

<u>Fifth Section</u>

$$y_5(n) = a_{51}y_5(n-1) + a_{52}y_5(n-2) + b_{50}x_5(n) + b_{51}x_5(n-1) + b_{52}x_5(n-2)$$

where

$$y_5(n) = \text{Filtered sample}$$

Software Development Tools

To program the 2920 signal processor for a specific function, the 2920 software support package (Intel Corporation, 1979, Intel Corporation, 1980a, Intel Corporation, 1980b), was used on the Intel Intellec 800 microcomputer development system (Intel Corporation, 1975). The operating system used was the Intel ISIS-II (Intel Corporation, 1981b). The combined hardware and software tools take a design from concept to implementation. With these tools, the designer can implement an entire design, including testing and debugging, before the 2920 signal processor is used in the actual application.

2920 Assembler

The 2920 Assembler (Intel Corporation, 1979) translates symbolic assembly language programs into machine-readable code. The assembler generates a listing file and an object code file (in hexadecimal). A listing file lists the source code with the corresponding machine code and memory locations. It also includes comments and titles (if any), error and warning diagnostics, the number of RAM and ROM locations used, and a table of user-defined symbols. A listing of the fifth-order Butterworth filter is shown in Appendix A and the tenth-order is shown in Appendix B. The object code files, also included, contain machine-readable code used to program the 2920 processor for real-time testing or for the use with the 2920 Simulator (Intel Corporation, 1980b), for off-line debugging.

2920 Software/Compiler

The SPAS20 Signal Processing Applications Software/Compiler accepts high-level (English-like) language input and produces 2920 assembly language code (Intel Corporation, 1980a). The SPAS20 Compiler gives the designer substantial interactive manipulation of parameters and constraints, both in design of filter stages and in optimization of code size and error limits.

Two main functions of the SPAS20 Compiler are:

1. to make it easy to specify, alter, and review design parameters, and
2. to save time in writing the detailed steps required to implement the necessary functions in assembly language code.

A macro is a labeled block of commands, executed in sequence when the macro name is used as a command. The block of commands is also referred to as the macro body. The SPAS20 Compiler provides a number of macros, which include the Butterworth filter and the bilinear transform. Parameters specified when invoking the Butterworth macro are:

1. order of the filter,
2. 3 dB cutoff frequency in hertz, and
3. a label to uniquely identify a pole.

The output of the Butterworth macro is the poles specified in the s-domain.

Next, the bilinear transform macro is used to transform the poles in the s-domain to poles in the z-domain based on a predefined sampling rate. Once all the poles have been transformed into the z-domain, the final step is to produce the 2920 assembly language instructions on a per-pole basis. In other words, each pole represents an individual filter section cascaded with the other filter sections to produce the required digital filter.

The command CODE is used to produce the required 2920 instructions for each pole section. An error parameter is imposed on the CODE command to minimize movement of the pole from the original coordinates. This constraint was equal to ±0.0001. Based on this constraint, the SPAS20 Compiler derives the necessary instructions to minimize the specified movement. This position error parameter is needed because the assembly language program generated by the CODE command implements a filter stage corresponding to a pole that has a slightly different location than the specified original pole. This difference is due to a finite word length (25 bits). After the compilation has been performed, the pole is moved to the location matching the code generated.

Once all the poles have been coded, the design is now ready to be assembled by the 2920 Assembler (Intel Corporation, 1979). Appendix C shows an example of the use of the SPAS20 Compiler in generating a fifth-order Butterworth filter. For additional details, refer to Intel Corporation (1980a).

2920 Simulator

The 2920 Simulator simulates the program as it would be executed in the 2920 integrated circuit. The simulator allows the designer access to registers, inputs and outputs, clock, memory locations and other points of interest in the signal processor when the performance and problems of the program are analyzed. Although the simulator simulates the functions of the 2920 signal processor, the simulator executes much more slowly than the 2920 processor. The 2920 processor is a real-time signal processor, but the simulator is a logical time signal processor.

Since a digital filter algorithm involves arithmetic operations, proper scaling must be implemented to prevent overflow. Three pins on the 2920 processor can be used for hardware debugging:

1. end of program (EOP),
2. overflow (OF), and
3. the instruction cycle (CCLK).

With the aid of the simulator, the designer can determine the instruction at which a register overflows.

Appendix D presents a demonstration of the 2920 Simulator. Initially, the designer directs the simulator to load the program to be simulated from disk to RAM. The next command, TPROG sets the sample period. In this case, the sample period is 153.6 microseconds (192 instructions multiplied by 800 nanoseconds instruction cycle time).

The TRACE command specifies the items to be traced. TIME, INO (the last character of the preceding item is a zero), and OUTO (the last character of the preceding item is a zero) are the items to be traced. The TIME increment is equal to the sample period or one pass through the program (TPROG). Next, the QUALIFIER parameter instructs the simulator when to collect a trace. For PC=0 (program counter = zero), a trace will be executed for every pass through the program. SIZE is set equal to the total number of 2920 instructions. During the actual simulation of the program, a break in simulation may be specified by using the BREAKPOINT command. The example shown in Appendix D indicates a break on an overflow condition (OVF=1) or when the program counter equals zero (PC=0) and when TIME is greater than or equal to 100 passes through the program (TIME≥100*TPROG).

Another important capability of the 2920 Simulator is its input facility, which enables the designer to specify input functions on any or all of the four, 2920 multiplexed, input pins. Therefore, the designer can input test signals as an analog designer would to measure the system's performance.

Whenever the simulator encounters the 2920 instruction "INO" (INO, means to sample channel zero), it evaluates a function using the current values for the variables used. In addition, the following symbolic constants can appear in the function:

1. PI = 3.14

2. HPI = PI/2

3. TPI = 2*PI

4. ONE = 1.00.

The input function specified in Appendix D is

```
INO= sin(100*TPI*TIME)
```

which indicates a sine wave at 100 Hz.

The command CONSOLE OFF directs the simulator not to output the TRACE data to the console during simulation. This command increases the speed of the simulation process, but doesn't affect the results of the simulation. To initiate the simulation of the program, the SIMULATE FROM 0 (zero) command is given. With a sine wave input of 100 Hz, the simulation time is approximately one second per pass of the simulated program. Therefore, there is a maximum of 100 seconds before the SIMULATION TERMINATED response is printed on the console. The end of simulation will occur according to what was specified in the BREAKPOINT command. The break in simulation is to occur if OVF=1 or if 100 passes through the simulated program has been completed.

When the OVF command is entered and the flag is observed to be equal to zero (OVF=0, overflow flag = zero), the break then occurs on the 100th pass through the simulated program. Therefore, there were no overflow errors in the simulated program. If there had been an overflow

(OVF=1), then by entering in the PC command, the simulator will print out the word address of the overflow.

The PRINT ALL command displays the entire TRACE buffer. When the GRAPHICS ON command is specified, each trace item's changing value is displayed as a curve. Since TIME was the first TRACE item specified, the vertical axis is labeled in seconds. For the horizontal axis, the left most value is minus one and the right most value is plus one. This range corresponds to the output limits of the 2920 processor. The graph of ones' and twos' corresponds to the data values of INO and OUTO respectively.

The following commands are a summary of what has just been described to simulate a five-pole, 100 Hz cutoff, Butterworth filter:

```
 1.  LOAD 5PROG.HEX
 2.  TPROG=192*0.0000008
 3.  TRACE=TIME,INO,OUTO
 4.  QUALIFIER=PC=0
 5.  SIZE=192
 6.  BREAKPOINT=OVF=1 OR PC=0 AND TIME≥100*TPROG
 7.  INO=SIN(100*TPI*TIME)
 8.  CONSOLE OFF
 9.  SIMULATE FROM 0
10.  OVF
11.  PRINT ALL
```

12. GRAPHICS ON
13. PRINT ALL

Once simulation is completed and the designer is satisfied with the performance of the program, the next step is to install the program into the EPROM of the 2920 integrated circuit. The 2920 integrated circuit is inserted into the socket of the Intel Universal Prom Programmer (UPP) (Intel Corporation, 1981c). The software package used to install the program into the EPROM of the 2920 signal processor is called the UPM which runs under ISIS-II. Instructions on the use of the UPM are given by Intel Corporation (1981c).

The fully programmed 2920 signal processor is then inserted into the Intel SDK-2920 kit (Intel Corporation, 1981d, Intel Corporation, 1981e), which has the necessary interface circuitry. Figure 4 represents the basic configuration of the 2920 signal processor for real-time applications. Intel Corporation (1981a) suggested using a value of 1000 pf for the input sampling capacitor to yield an offset of less than minus one-half of the least significant bit.

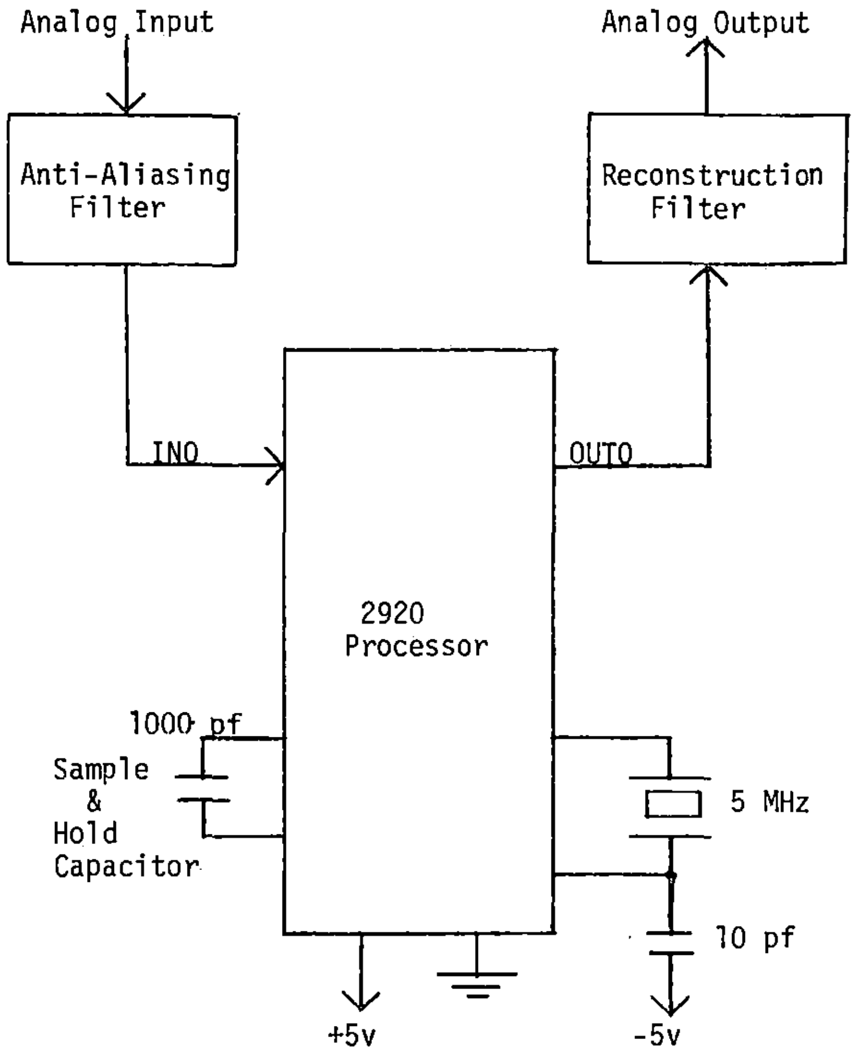


FIGURE 4. Basic configuration of the 2920 signal processor in real-time

COLLECTION OF EVOKED POTENTIALS

A dog anesthetized with halothane was outfitted with scalp electrodes. Cortical electrical activity was amplified with a Grass amplifier (model P511) having a bandpass of 1 to 100 Hz. Gain was set at 1000 and the 60 Hz notch filter was enabled on the Grass amplifier. Averaged visual evoked responses were obtained following 32 presentations of a visual stimulus using an averaging computer (Nicolet model 1072). A strobe light placed 10 cm away, on axis, from the dog's right cornea was illuminated every 500 milliseconds to serve as the visual stimulus. The averaging computer was synchronized to begin sampling at the flash onset and the averaging period had a duration of 256 milliseconds following the flash onset. During the averaging period, 1023 samples were collected in which the time interval between each sample was 250 microseconds. Recordings of the averaged visual evoked responses were then made on a Honeywell 5600 tape recorder.

End-tidal halothane concentrations were measured with a calibrated Beckman LB-2 gas analyzer. The recorded visual evoked responses were performed at 0.7, 1.0, and 2.0 per cent end-tidal halothane concentrations. The visual evoked response recordings were initiated 15 minutes after induction of anesthesia and, thereafter, when halothane had been maintained at a given concentration for at least 2 minutes.

Precautions were taken to prevent the dog's cornea from drying out, since, self-retaining lid retractors had been placed to keep the palpebral fissures open. The recorded waveforms were then amplified by a Threshold Model NS10 Preamplifier to a maximum value of ±1 volt and fed into the input of the 2920 signal processor on the SDK-2920 kit.

RESULTS

The following oscilloscope recordings represent the averaged visual evoked responses at three different concentrations of halothane. The top trace in each recording is the unfiltered waveform and the bottom trace is the filtered response.

Figure 5 represents the five-pole Butterworth filter at a 0.7% concentration of halothane. Figure 6 represents the ten-pole Butterworth filter at a 0.7% concentration of halothane. Figures 7 and 8 are recorded at a faster oscilloscope sweep speed to enhance the definition of the waveform.

Figure 9 represents the five-pole filter at a 1% concentration of halothane. Figure 10 represents the ten-pole filter at a 1% concentration of halothane. The multiple images of the waveform are due to the incorrect triggering of the oscilloscope which was beyond the author's control. Note how the ten-pole filter severely diminishes the amplitude of the evoked potential.

Figure 11 represents the five-pole filter at a 2% concentration of halothane. Figure 12 represents the ten-pole filter at a 2% concentration of halothane. Notice again how the ten-pole filter attenuates the amplitude of the evoked potential.

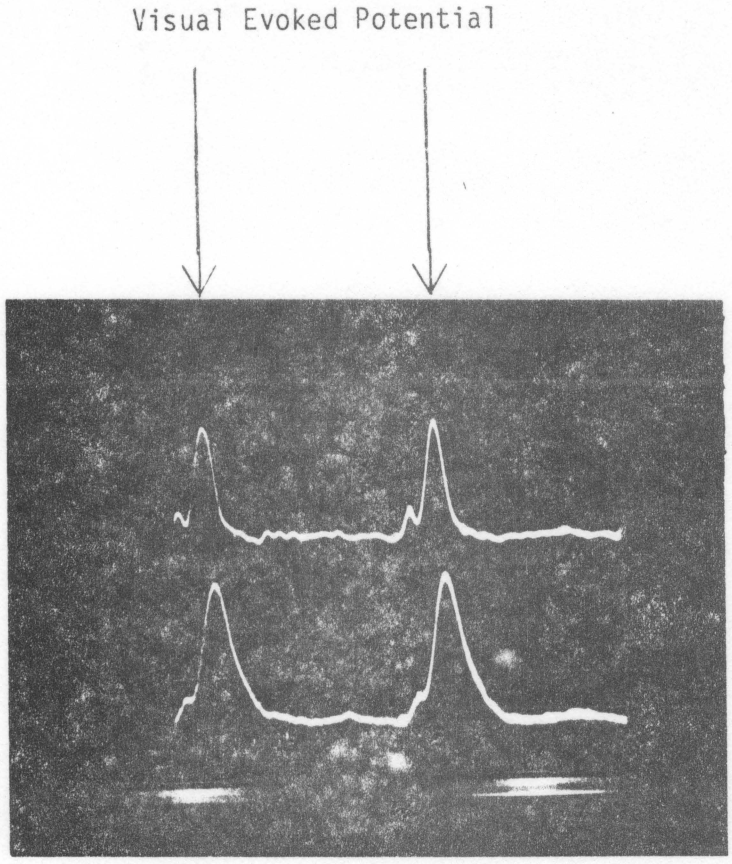


FIGURE 5. Five-pole, 0.7% halothane (vertical axis= .5 v/cm, horizontal axis= 5 msec/cm)

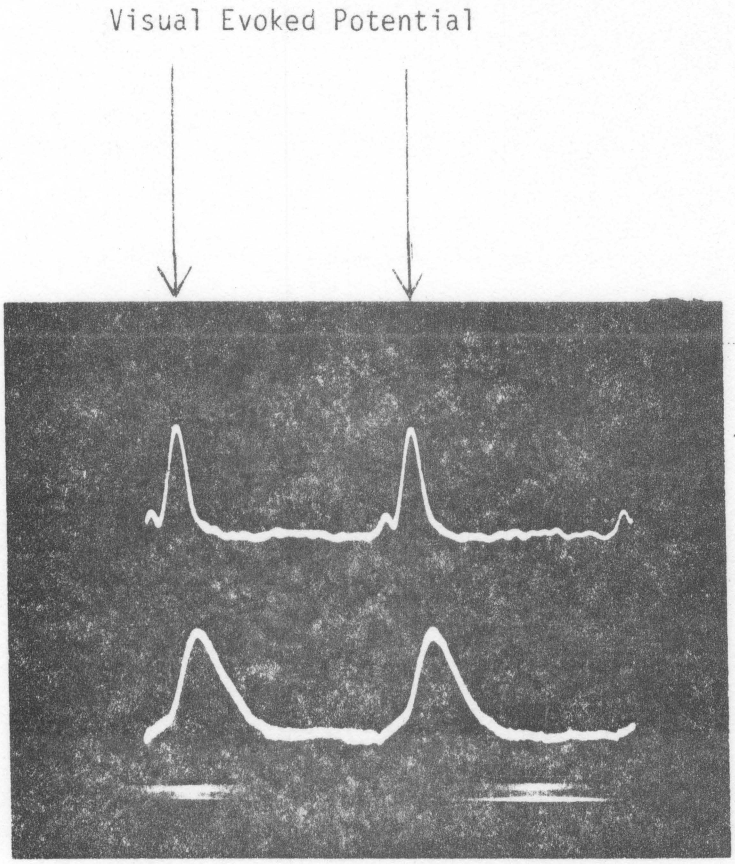


FIGURE 6. Ten-pole, 0.7% halothane (vertical axis= .5 v/cm, horizontal axis= 5 msec/cm)

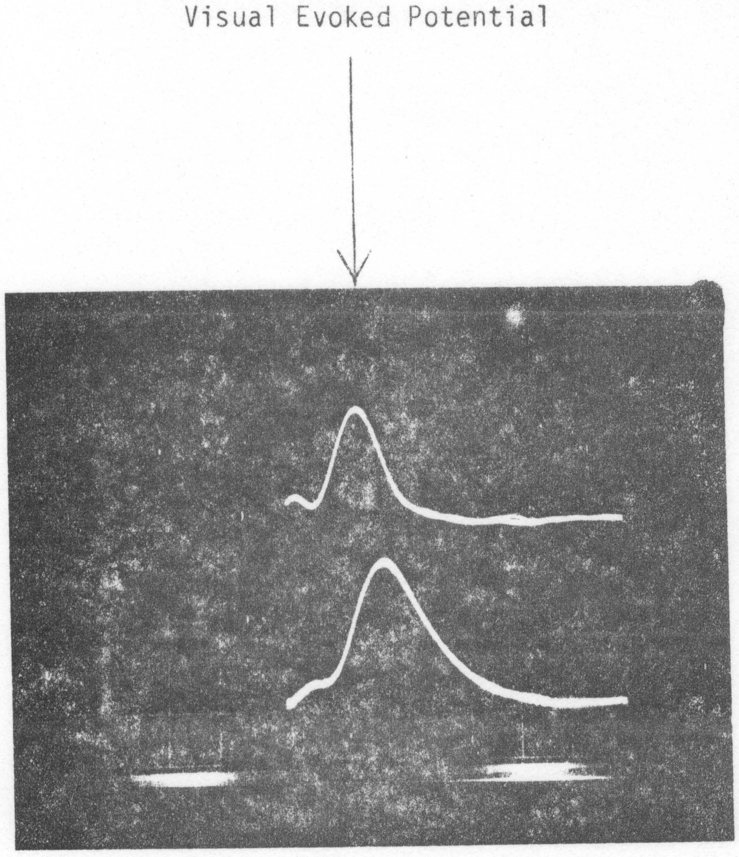


FIGURE 7. Five-pole, 0.7% halothane (vertical axis= .5 v/cm, horizontal axis= 2 msec/cm)

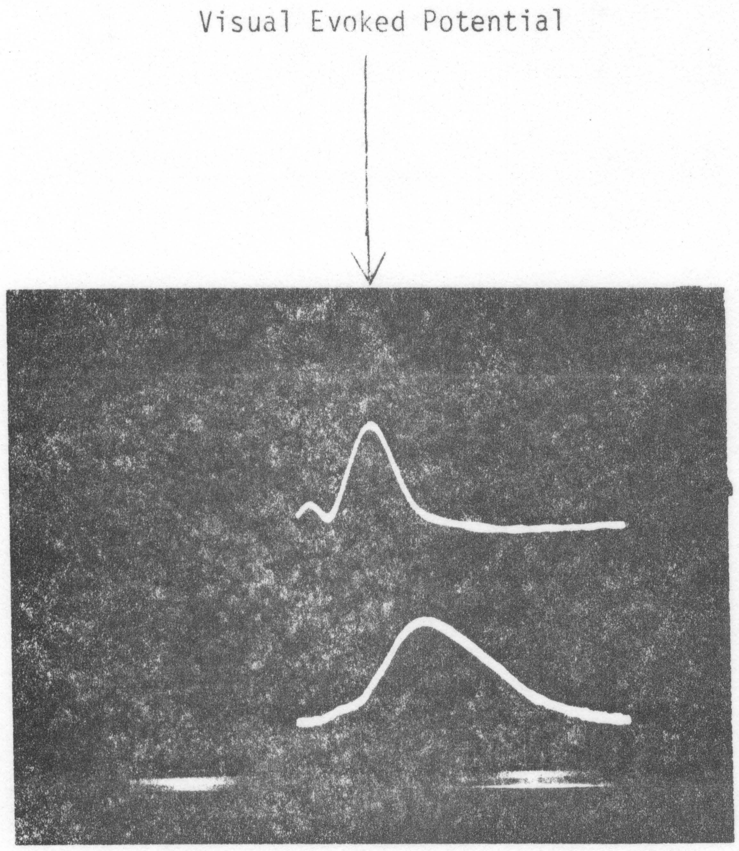


FIGURE 8. Ten-pole, 0.7% halothane (vertical axis= .5 v/cm, horizontal axis= 2 msec/cm)

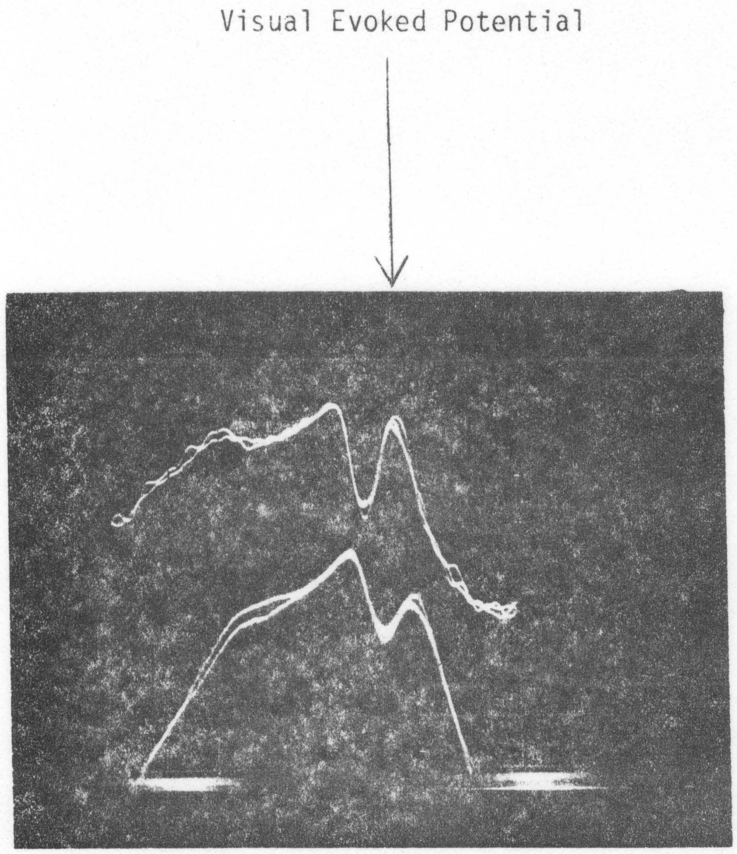


FIGURE 9. Five-pole, 1% halothane (vertical axis= .5 v/cm, horizontal axis= 2 msec/cm)

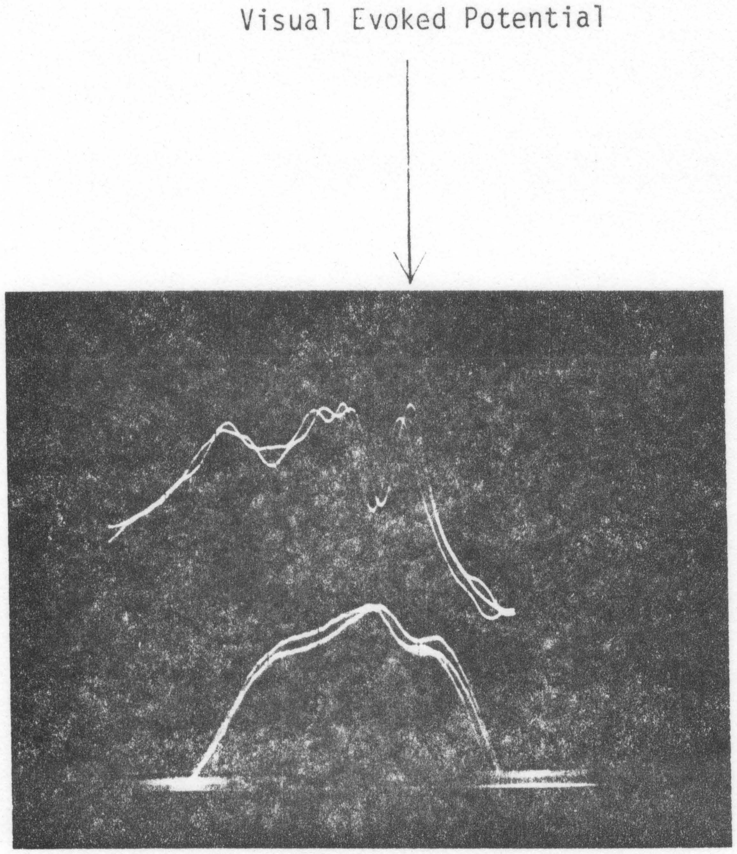


FIGURE 10. Ten-pole, 1% halothane (vertical axis= .5 v/cm, horizontal axis= 2 msec/cm)

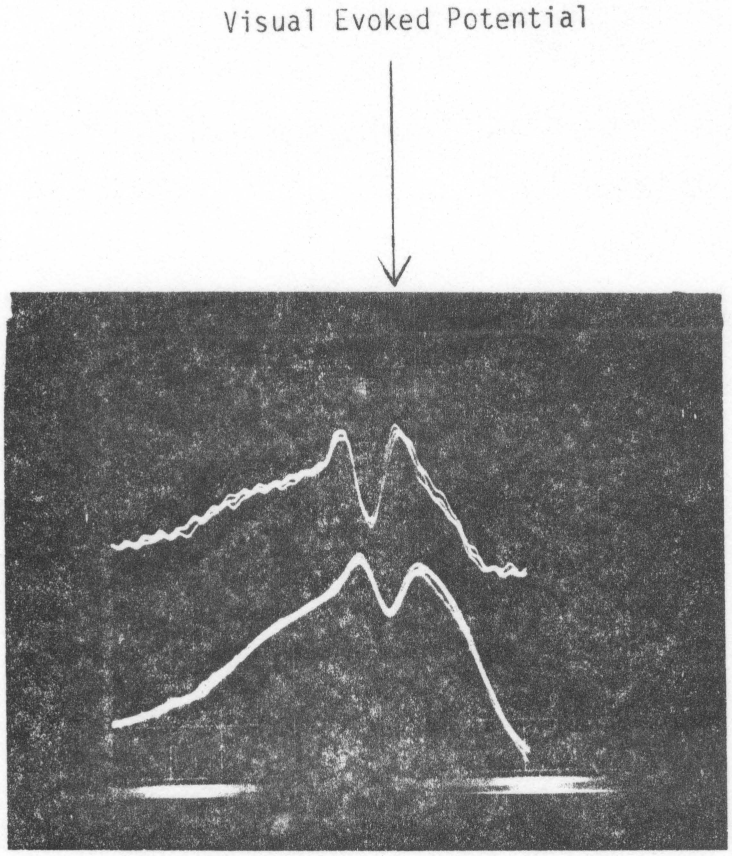


FIGURE 11. Five-pole, 2% halothane (vertical axis= .5 v/cm, horizontal axis= 2 msec/cm)

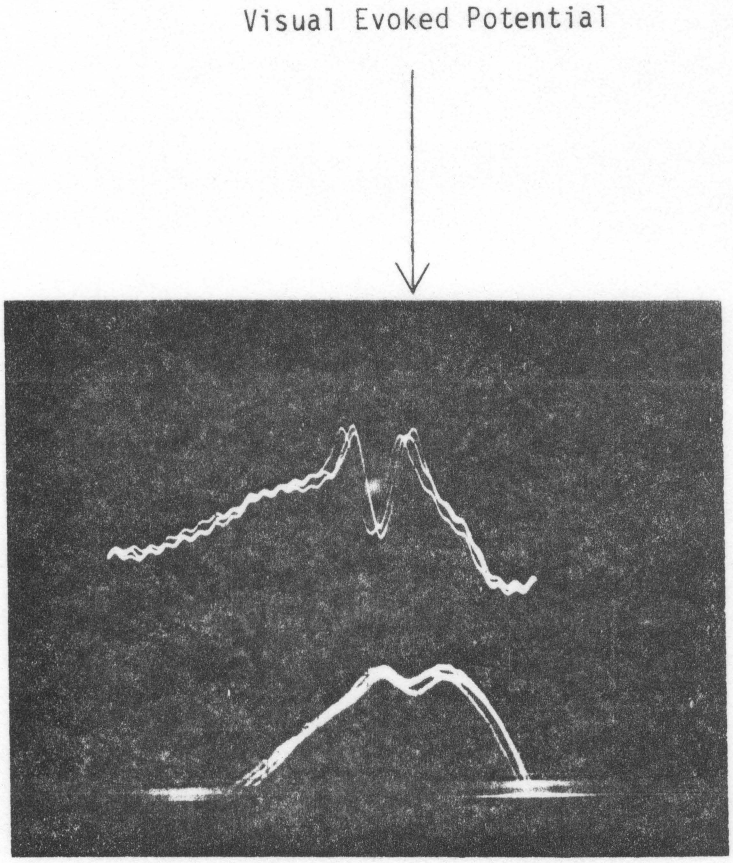
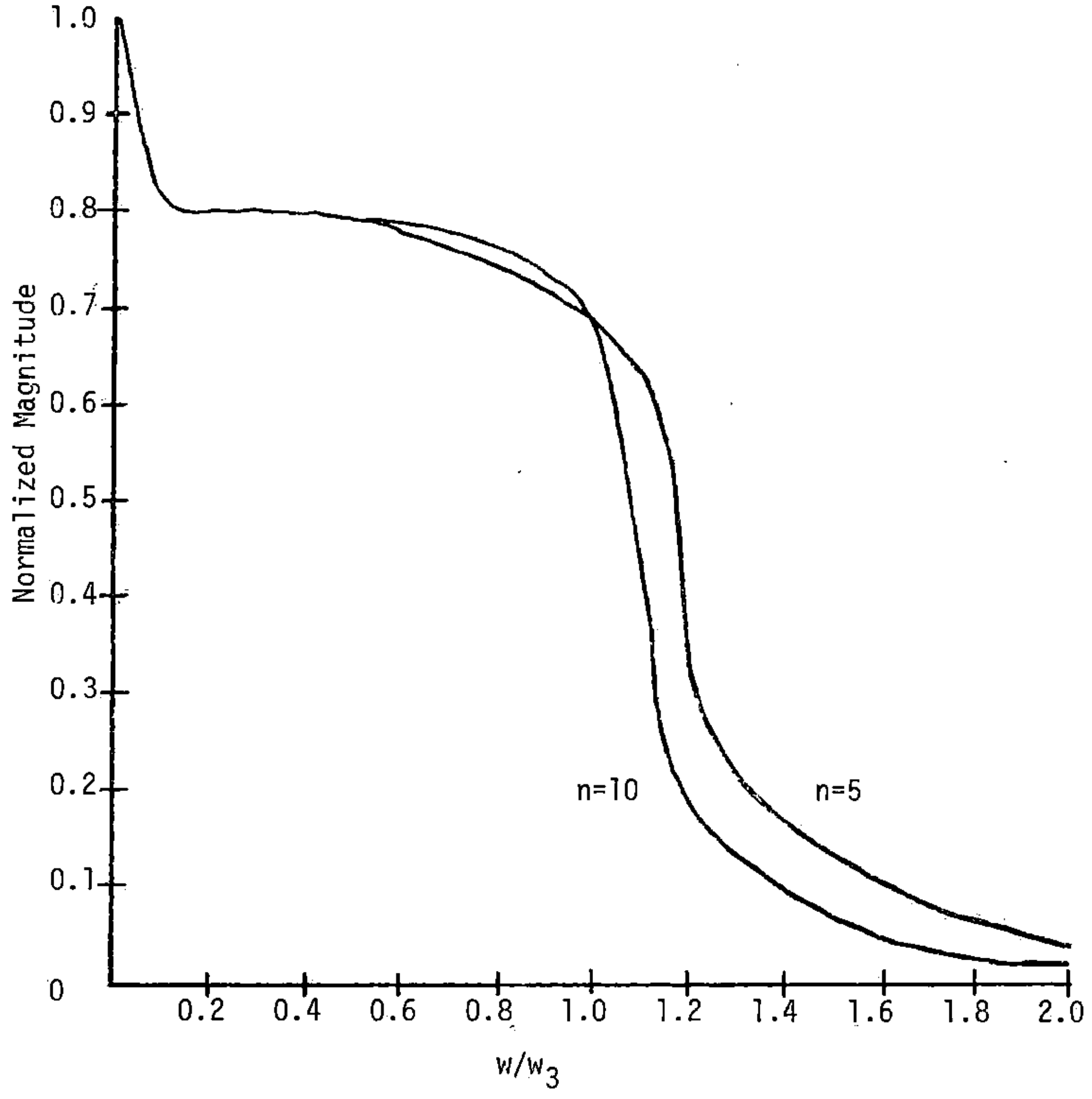


FIGURE 12. Ten-pole, 2% halothane (vertical axis= .5 v/cm, horizontal axis= 2 msec/cm)

The ten-pole filter responses display slow rise times and longer settling times in comparison to the five-pole filter responses. This is characteristic of a higher order Butterworth filter (Johnson, 1976). In comparison, a Bessel filter will have a shorter rise time as the order of the filter increases (Johnson, 1976).

Since, the raw visual evoked potentials were already frequency band-limited at 100 Hz before being processed by the 2920 signal processor, it was necessary to perform frequency and phase studies of the 2920 signal processor. A ±1 volt sine wave generated from an Interstate Electronics Corporation Model F34 Function Generator was fed into the input of the 2920 signal processor. The output signal versus the input signal in terms of normalized magnitude and phase responses, are depicted in Figure 13 and Figure 14 respectively. The magnitude is reasonably flat and has a fairly linear phase response in the passband for both the five pole and ten pole filters.

A Tektronix Model 465 Oscilloscope was used to monitor the input and output ports of the 2920 signal processor. Consequently, the magnitude and phase data were calculated from the waveforms displayed on the cathode ray tube. Measurement of the phase difference was based on the Lissajous method.



n = order of the Butterworth filter
w_3 = cutoff radian frequency

FIGURE 13. Magnitude response

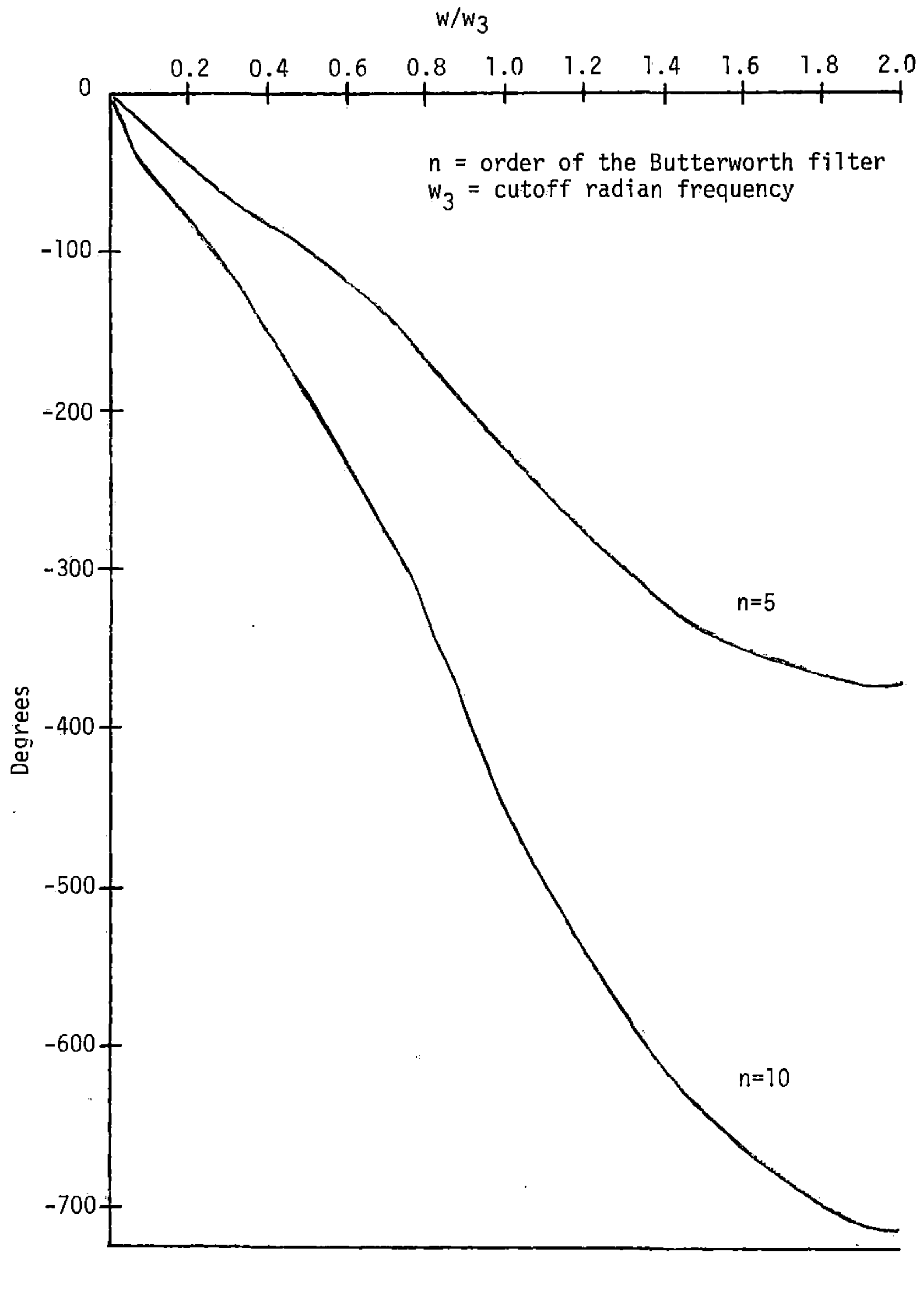


FIGURE 14. Phase response

CONCLUSIONS

The visual evoked potentials at a 0.7% halothane concentration show the filtering and analog-to-digital actions of the 2920 signal processor. However, pictorially, the visual evoked potentials at 1% and 2% halothane concentrations portray multiple images due to incorrect triggering of the oscilloscope. From a morphological point of view, the 2920 signal processor performs correctly in filtering the visual evoked potentials at a 0.7% halothane concentration. It clearly shows the slow rise times and longer settling times that are associated with higher order Butterworth filters.

The ten-pole filter severely diminished the amplitude of the visual evoked potentials at 1% and 2% concentrations of halothane. The magnitude response portrayed in Figure 13 show the amplitude response should be at least 80% of the original signal.

The amplitude responses of the five-pole filter at 1% and 2% halothane concentrations were diminished but not as severe as the responses from the ten-pole filter. However, the magnitude response given in Figure 13 for the five-pole filter show the magnitude should be at least 80% of the original signal. Again, as in the case of the ten-pole filter, in terms of magnitude response, the five-pole filter did not perform as expected. Only in the 0.7% halothane concentration does the 2920 signal processor support the results of the magnitude responses shown in Figure 13.

The magnitude and phase responses depicted in Figure 13 and Figure 14 respectively, clearly show the 2920 signal processor to perform as a Butterworth low-pass filter should (Budak, 1974). On the other hand, the results of the filtering action of the evoked potentials at the 1% and 2% halothane concentrations prove somewhat contradictory.

Given the above mentioned results, the author cannot offer any absolute answers to the perplexity of the findings. More experiments will have to be performed and the results scrutinized to determine if the 2920 signal processor is suitable as a low-pass filter and analog-to-digital converter preprocessor for the microcomputer instrument.

In regards to the 2920 software support package, it is a very powerful tool in developing visual evoked potential filter programs for the 2920 signal processor. The SPAS20 Compiler program provides the necessary commands to specify and develop the low-pass Butterworth filter easily. The 2920 Simulator allows the designer to thoroughly test the program before programming the 2920 signal processor. Also, the simulator has the capability to graph the input and output waveforms versus time on a hard copy device.

BIBLIOGRAPHY

Ackmann, J. J., P. P. Elko, and S. J. Wu. 1979. Digital filtering of on-line evoked potentials. International Journal of Bio-Medical Computing 10:291-303.

Agarwal, P., R. Priemer. 1979. Microprocessor based digital signal processing system. Comput. Biol. Med. 9:87-95.

Bennett, J. R., J. S. Macdonald, S. M. Drance, and K. Uenoyama. 1971. Some statistical properties of the visual evoked potential in man and their application as a criterion of normality. I.E.E.E. Transactions on Bio-Medical Engineering BME-18(1):23-34.

Budak, A. 1974. Passive and active network analysis and synthesis. Houghton Mifflin Company, Boston, MA.

Desmedt, J. E., E. Brunko, J. Debecker, and J. Carmeliet. 1974. The system bandpass required to avoid distortion of early components when averaging somatosensory evoked potentials. Electroencephalography and Clinical Neurophysiology 37:407-410.

Intel Corporation. 1975. Intellec 800 microcomputer development system operator's manual. Number 9800129A. Santa Clara, California.

Intel Corporation. 1979. 2920 assembly language manual. Number 9800987-01. Santa Clara, California.

Intel Corporation. 1980a. 2920 signal processing applications software/compiler user's guide. Number 121529-002, Revision B. Santa Clara, California.

Intel Corporation. 1980b. 2920 simulator user's guide. Number 9800988-02, Revision B. Santa Clara, California.

Intel Corporation. 1981a. Component data catalog. Pages 4-45 to 4-55. Santa Clara, California.

Intel Corporation. 1981b. ISIS-II user's guide. Number 9800306-06. Santa Clara, California.

Intel Corporation. 1981c. Universal prom programmer user's manual. Number 9800819-03. Santa Clara, California.

Intel Corporation. 1981d. SDK-2920 system design kit assembly manual. Number 162421-002. Santa Clara, California.

Intel Corporation. 1981e. SDK-2920 system design kit user's guide. Number 162418-001, Revision A. Santa Clara, California.

Johnson, D. E. 1976. Introduction to filter theory. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Oppenheim, A. V. and R. W. Schafer. 1975. Digital signal analysis. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

Perry, N. W. and D. G. Childers. 1969. The human visual evoked response: method and theory. Charles C. Thomas, Publisher, Springfield, Illinois.

Stanley, W. D. 1975. Digital signal processing. Reston Publishing Company, Inc., Reston, Virginia.

Uhl, R. R., K. C. Squires, D. L. Bruce, and A. Starr. 1980. Effect of halothane anesthesia on the human cortical visual evoked response. Anesthesiology 53:273-276.

Walstrom, R. W. 1979. A microprocessor controlled digital filter. Ph.D. Thesis. Iowa State University, Ames, IA.

APPENDIX A: FIFTH-ORDER BUTTERWORTH FILTER LISTING

```
ISIS-II 2920 ASSEMBLER V1.0

ASSEMBLER INVOKED BY: :F1:AS2920 5PROG.BUT

  LINE   LOC OBJECT SOURCE STATEMENT

     1              ;
     2              ;A/D CONVERSION ROUTINE
     3              ;
     4     0 40C6EF     LDA DAR,KP0 ;CLEAR DAR
     5     1 0000EF     IN0
     6     2 0000EF     IN0
     7     3 0000EF         IN0
     8     4 0000EF         IN0
     9     5 0000EF         IN0
    10     6 0000EF         IN0
    11     7 0000EF         IN0
    12     8 0000EF         IN0
    13     9 4000EF         NOP
    14    10 6000EF         CVTS
    15    11 4000EF         NOP
    16    12 4000EF         NOP
    17    13 7100EF         CVT7
    18    14 4000EF         NOP
    19    15 4000EF         NOP
    20    16 6100EF         CVT6
    21    17 4000EF         NOP
    22    18 4000EF         NOP
    23    19 5100EF         CVT5
    24    20 4000EF         NOP
    25    21 4000EF         NOP
    26    22 4100EF         CVT4
    27    23 4000EF         NOP
    28    24 4000EF         NOP
    29    25 3100EF         CVT3
    30    26 4000EF         NOP
    31    27 4000EF         NOP
    32    28 2100EF         CVT2
    33    29 4000EF         NOP
    34    30 4000EF         NOP
    35    31 1100EF         CVT1
    36    32 4000EF         NOP
    37    33 4000EF         NOP
    38    34 0100EF         CVT0
    39    35 4000EF         NOP
    40              ;
    41              ;
```

```
                 ;5TH ORDER BUTTERWORTH FILTER
                 ;100 HZ CUTOFF
                 ;
  36 40222E         LDA INO_P10,DAR,R2
                 ;
                 ;
                 OUT2_P10 EQU TEMP
  37 4200FF LDA OUT2_P10,OUT1_P10,R00
                    ; OUT2_P10=1.00000000*OUT1_P10
  38 4608EF LDA OUT1_P10,OUT0_P10,R00
                    ; OUT1_P10=1.00000000*OUT0_P10
  39 44087E LDA OUT0_P10,OUT2_P10,R04
                    ; OUT0_P10=0.062500000*OUT2_P10
  40 44083B SUB OUT0_P10,OUT2_P10,R10
                    ; OUT0_P10=0.061523437*OUT2_P10
  41 4408FB SUB OUT0_P10,OUT2_P10,R00
                    ; OUT0_P10=-0.93847656*OUT2_P10
  42 4608FC ADD OUT0_P10,OUT0_P10,R08
                    ; OUT0_P10=-0.94214248*OUT2_P10
  43 4600DD ADD OUT0_P10,OUT1_P10,L01
                    ; OUT0_P10=2.0000000
;*OUT1_P10-0.94214248*OUT2_P10
  44 46007A SUB OUT0_P10,OUT1_P10,R04
                    ; OUT0_P10=1.9375000
;*OUT1_P10-0.94214248*OUT2_P10
  45 4600FA SUB OUT0_P10,OUT1_P10,R08
                    ; OUT0_P10=1.9335937
;*OUT1_P10-0.94214248*OUT2_P10
  46 46005B SUB OUT0_P10,OUT1_P10,R11
                    ; OUT0_P10=1.9331054
;*OUT1_P10-0.94214248*OUT2_P10
  47 4400FD ADD OUT0_P10,INO_P10,R00
                    ; OUT0_P10=1.9331054
;*OUT1_P10-0.94214248
;*OUT2_P10+1.00000000*INO_P10
                 ;
                 ;
                 INO_P11 EQU OUT0_P10
                 ;
  48 46083E         LDA INO_P11,OUT0_P10,R2
;SHIFT INPUT TO POLE 11 BY 2
                 ;
                 ;
                 OUT2_P11 EQU TEMP
  49 4800FF LDA OUT2_P11,OUT1_P11,R00
                    ; OUT2_P11=1.00000000*OUT1_P11
  50 4818EF LDA OUT1_P11,OUT0_P11,R00
                    ; OUT1_P11=1.00000000*OUT0_P11
  51 40185E LDA OUT0_P11,OUT2_P11,R03
                    ; OUT0_P11=0.12500000*OUT2_P11
```

```
 52 4018DA SUB OUT0_P11,OUT2_P11,R07
               ; OUT0_P11=0.117187500*OUT2_P11
 53 48189D ADD OUT0_P11,OUT0_P11,R13
               ; OUT0_P11=0.117201806*OUT2_P11
 54 4018FB SUB OUT0_P11,OUT2_P11,R00
               ; OUT0_P11=-0.88279824*OUT2_P11
 55 48189A SUB OUT0_P11,OUT0_P11,R05
               ; OUT0_P11=-0.85521074*OUT2_P11
 56 4810DD ADD OUT0_P11,OUT1_P11,L01
               ; OUT0_P11=2.0000000
;*OUT1_P11-0.85521074*OUT2_P11
 57 48105A SUB OUT0_P11,OUT1_P11,R03
               ; OUT0_P11=1.8750000
;*OUT1_P11-0.85521074*OUT2_P11
 58 48109A SUB OUT0_P11,OUT1_P11,R05
               ; OUT0_P11=1.8437500
;*OUT1_P11-0.85521074*OUT2_P11
 59 48101D ADD OUT0_P11,OUT1_P11,R09
               ; OUT0_P11=1.8457031
;*OUT1_P11-0.85521074*OUT2_P11
 60 48103D ADD OUT0_P11,OUT1_P11,R10
               ; OUT0_P11=1.8466796
;*OUT1_P11-0.85521074*OUT2_P11
 61 48109B SUB OUT0_P11,OUT1_P11,R13
               ; OUT0_P11=1.8465576
;*OUT1_P11-0.85521074*OUT2_P11
 62 4218FD ADD OUT0_P11,IN0_P11,R00
               ; OUT0_P11=1.8465576
;*OUT1_P11-0.85521074
;*OUT2_P11+1.00000000*IN0_P11
           ;
           ;
           IN0_P12 EQU OUT0_P11
 63 48181E     LDA IN0_P12,OUT0_P11,R01
;RIGHT SHIFT INPUT TO POLE 12 BY 1
           ;
           ;
           ;
           OUT1_P12 EQU TEMP
 64 4A00FF LDA OUT1_P12,OUT0_P12,R00
               ; OUT1_P12=1.00000000*OUT0_P12
 65 44186A SUB OUT0_P12,OUT1_P12,R04
               ; OUT0_P12=1.00000000
;*OUT0_P12-0.062500000*OUT1_P12
 66 4E108A SUB OUT0_P12,OUT0_P12,R05
               ; OUT0_P12=0.96875000
;*OUT0_P12-0.060546875*OUT1_P12
 67 4E104B SUB OUT0_P12,OUT0_P12,R11
               ; OUT0_P12=0.96827695
;*OUT0_P12-0.060517309*OUT1_P12
 68 4C18ED ADD OUT0_P12,IN0_P12,R00
```

```
                        ; OUTO_P12=0.96827695
;*OUTO_P12-0.060517309
;*OUT1_P12+1.00000000*INO_P12
                    ;
                    ;
                    ;OUTPUT SEQUENCE :
                    ;
         69 4A44EF      LDA DAR,OUTO_P12
;STUFF FINAL VALUE INTO DAR.
                    ;
                    ;OUTPUT DAR.
                    ;
         70 4000EF      NOP            ;SETTLE D/A CONVERTER.
         71 4000EF      NOP
         72 4000EF      NOP
         73 4000EF      NOP
                    ;
                    ;
         74 8000EF      OUTO               ;MININUM OF 3.5/10**6
         75 8000EF      OUTO               ;TO SETTLE S&H OUTPUT.
         76 8000EF      OUTO               ; 800/10**9 SEC CYCLE
         77 8000EF      OUTO
         78 8000EF      OUTO
                    ;
                    ;
         79 9000EF      OUT1
         80 9000EF      OUT1
         81 9000EF      OUT1
         82 9000EF      OUT1
         83 9000EF      OUT1
                    ;
                    ;
         84 A000EF      OUT2
         85 A000EF      OUT2
         86 A000EF      OUT2
         87 A000EF      OUT2
         88 A000EF      OUT2
                    ;
                    ;
         89 B000EF      OUT3
         90 B000EF      OUT3
         91 B000EF      OUT3
         92 B000EF      OUT3
         93 B000EF      OUT3
                    ;
                    ;
         94 C000EF      OUT4
         95 C000EF      OUT4
         96 C000EF      OUT4
         97 C000EF      OUT4
```

```
 98 C000EF       OUT4
              ;
              ;
 99 D000EF       OUT5
100 D000EF       OUT5
101 D000EF       OUT5
102 D000EF       OUT5
103 D000EF       OUT5
              ;
              ;
104 E000EF       OUT6
105 E000EF       OUT6
106 E000EF       OUT6
107 E000EF       OUT6
108 E000EF       OUT6
              ;
              ;
109 F000EF       OUT7
110 F000EF       OUT7
111 F000EF       OUT7
112 F000EF       OUT7
113 F000EF       OUT7
              ;
              ;
114 4000EF       NOP
115 4000EF       NOP
116 4000EF       NOP
117 4000EF       NOP
118 4000EF       NOP
119 4000EF       NOP
120 4000EF       NOP
121 4000EF       NOP
122 4000EF       NOP
123 4000EF       NOP
124 4000EF       NOP
125 4000EF       NOP
126 4000EF       NOP
127 4000EF       NOP
128 4000EF       NOP
129 4000EF       NOP
130 4000EF       NOP
131 4000EF       NOP
132 4000EF       NOP
133 4000EF       NOP
134 4000EF       NOP
135 4000EF       NOP
136 4000EF       NOP
137 4000EF       NOP
138 4000EF       NOP
139 4000EF       NOP
```

```
221    140  4000EF          NOP
222    141  4000EF          NOP
223    142  4000EF          NOP
224    143  4000EF          NOP
225    144  4000EF          NOP
226    145  4000EF          NOP
227    146  4000EF          NOP
228    147  4000EF          NOP
229    148  4000EF          NOP
230    149  4000EF          NOP
231    150  4000EF          NOP
232    151  4000EF          NOP
233    152  4000EF          NOP
234    153  4000EF          NOP
235    154  4000EF          NOP
236    155  4000EF          NOP
237    156  4000EF          NOP
238    157  4000EF          NOP
239    158  4000EF          NOP
240    159  4000EF          NOP
241    160  4000EF          NOP
242    161  4000EF          NOP
243    162  4000EF          NOP
244    163  4000EF          NOP
245    164  4000EF          NOP
246    165  4000EF          NOP
247    166  4000EF          NOP
248    167  4000EF          NOP
249    168  4000EF          NOP
250    169  4000EF          NOP
251    170  4000EF          NOP
252    171  4000EF          NOP
253    172  4000EF          NOP
254    173  4000EF          NOP
255    174  4000EF          NOP
256    175  4000EF          NOP
257    176  4000EF          NOP
258    177  4000EF          NOP
259    178  4000EF          NOP
260    179  4000EF          NOP
261    180  4000EF          NOP
262    181  4000EF          NOP
263    182  4000EF          NOP
264    183  4000EF          NOP
265    184  4000EF          NOP
266    185  4000EF          NOP
267    186  4000EF          NOP
268    187  4000EF          NOP
```

```
   269   188 5000EF       EOP
   270   189 400QEF       NOP
   271   190 4000EF       NOP
   272   191 4000EF       NOP
PREMATURE EOF


SYMBOL:                                   VALUE:

INO_P10                                         0
OUT2_P10                                        1
TEMP                                            1
OUT1_P10                                        2
OUT0_P10                                        3
INO_P11                                         3
OUT2_P11                                        1
OUT1_P11                                        4
OUT0_P11                                        5
INO_P12                                         5
OUT1_P12                                        1
OUT0_P12                                        6


ASSEMBLY COMPLETE
ERRORS     =    0
WARNINGS   =    0
RAMSIZE    =    7
ROMSIZE    =  192
```

```
:18000000F4F0FCF6FEFFF0F0F0F0FEFFF0F0F0F0FEFFF0F0F0F0FEFFDE
:18001800F0F0F0F0FEFFF0F0F0F0FEFFF0F0F0F0FEFFF0F0F0F0FEFFDC
:18003000F0F0F0F0FEFFF4F0F0F0FEFFF6F0F0F0FEFFF4F0F0F0FEFFB6
:18004800F4F0F0F0FEFFF7F1F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF98
:18006000F6F1F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF5F1F0F0FEFF7F
:18007800F4F0F0F0FEFFF4F0F0F0FEFFF4F1F0F0FEFFF4F0F0F0FEFF6B
:18009000F4F0F0F0FEFFF3F1F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF54
:1800A800F2F1F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF1F1F0F0FEFF3F
:1800C000F4F0F0F0FEFFF4F0F0F0FEFFF0F1F0F0FEFFF4F0F0F0FEFF27
:1800D800F4F0F2F2F2FEF4F2F0F0FFFFF4F6F0F8FEFFF4F4F0F8F7FE00
:1800F000F4F4F0F8F3FBF4F4F0F8FFFBF4F6F0F8FFFCF4F6F0F0FDFDDF
:18010800F4F6F0F0F7FAF4F6F0F0FFFAF4F6F0F0F5FBF4F4F0F0FFFDE3
:18012000F4F6F0F8F3FEF4F8F0F0FFFFF4F8F1F8FEFFF4F0F1F8F5FEA8
:18013800F4F0F1F8FDFAF4F8F1F8F9FDF4F0F1F8FFFBF4F8F1F8F9FA91
:18015000F4F8F1F0FDFDF4F8F1F0F5FAF4F8F1F0F9FAF4F8F1F0F1FD99
:18016800F4F8F1F0F3FDF4F8F1F0F9FBF4F2F1F8FFFDF4F8F1F8F1FE72
:18018000F4FAF0F0FFFFF4F4F1F8F6FAF4FEF1F0F8FAF4FEF1F0F4FB53
:18019800F4FCF1F8FEFDF4FAF4F4FEFFF4F0F0F0FEFFF4F0F0F0FEFF26
:1801B000F4F0F0F0FEFFF4F0F0F0FEFFF8F0F0F0FEFFF8F0F0F0FEFF2B
:1801C800F8F0F0F0FEFFF8F0F0F0FEFFF8F0F0F0FEFFF9F0F0F0FEFF0A
:1801E000F9F0F0F0FEFFF9F0F0F0FEFFF9F0F0F0FEFFF9F0F0F0FEFFEF
:1801F800FAF0F0F0FEFFFAF0F0F0FEFFFAF0F0F0FEFFFAF0F0F0FEFFD3
:18021000FAF0F0F0FEFFFBF0F0F0FEFFFBF0F0F0FEFFFBF0F0F0FEFFB7
:18022800FBF0F0F0FEFFFBF0F0F0FEFFFCF0F0F0FEFFFCF0F0F0FEFF9C
:18024000FCF0F0F0FEFFFCF0F0F0FEFFFCF0F0F0FEFFFDF0F0F0FEFF81
:18025800FDF0F0F0FEFFFDF0F0F0FEFFFDF0F0F0FEFFFDF0F0F0FEFF66
:18027000FEF0F0F0FEFFFEF0F0F0FEFFFEF0F0F0FEFFFEF0F0F0FEFF4A
:18028800FEF0F0F0FEFFFFF0F0F0FEFFFFF0F0F0FEFFFFF0F0F0FEFF2F
:1802A000FFF0F0F0FEFFFFF0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF2C
:1802B800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF2A
:1802D000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF12
:1802E800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFFA
:18030000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFE1
:18031800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFC9
:18033000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFB1
:18034800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF99
:18036000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF81
:18037800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF69
:18039000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF51
:1803A800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF39
:1803C000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF21
:1803D800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF09
:1803F000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF1
:18040800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFD8
:18042000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFC0
:18043800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFA8
:18045000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF90
:18046800F5F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF77
:00000001FF
```

APPENDIX B: TENTH-ORDER BUTTERWORTH FILTER LISTING

```
ISIS-II 2920 ASSEMBLER V1.0

ASSEMBLER INVOKED BY: :F1:AS2920 10PROG.BUT

  LINE  LOC OBJECT SOURCE STATEMENT

     1                 ;
     2                 ;A/D CONVERSION ROUTINE
     3                 ;
     4      0 40C6EF       LDA DAR,KP0 ;CLEAR DAR
     5      1 0000EF       INO
     6      2 0000EF       INO
     7      3 0000EF           INO
     8      4 0000EF           INO
     9      5 0000EF           INO
    10      6 0000EF           INO
    11      7 0000EF           INO
    12      8 0000EF           INO
    13      9 4000EF           NOP
    14     10 6000EF           CVTS
    15     11 4000EF           NOP
    16     12 4000EF           NOP
    17     13 7100EF           CVT7
    18     14 4000EF           NOP
    19     15 4000EF           NOP
    20     16 6100EF           CVT6
    21     17 4000EF           NOP
    22     18 4000EF           NOP
    23     19 5100EF           CVT5
    24     20 4000EF           NOP
    25     21 4000EF           NOP
    26     22 4100EF           CVT4
    27     23 4000EF           NOP
    28     24 4000EF           NOP
    29     25 3100EF           CVT3
    30     26 4000EF           NOP
    31     27 4000EF           NOP
    32     28 2100EF           CVT2
    33     29 4000EF           NOP
    34     30 4000EF           NOP
    35     31 1100EF           CVT1
    36     32 4000EF           NOP
    37     33 4000EF           NOP
    38     34 0100EF           CVT0
    39     35 4000EF           NOP
    40                 ;
```

```
                ;
                ;10TH ORDER BUTTER FILTER
                ;
36 40224E LDA IN0_P10,DAR,R3
                ;
                ;
                OUT2_P10 EQU TEMP
37 4200FF LDA OUT2_P10,OUT1_P10,R00
                     ; OUT2_P10=1.00000000*OUT1_P10
38 4608EF LDA OUT1_P10,OUT0_P10,R00
                     ; OUT1_P10=1.00000000*OUT0_P10
39 4400FF LDA OUT0_P10,IN0_P10,R00
                     ; OUT0_P10=1.00000000*IN0_P10
40 4408FB SUB OUT0_P10,OUT2_P10,R00
                     ; OUT0_P10=-1.00000000
;*OUT2_P10+1.00000000*IN0_P10
41 44089C ADD OUT0_P10,OUT2_P10,R05
                     ; OUT0_P10=-0.96875000
;*OUT2_P10+1.00000000*IN0_P10
42 44081B SUB OUT0_P10,OUT2_P10,R09
                     ; OUT0_P10=-0.97070312
;*OUT2_P10+1.00000000*IN0_P10
43 44085D ADD OUT0_P10,OUT2_P10,R11
                     ; OUT0_P10=-0.97021484
;*OUT2_P10+1.00000000*IN0_P10
44 4600DD ADD OUT0_P10,OUT1_P10,L01
                     ; OUT0_P10=2.0000000
;*OUT1_P10-0.97021484
;*OUT2_P10+1.00000000*IN0_P10
45 46009A SUB OUT0_P10,OUT1_P10,R05
                     ; OUT0_P10=1.9687500
;*OUT1_P10-0.97021484
;*OUT2_P10+1.00000000*IN0_P10
46 4600DA SUB OUT0_P10,OUT1_P10,R07
                     ; OUT0_P10=1.9609375
;*OUT1_P10-0.97021484
;*OUT2_P10+1.00000000*IN0_P10
47 46009D ADD OUT0_P10,OUT1_P10,R13
                     ; OUT0_P10=1.9610595
;*OUT1_P10-0.97021484
;*OUT2_P10+1.00000000*IN0_P10
                ;
                ;
                IN0_P11 EQU OUT0_P10
                ;
48 46081E LDA IN0_P11,OUT0_P10,R1
                ;
                ;
```

```
                  OUT2_P11 EQU TEMP
      49 4800FF LDA OUT2_P11,OUT1_P11,R00
                        ; OUT2_P11=1.00000000*OUT1_P11
      50 4818EF LDA OUT1_P11,OUT0_P11,R00
                        ; OUT1_P11=1.00000000*OUT0_P11
      51 48109A SUB OUT0_P11,OUT1_P11,R05
                        ; OUT0_P11=1.00000000
;*OUT0_P11-0.031250000*OUT1_P11
      52 4818BA SUB OUT0_P11,OUT0_P11,R06
                        ; OUT0_P11=0.98437500
;*OUT0_P11-0.030761718*OUT1_P11
      53 4818FD ADD OUT0_P11,OUT0_P11,R00
                        ; OUT0_P11=1.9687500
;*OUT0_P11-0.061523437*OUT1_P11
      54 4018FB SUB OUT0_P11,OUT2_P11,R00
                        ; OUT0_P11=1.9687500
;*OUT0_P11-0.061523437
;*OUT1_P11-1.00000000*OUT2_P11
      55 40187C ADD OUT0_P11,OUT2_P11,R04
                        ; OUT0_P11=1.9687500
;*OUT0_P11-0.061523437
;*OUT1_P11-0.93750000*OUT2_P11
      56 4018BC ADD OUT0_P11,OUT2_P11,R06
                        ; OUT0_P11=1.9687500
;*OUT0_P11-0.061523437
;*OUT1_P11-0.92187500*OUT2_P11
      57 4018FC ADD OUT0_P11,OUT2_P11,R08
                        ; OUT0_P11=1.9687500
;*OUT0_P11-0.061523437
;*OUT1_P11-0.91796875*OUT2_P11
      58 40181D ADD OUT0_P11,OUT2_P11,R09
                        ; OUT0_P11=1.9687500
;*OUT0_P11-0.061523437
;*OUT1_P11-0.91601562*OUT2_P11
      59 40189B SUB OUT0_P11,OUT2_P11,R13
                        ; OUT0_P11=1.9687500
;*OUT0_P11-0.061523437
;*OUT1_P11-0.91613769*OUT2_P11
      60 4218FD ADD OUT0_P11,IN0_P11,R00
                        ; OUT0_P11=1.9687500
;*OUT0_P11-0.061523437
;*OUT1_P11-0.91613769
;*OUT2_P11+1.00000000*IN0_P11
                  ;
                  ;
                  IN0_P12 EQU OUT0_P11
                  ;
      61 48181E LDA IN0_P12,OUT0_P11,R1
```

```
                   ;
                   ;
                   OUT2_P12 EQU TEMP
         62 4A00FF LDA OUT2_P12,OUT1_P12,R00
                       ; OUT2_P12=1.00000000*OUT1_P12
         63 4E18EF LDA OUT1_P12,OUT0_P12,R00
                       ; OUT1_P12=1.00000000*OUT0_P12
         64 4E103C ADD OUT0_P12,OUT1_P12,R02
                       ; OUT0_P12=1.00000000*
;OUT0_P12+0.25000000*OUT1_P12
         65 4E10DA SUB OUT0_P12,OUT1_P12,R07
                       ; OUT0_P12=1.00000000*
;OUT0_P12+0.24218750*OUT1_P12
         66 4E181C ADD OUT0_P12,OUT0_P12,R01
                       ; OUT0_P12=1.50000000*
;OUT0_P12+0.36328125*OUT1_P12
         67 4E109D ADD OUT0_P12,OUT1_P12,R13
                       ; OUT0_P12=1.50000000*
;OUT0_P12+0.36340332*OUT1_P12
         68 4E187D ADD OUT0_P12,OUT0_P12,R12
                       ; OUT0_P12=1.50036621*
;OUT0_P12+0.36349204*OUT1_P12
         69 4418FB SUB OUT0_P12,OUT2_P12,R00
                       ; OUT0_P12=1.50036621*
;OUT0_P12+0.36349204
;*OUT1_P12-1.00000000*OUT2_P12
         70 44185C ADD OUT0_P12,OUT2_P12,R03
                       ; OUT0_P12=1.50036621*
;OUT0_P12+0.36349204
;*OUT1_P12-0.87500000*OUT2_P12
         71 44181D ADD OUT0_P12,OUT2_P12,R09
                       ; OUT0_P12=1.50036621*
;OUT0_P12+0.36349204
;*OUT1_P12-0.87304687*OUT2_P12
         72 44185D ADD OUT0_P12,OUT2_P12,R11
                       ; OUT0_P12=1.50036621*
;OUT0_P12+0.36349204
;*OUT1_P12-0.87255859*OUT2_P12
         73 4C18FD ADD OUT0_P12,IN0_P12,R00
                       ; OUT0_P12=1.50036621*
;OUT0_P12+0.36349204
;*OUT1_P12-0.87255859
;*OUT2_P12+1.00000000*IN0_P12
                   ;
                   ;
                   IN0_P13 EQU OUT0_P12
                   ;
         74 4E181E LDA IN0_P13,OUT0_P12,R1
```

```
                       ;
                       ;
                       OUT2_P13 EQU TEMP
          75 4020FF LDA OUT2_P13,OUT1_P13,R00
                              ; OUT2_P13=1.00000000*OUT1_P13
          76 4068EF LDA OUT1_P13,OUT0_P13,R00
                              ; OUT1_P13=1.00000000*OUT0_P13
          77 40601C ADD OUT0_P13,OUT1_P13,R01
                              ; OUT0_P13=1.00000000
;*OUT0_P13+0.50000000*OUT1_P13
          78 40609A SUB OUT0_P13,OUT1_P13,R05
                              ; OUT0_P13=1.00000000
;*OUT0_P13+0.46875000*OUT1_P13
          79 40601B SUB OUT0_P13,OUT1_P13,R09
                              ; OUT0_P13=1.00000000
;*OUT0_P13+0.46679687*OUT1_P13
          80 40683C ADD OUT0_P13,OUT0_P13,R02
                              ; OUT0_P13=1.25000000
;*OUT0_P13+0.58349609*OUT1_P13
          81 4048FB SUB OUT0_P13,OUT2_P13,R00
                              ; OUT0_P13=1.25000000
;*OUT0_P13+0.58349609
;*OUT1_P13-1.00000000*OUT2_P13
          82 40485C ADD OUT0_P13,OUT2_P13,R03
                              ; OUT0_P13=1.25000000
;*OUT0_P13+0.58349609
;*OUT1_P13-0.87500000*OUT2_P13
          83 40489C ADD OUT0_P13,OUT2_P13,R05
                              ; OUT0_P13=1.25000000
;*OUT0_P13+0.58349609
;*OUT1_P13-0.84375000*OUT2_P13
          84 40481D ADD OUT0_P13,OUT2_P13,R09
                              ; OUT0_P13=1.25000000
;*OUT0_P13+0.58349609
;*OUT1_P13-0.84179687*OUT2_P13
          85 40487B SUB OUT0_P13,OUT2_P13,R12
                              ; OUT0_P13=1.25000000
;*OUT0_P13+0.58349609
;*OUT1_P13-0.84204101*OUT2_P13
          86 4A48FD ADD OUT0_P13,IN0_P13,R00
                              ; OUT0_P13=1.25000000
;*OUT0_P13+0.58349609
;*OUT1_P13-0.84204101
;*OUT2_P13+1.00000000*IN0_P13
                       ;
                       ;
                       IN0_P14 EQU OUT0_P13
                       ;
          87 40689E LDA IN0_P14,OUT0_P13,R5
                       ;
                       ;
```

```
                OUT2_P14 EQU TEMP
  88 4220FF LDA OUT2_P14,OUT1_P14,R00
                    ; OUT2_P14=1.00000000*OUT1_P14
  89 4668EF LDA OUT1_P14,OUT0_P14,R00
                    ; OUT1_P14=1.00000000*OUT0_P14
  90 46607E LDA OUT0_P14,OUT1_P14,R04
                    ; OUT0_P14=0.062500000*OUT1_P14
  91 46603A SUB OUT0_P14,OUT1_P14,R02
                    ; OUT0_P14=-0.18750000*OUT1_P14
  92 4660DD ADD OUT0_P14,OUT1_P14,L01
                    ; OUT0_P14=1.8125000*OUT1_P14
  93 46681D ADD OUT0_P14,OUT0_P14,R09
                    ; OUT0_P14=1.8160400*OUT1_P14
  94 46681B SUB OUT0_P14,OUT0_P14,R09
                    ; OUT0_P14=1.8124930*OUT1_P14
  95 46681D ADD OUT0_P14,OUT0_P14,R09
                    ; OUT0_P14=1.8160331*OUT1_P14
  96 46683D ADD OUT0_P14,OUT0_P14,R10
                    ; OUT0_P14=1.8178065*OUT1_P14
  97 4448FB SUB OUT0_P14,OUT2_P14,R00
                    ; OUT0_P14=1.8178065
;*OUT1_P14-1.00000000*OUT2_P14
  98 44485C ADD OUT0_P14,OUT2_P14,R03
                    ; OUT0_P14=1.8178065
;*OUT1_P14-0.87500000*OUT2_P14
  99 44487C ADD OUT0_P14,OUT2_P14,R04
                    ; OUT0_P14=1.8178065
;*OUT1_P14-0.81250000*OUT2_P14
 100 4448BA SUB OUT0_P14,OUT2_P14,R06
                    ; OUT0_P14=1.8178065
;*OUT1_P14-0.82812500*OUT2_P14
 101 44481D ADD OUT0_P14,OUT2_P14,R09
                    ; OUT0_P14=1.8178065
;*OUT1_P14-0.82617187*OUT2_P14
 102 44489B SUB OUT0_P14,OUT2_P14,R13
                    ; OUT0_P14=1.8178065
;*OUT1_P14-0.82629394*OUT2_P14
 103 4468FD ADD OUT0_P14,IN0_P14,R00
                    ; OUT0_P14=1.8178065
;*OUT1_P14-0.82629394
;*OUT2_P14+1.00000000*IN0_P14
                ;
                ;
 104 426CCF LDA DAR,OUT0_P14,L01
                ;
                ;
                ;
                ;TAKE WHAT'S IN THE DAR & SHOVE IT OUT.
                ;
 105 4000EF         NOP
```

```
106 4000EF        NOP
107 4000EF        NOP
108 4000EF        NOP
               ;
               ;
109 8000EF        OUT0                ;MININUM OF 3.5/10**6 SEC
110 8000EF        OUT0                ;TO SETTLE S&H OUTPUT.
111 8000EF        OUT0                ; 800/10**9 SEC CYCLE TIME
112 8000EF        OUT0
113 8000EF        OUT0
               ;
               ;
114 9000EF        OUT1
115 9000EF        OUT1
116 9000EF        OUT1
117 9000EF        OUT1
118 9000EF        OUT1
               ;
               ;
119 A000EF        OUT2
120 A000EF        OUT2
121 A000EF        OUT2
122 A000EF        OUT2
123 A000EF        OUT2
               ;
               ;
124 B000EF        OUT3
125 B000EF        OUT3
126 B000EF        OUT3
127 B000EF        OUT3
128 B000EF        OUT3
               ;
               ;
129 C000EF        OUT4
130 C000EF        OUT4
131 C000EF        OUT4
132 C000EF        OUT4
133 C000EF        OUT4
               ;
               ;
134 D000EF        OUT5
135 D000EF        OUT5
136 D000EF        OUT5
137 D000EF        OUT5
138 D000EF        OUT5
               ;
               ;
139 E000EF        OUT6
140 E000EF        OUT6
141 E000EF        OUT6
```

```
142 E000EF          OUT6
143 E000EF          OUT6
                ;
                ;
144 F000EF          OUT7
145 F000EF          OUT7
146 F000EF          OUT7
147 F000EF          OUT7
148 F000EF          OUT7
149 4000EF          NOP
150 4000EF          NOP
151 4000EF          NOP
152 4000EF          NOP
153 4000EF          NOP
154 4000EF          NOP
155 4000EF          NOP
156 4000EF          NOP
157 4000EF          NOP
158 4000EF          NOP
159 4000EF          NOP
160 4000EF          NOP
161 4000EF          NOP
162 4000EF          NOP
163 4000EF          NOP
164 4000EF          NOP
165 4000EF          NOP
166 4000EF          NOP
167 4000EF          NOP
168 4000EF          NOP
169 4000EF          NOP
170 4000EF          NOP
171 4000EF          NOP
172 4000EF          NOP
173 4000EF          NOP
174 4000EF          NOP
175 4000EF          NOP
176 4000EF          NOP
177 4000EF          NOP
178 4000EF          NOP
179 4000EF          NOP
180 4000EF          NOP
181 4000EF          NOP
182 4000EF          NOP
183 4000EF          NOP
184 4000EF          NOP
185 4000EF          NOP
```

```
   311   186 4000EF       NOP
   312   187 4000EF       NOP
   313   188 5000EF       EOP
   314   189 4000EF       NOP
   315   190 4000EF       NOP
   316   191 4000EF       NOP
PREMATURE EOF


SYMBOL:                             VALUE:

INO_P10                                  0
OUT2_P10                                 1
TEMP                                     1
OUT1_P10                                 2
OUT0_P10                                 3
INO_P11                                  3
OUT2_P11                                 1
OUT1_P11                                 4
OUT0_P11                                 5
INO_P12                                  5
OUT2_P12                                 1
OUT1_P12                                 6
OUT0_P12                                 7
INO_P13                                  7
OUT2_P13                                 1
OUT1_P13                                 8
OUT0_P13                                 9
INO_P14                                  9
OUT2_P14                                 1
OUT1_P14                                10
OUT0_P14                                11


ASSEMBLY COMPLETE
ERRORS     =     0
WARNINGS   =     0
RAMSIZE    =    12
ROMSIZE    =   192
```

```
:18000000F4F0FCF6FEFFF0F0F0F0FEFFF0F0F0F0FEFFF0F0F0F0FEFFDE
:18001800F0F0F0F0FEFFF0F0F0F0FEFFF0F0F0F0FEFFF0F0F0F0FEFFDC
:18003000F0F0F0F0FEFFF4F0F0F0FEFFF6F0F0F0FEFFF4F0F0F0FEFFB6
:18004800F4F0F0F0FEFFF7F1F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF98
:18006000F6F1F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF5F1F0F0FEFF7F
:18007800F4F0F0F0FEFFF4F0F0F0FEFFF4F1F0F0FEFFF4F0F0F0FEFF6B
:18009000F4F0F0F0FEFFF3F1F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF54
:1800A800F2F1F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF1F1F0F0FEFF3F
:1800C000F4F0F0F0FEFFF4F0F0F0FEFFF0F1F0F0FEFFF4F0F0F0FEFF27
:1800D800F4F0F2F2F4FEF4F2F0F0FFFFF4F6F0F8FEFFF4F4F0F0FFFFFD
:1800F000F4F4F0F8FFFBF4F4F0F8F9FCF4F4F0F8F1FBF4F4F0F8F5FDEB
:18010800F4F6F0F0FDFDF4F6F0F0F9FAF4F6F0F0FDFAF4F6F0F0F9FDDD
:18012000F4F6F0F8F1FEF4F8F0F0FFFFF4F8F1F8FEFFF4F8F1F0F9FAAA
:18013800F4F8F1F8FBFAF4F8F1F8FFFDF4F0F1F8FFFBF4F0F1F8F7FC8D
:18015000F4F0F1F8FBFCF4F0F1F8FFFCF4F0F1F8F1FDF4F0F1F8F9FB8F
:18016800F4F2F1F8FFFDF4F8F1F8F1FEF4FAF0F0FFFFF4FEF1F8FEFF4C
:18018000F4FEF1F0F3FCF4FEF1F0FDFAF4FEF1F8F1FCF4FEF1F0F9FD4A
:18019800F4FEF1F8F7FDF4F4F1F8FFFBF4F4F1F8F5FCF4F4F1F8F1FD34
:1801B000F4F4F1F8F5FDF4FCF1F8FFFDF4FEF1F8F1FEF4F0F2F0FFFF11
:1801C800F4F0F6F8FEFFF4F0F6F0F1FCF4F0F6F0F9FAF4F0F6F0F1FB26
:1801E000F4F0F6F8F3FCF4F0F4F8FFFBF4F0F4F8F5FCF4F0F4F8F9FCF6
:1801F800F4F0F4F8F1FDF4F0F4F8F7FBF4FAF4F8FFFDF4F0F6F8F9FED0
:18021000F4F2F2F0FFFFF4F6F6F8FEFFF4F6F6F0F7FEF4F6F6F0F3FAB9
:18022800F4F6F6F0FDFDF4F6F6F8F1FDF4F6F6F8F1FBF4F6F6F8F1FDA4
:18024000F4F6F6F8F3FDF4F4F4F8FFFBF4F4F4F8F5FCF4F4F4F8F7FC84
:18025800F4F4F4F8FBFAF4F4F4F8F1FDF4F4F4F8F9FBF4F4F6F8FFFD69
:18027000F4F2F6FCFCFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF60
:18028800F4F0F0F0FEFFF8F0F0F0FEFFF8F0F0F0FEFFF8F0F0F0FEFF4E
:1802A000F8F0F0F0FEFFF8F0F0F0FEFFF9F0F0F0FEFFF9F0F0F0FEFF30
:1802B800F9F0F0F0FEFFF9F0F0F0FEFFF9F0F0F0FEFFFAF0F0F0FEFF15
:1802D000FAF0F0F0FEFFFAF0F0F0FEFFFAF0F0F0FEFFFAF0F0F0FEFFFA
:1802E800FBF0F0F0FEFFFBF0F0F0FEFFFBF0F0F0FEFFFBF0F0F0FEFFDE
:18030000FBF0F0F0FEFFFCF0F0F0FEFFFCF0F0F0FEFFFCF0F0F0FEFFC2
:18031800FCF0F0F0FEFFFCF0F0F0FEFFFDF0F0F0FEFFFDF0F0F0FEFFA7
:18033000FDF0F0F0FEFFFDF0F0F0FEFFFDF0F0F0FEFFFEF0F0F0FEFF8C
:18034800FEF0F0F0FEFFFEF0F0F0FEFFFEF0F0F0FEFFFEF0F0F0FEFF71
:18036000FFF0F0F0FEFFFFF0F0F0FEFFFFF0F0F0FEFFFFF0F0F0FEFF55
:18037800FFF0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF5E
:18039000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF51
:1803A800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF39
:1803C000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF21
:1803D800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF09
:1803F000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF1
:18040800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFD8
:18042000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFC0
:18043800F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFA8
:18045000F4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF90
:18046800F5F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFFF4F0F0F0FEFF77
:00000001FF
```

APPENDIX C: SPAS20 COMPILER EXAMPLE

```
*:F1:SPAS20             ;Start up the SPAS20 compiler.
*:BUTTER 5,100,0        ;Call the BUTTERWORTH macro.
;
;        The following listing is
;        produced when the macro is called.
;
.*; This is a BUTTERWORTH FILTER GENERATOR for SPAS20;
.*;
.*; Calling sequence :BUTTER ORDER, Fco, LABEL where
.*;   :BUTTER calls the MACRO.
.*;    ORDER is the order of the filter.
.*;    Fco is the cut-off frequency in Hz.
.*;    LABEL is the starting point for POLE numbering.
;
;   The macro expansion has been omitted.
;   The following poles have been generated by the
;   BUTTERWORTH macro.
;
POLE 0 = -30.901697, 95.105651,CONTINUOUS
POLE 1 = -80.901702, 58.778526,CONTINUOUS
POLE 2 = -100.000000, 0.00000000,CONTINUOUS; REAL
;
; Specify the sampling frequency.
; 192= number of 2920 program words.
; 800/10**9= 800 nanoseconds
;
*TS=192*800/10**9       ;SAMPLING FREQUENCY
TS = 1.53599833/10**4
;
;   Now call the bilinear transform.
;
*:BTP 0,10              ;Do a bilinear transform on POLE 0.
;
;   This listing is produced when the bilinear transform
;   is invoked.
;
.*        ;This macro generates a Bilinear transform of a
.*        ;given pole.
.*        ;Calling sequence  :BTP POLE # IN S, POLE # IN Z
.
;
;   The bilinear transform has completed for pole 0.
```

```
;
*:BTP 1,11                ;Bilinear transform on POLE 1.
*:BTP 2,12                ;Bilinear transform on POLE 2.
;
; NOTE : The macro expansion has been omitted for the BTP.
;
*PZ       ;print out all the poles.
POLE 0 = -30.901697, 95.105651,CONTINUOUS
POLE 1 = -80.901702, 58.778526,CONTINUOUS
POLE 2 = -100.000000,0.00000000,CONTINUOUS; REAL
POLE 10 = 0.97067529, 0.091745910,Z
POLE 11 = 0.92491406, 0.056791192,Z
POLE 12 = 0.90793417, 0.00000000,Z; REAL
;
; Now generate the 2920 instructions for each pole.
; Position error has been specified as 0.0001.
;
*CODE POLE 10 PERROR<.0001,.0001
*CODE POLE 11 PERROR<.0001,.0001
*CODE POLE 12 PERROR<.0001,.0001
;
; Text produced by the SPAS20 compiler has been omitted
; between each 'CODE' command.
;
*PZ       ;print out all the poles.
;
POLE 0 = -30.901697, 95.105651,CONTINUOUS
POLE 1 = -80.901702, 58.778526,CONTINUOUS
POLE 2 = -100.000000, 0.00000000,CONTINUOUS; REAL
POLE 10 = 0.97076406, 0.091818725,Z
POLE 11 = 0.92491552, 0.056793292,Z
POLE 12 = 0.90795898, 0.00000000,Z; REAL
;
; Note : Poles 10-12 reflect the generated 2920 instructions.
;
*EXIT     ;Return to the ISIS-II operating system.
```

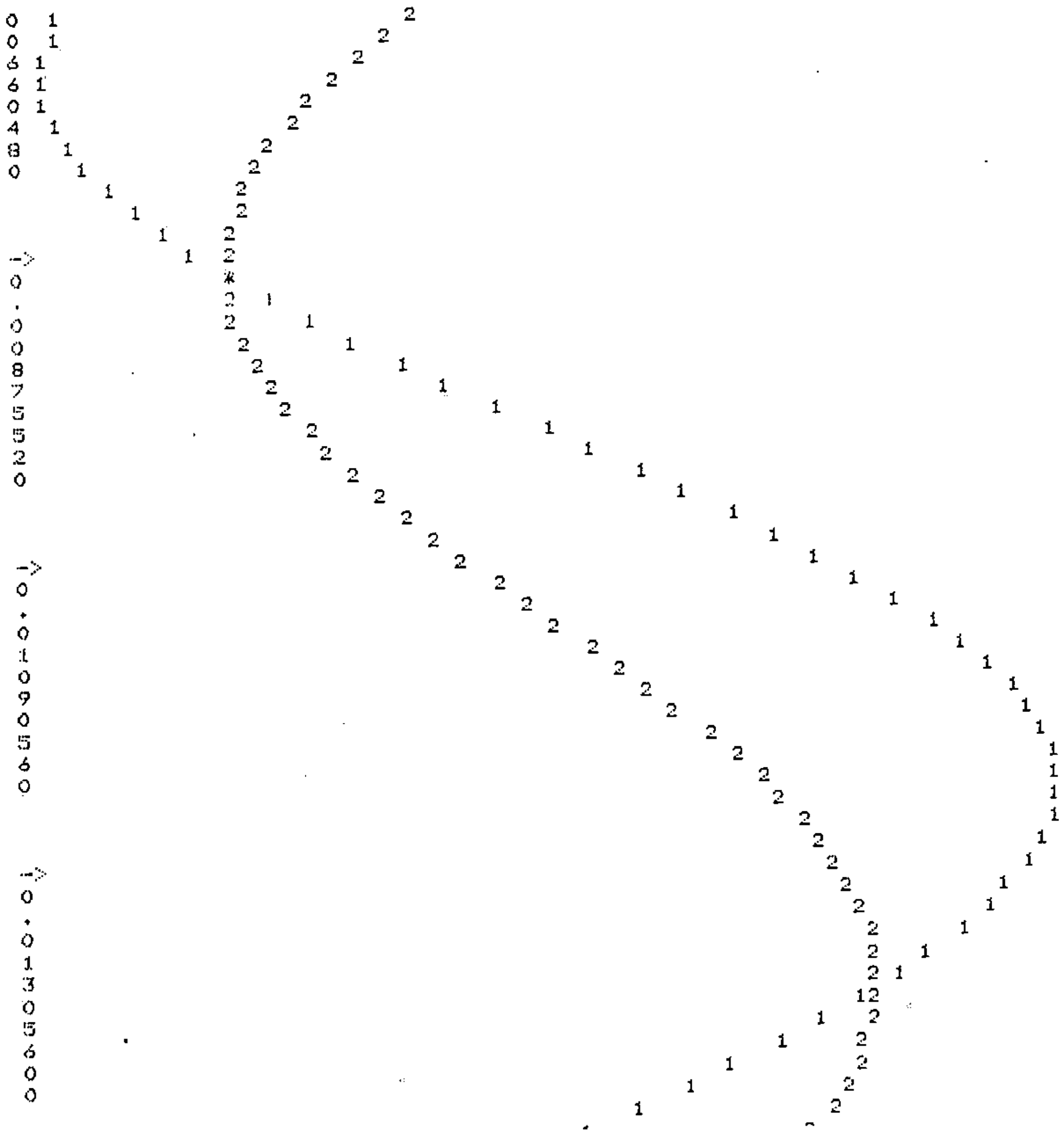
APPENDIX D: 2920 SIMULATOR EXAMPLE

```
*LOAD 5PROG.HEX
*TPROG=192*0.0000008
*TRACE=TIME,INO,OUTO
*QUALIFIER=PC=0
*SIZE=192
*BREAKPOINT=OVF=1 OR PC=0 AND TIME=100*TPROG
*INO=SIN(100*TPI*TIME)
*CONSOLE OFF
*LIST :CO:
*OVF
OVF =   0.00000000
*PRINT ALL
    TIME               INO                OUTO
     0.00015360         0.09635997         0.39648438
     0.00030720         0.19182316         0.36914063
     0.00046080         0.28550103         0.33007813
     0.00061440         0.37652180         0.30273438
     0.00076800         0.46403833         0.29492188
     0.00092160         0.54723599         0.30273438
     0.00107520         0.62534072         0.32617188
     0.00122880         0.69762539         0.34960938
     0.00138240         0.76341748         0.37695313
     0.00153600         0.82210430         0.40429688
     0.00168960         0.87313994         0.43554688
     0.00184320         0.91604951         0.46679688
     0.00199680         0.95043330         0.49804688
     0.00215040         0.97597158         0.53320313
     0.00230400         0.99242656         0.56445313
     0.00245760         0.99964521         0.59570313
     0.00261120         0.99756016         0.62304688
     0.00276480         0.98619092         0.64648438
     0.00291840         0.96564336         0.66992188
     0.00307200         0.93610869         0.68945313
     0.00322560         0.89786162         0.70507813
     0.00337920         0.85125830         0.71289063
     0.00353280         0.79673237         0.72070313
     0.00368640         0.73479141         0.72070313
     0.00384000         0.66601167         0.71679688
     0.00399360         0.59103364         0.70507813
     0.00414720         0.51055483         0.68945313
     0.00430080         0.42532427         0.66992188
     0.00445440         0.33613538         0.64257813
     0.00460800         0.24381787         0.61523438
     0.00476160         0.14923159         0.58007813
     0.00491520         0.05325601         0.53710938
     0.00506880        -0.04321477         0.49414063
     0.00522240        -0.13928383         0.44726563
     0.00537600        -0.23405635         0.39257813
```

```
0.00552960     -0.32665073     0.33789063
0.00568320     -0.41620464     0.28320313
0.00583680     -0.50188530     0.22070313
0.00599040     -0.58289463     0.16210938
0.00614400     -0.65847939     0.09960938
0.00629760     -0.72793569     0.03320313
0.00645120     -0.79061704    -0.02929688
0.00660480     -0.84594004    -0.09179688
0.00675840     -0.89339023    -0.15039063
0.00691200     -0.93252578    -0.21289063
0.00706560     -0.96298223    -0.26757813
0.00721920     -0.98447627    -0.32617188
0.00737280     -0.99680791    -0.37695313
0.00752640     -0.99986240    -0.42382813
0.00768000     -0.99361123    -0.46679688
0.00783360     -0.97811279    -0.50585938
0.00798720     -0.95351094    -0.54101563
0.00814080     -0.92003486    -0.57226563
0.00829440     -0.87799629    -0.59570313
0.00844800     -0.82778633    -0.61523438
0.00860160     -0.76987197    -0.62695313
0.00875520     -0.70479243    -0.63476563
0.00890880     -0.63315386    -0.63476563
0.00906240     -0.55562217    -0.63085938
0.00921600     -0.47291943    -0.61914063
0.00936960     -0.38581563    -0.59960938
0.00952320     -0.29512114    -0.58007813
0.00967680     -0.20167949    -0.55273438
0.00983040     -0.10636086    -0.51757813
0.00998400     -0.01005281    -0.47851563
0.01013760      0.08634880    -0.43945313
0.01029120      0.18194724    -0.39257813
0.01044480      0.27585234    -0.34179688
0.01059840      0.36718965    -0.28710938
0.01075200      0.45510996    -0.23242188
0.01090560      0.53879463    -0.17382813
0.01105920      0.61746470    -0.11132813
0.01121280      0.69038750    -0.05273438
0.01136640      0.75688564     0.00976563
0.01152000      0.81633950     0.07226563
0.01167360      0.86819561     0.13085938
0.01182720      0.91197119     0.18945313
0.01198080      0.94725947     0.24804688
0.01213440      0.97373174     0.30664063
0.01228800      0.99114160     0.35742188
0.01244160      0.99932686     0.40820313
0.01259520      0.99821152     0.45117188
0.01274880      0.98780605     0.49414063
0.01290240      0.96820703     0.52929688
```

0.01305600	0.93959697	0.56445313
0.01320960	0.90224248	0.58789063
0.01336320	0.85649063	0.61132813
0.01351680	0.80276743	0.62695313
0.01367040	0.74157300	0.63476563
0.01382400	0.67347671	0.63867188
0.01397760	0.59911240	0.63867188
0.01413120	0.51917231	0.63085938
0.01428480	0.43440103	0.61523438
0.01443840	0.34558604	0.59570313
0.01459200	0.25355474	0.56835938
0.01474560	0.15916360	0.54101563
0.01489920	0.06329210	0.50585938
0.01505280	-0.03316940	0.46289063
0.01520640	-0.12932222	0.41992188
0.01536000	-0.22427144	0.36914063

```
GRAPHICS ON
*PRINT ALL
   TIME                INO               OUTO
->                                           1          2
0                                               1      2
.                                                   1 2
0                                                    2 1
0                                                   2      1
0                                                    2        1
1                                                    2           1
5                                                     2             1
3                                                      2              1
6                                                       2               1
0                                                         2               1
                                                           2                1
                                                            2                1
                                                             2                1
->                                                            2                1
0                                                               2              1
.                                                                2             1
0                                                                 2           1
0                                                                 2           1
2                                                                  2         1
3                                                                   2      1
0                                                                   2    1
4                                                                   2  1
0                                                                   21
0                                                                 1 2
                                                               1    2
                                                            1      2
                                                         1        2
->                                                    1          2
0                                                 1             2
.                                              1               2
0                                          1                 2
0                                      1                    2
4                                    1                     2
4                               1                       2
5                            1                        2
4                        1                          2
4                     1                          2
0                  1                           2
                1                            2
             1                            2
          1                             2
->       1                            2
0      1                           2
.     1                          2
```


00660480
-> 0.00875520
-> 0.01090560
-> 0.01305600