(NASA-CR-177000) A MICROPROCESSOR BASED ANTI-ALIASING FILTER FOR A PCM SYSTEM Final Report, 31 Oct. 1980 - 31 Oct. 1984 (California Polytechnic State Univ.) 160 p
N86-29117
Unclas
CSCL 17B G3/32 43345

A MICROPROCESSOR BASED ANTI-ALIASING FILTER FOR A PCM SYSTEM

FINAL REPORT

PRINCIPAL INVESTIGATORS

Daniel Creighton Morrow

Doral R. Sandlin

10/31/80 - 10/31/84

California Polytechnic State University

Aeronautical Engineering Department

San Luis Obispo, California

Grant No. NCC 4-1

ABSTRACT

A MICROPROCESSOR BASED ANTI-ALIASING FILTER

FOR A PCM SYSTEM

DANIEL CREIGHTON MORROW

March 1983

This project report describes the design and evaluation of a microprocessor based digital filter. The filter was designed to investigate the feasibility of a digital replacement for the analog pre-sampling filters used in telemetry systems at the NASA Ames-Dryden Flight Research Facility (DFRF). The digital filter will utilize an Intel 2920 Analog Signal Processor (ASP) chip. Testing includes measurements of (a) the filter frequency response and (b) the filter signal resolution. The evaluation of the digital filter was made on the basis of circuit size, projected environmental stability and filter resolution. The 2920 based digital filter was found to meet or exceed the pre-sampling filter specifications for limited signal resolution applications.

ACKNOWLEDGEMENTS

The advice and comments of Harry Chiles and Richard Glover of the Ames-Dryden Flight Research Facility are gratefully acknowledged. I would also like to thank Dr. Gustav Wassel, my project advisor, for his guidance and assistance. The NASA Ames-Dryden Flight Research Facility and the California Polytechnic State University have participated jointly in a graduate student program funded by NASA under grant NCC 4-1. The Cal Poly program director is Dr. Doral Sandlin. I would like to thank both Cal Poly and DFRF for the opportunity to participate in this rewarding program.

TABLE OF CONTENTS

Appendixes Page

LIST OF FIGURES

Chapter One: Introduction

Chapter Two: Digital Filter Design

Chapter Three: Results and Measurements

Appendix A: Transfer Function Graphs

Appendix B: 2920 Hardware Support Circuit

Appendix C: 2920 Processor Overview

LIST OF TABLES

Appendix B: 2920 Hardware Support Circuit

Appendix C: 2920 Processor Overview

LIST OF COMPUTER SOURCE CODES

Appendix D: Pre-Sampling Filter Software Support

Appendix E: Other Support Programs

LIST OF COMPUTER OUTPUT SAMPLES

Appendix E: Other Support Programs

CHAPTER 1

INTRODUCTION

A substantial portion of the research conducted at NASA's Ames-Dryden Flight Research Facility (DFRF) is flight oriented. Airborne test beds allow experiments to be conducted under conditions which cannot be approximated in wind tunnels or pressure chambers. Storing the data collected during an airborne experiment is a definite challenge in the space restricted flight environment. Often, the volume of data collected during flight tests make on-board storage of data, at best, impractical. Therefore, a pulse code modulation (PCM)[1] telemetry system is used to collect, multiplex and transmit data to the ground station for storage and analysis (see Figure 1-1). The ability of the PCM system to accurately measure and transmit data to the ground station is critical to the success of airborne experiments.

---

[1]Ferrel G. Stremler, Introduction to Communication Systems, ed. David Cheng, Leonard A. Gould and Fred Manasse, (Phillipines:Addison-Wesley Publishing Company, Inc., 1977), pp. 355-366.

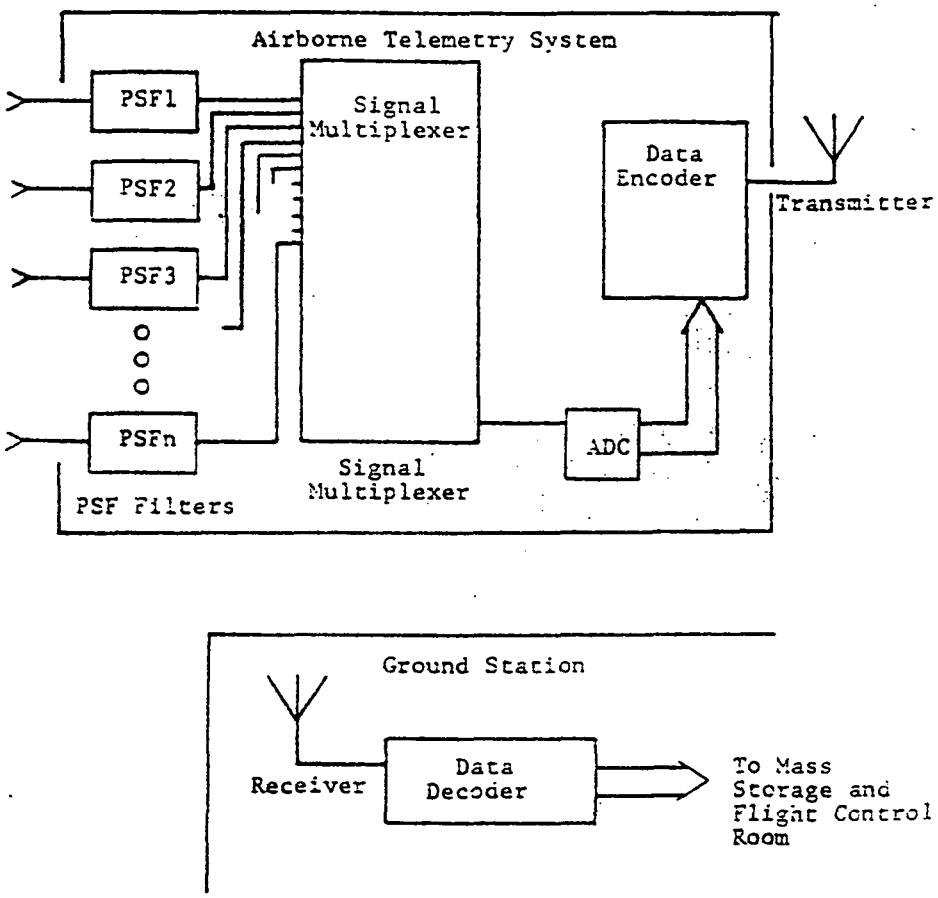


Figure 1-1 Block Diagram of the Telemetry System.

The flight environment is full of electrical noise. The major contributors are the aircraft 400 Hz power supply, the avionics and the power plant. The pre-sampling filters (PSF) perform a twofold service for the system. In the process of attenuating noise signals in frequency bands above that of the data, these low pass filters act as anti-aliasing filters[2,3,4] for the PCM system signal multiplexer. Without these filters, aliasing of high frequency noise into the data bands could seriously impair system accuracy.

The Analog Pre-Sampling Filter

The circuit diagram of the analog PSF is shown in Figure 1-2. If the resistors and capacitors of the circuit are matched, the transfer function of the circuit becomes:

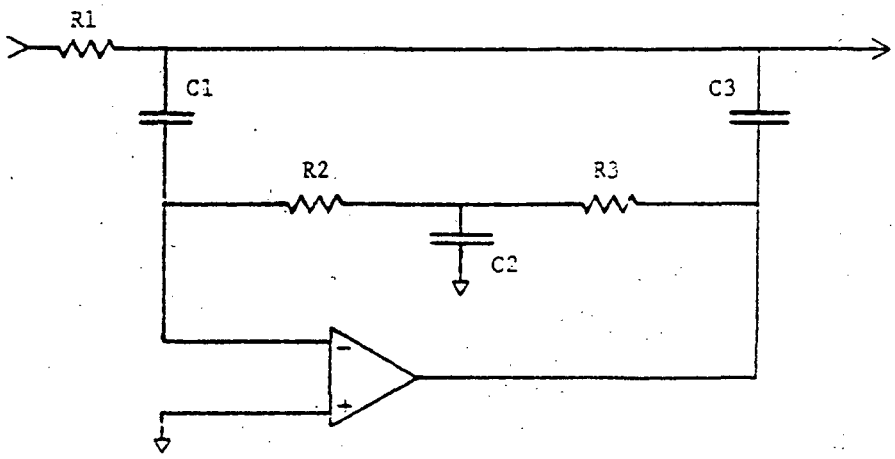
$$H(s) = \frac{W_c^3}{s^3 + 2W_c s^2 + 2W_c^2 s + W_c^3}$$

---

[2]Andreas Antoniou, Digital Filters: Analysis and Design, (United States: McGraw-Hill, Inc., 1979), pp. 151-154.

[3]Lawrence R. Rabiner, Theory and Application of Digital Signal Processing, (New Jersey:Prentice Hall, Inc., 1975), pp. 26-28.

[4]2920 Analog Signal Processor Design Handbook, (Santa Clara: Intel Corporation, 1980), pp. 2-2, 2-3.



$$H(s) = \frac{1}{As^3 + Bs^2 + Cs + 1}$$

where:

A = R1*R2*R3*C1*C2*C3

B = R1*(R2+R3)*C1*C3

C = R1*(C1+C3)

Figure 1-2 Analog Pre-sampling Filter Circuit

$$\text{where: } W_c = \frac{1}{RC},$$

$$R = R1 = R2 = R3,$$

$$C = C1 = C2 = C3.$$

Thus, if the circuit is assembled using matched components, it provides a very efficient implementation of a third order Butterworth filter. The circuit requires few parts and occupies approximately two square inches on a circuit board. While the simplicity of this circuit allows a space efficient design, it also leads to sensitivity to environmental stresses.

As the components age and go through the thermal extremes of flight, the values of the various circuit components tend to drift. The program TRCLC (see appendix E) was used to evaluate the magnitude and phase response of the circuit as the value of R1 varies. Figure 1-3 shows the magnitude response curves when R1 is allowed to vary 10%. At the filter corner frequency (Wa=1 Hz), the gain error associated with a 10% variation in R1 is 1.2 dB. The distortions in the PSF transfer function lead to system errors in measured data. Fortunately, if the distorted transfer function is known, the impact on the final measurements can be minimized by post flight data reduction.

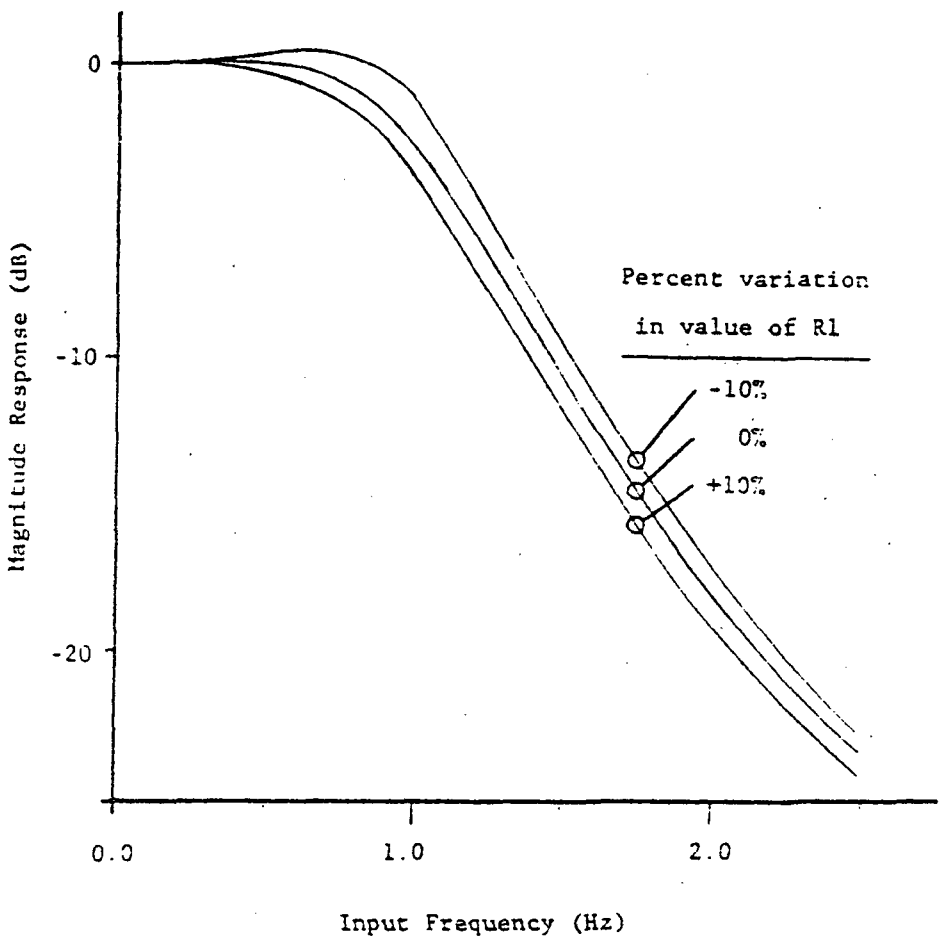


Figure 1-3 Magnitude Response of the Analog Filter for Varying Values of R1

The Digital Alternative

A digital implementation of the PSF filters could provide a significant improvement to the existing PCM system. The transfer functions of digital filters are primarily dependent on the program which drives the processor, rather than the values of the components which form the circuit. Therefore, the digital filter is expected to exhibit improved transfer function stability. By the same argument, improved filter repeatability is also expected. The potential also exists for reducing the physical size of the filter.

This project report describes the design and evaluation of a real-time microprocessor based digital PSF. The goal of the project was to investigate the properties of a digital filter which approximates the analog PSF circuit. The emphasis of the design is on size efficiency and transfer function stability, not on the implementation of an ideal Butterworth filter. The processor utilized to implement the digital filter is the Intel 2920 Analog Signal Processor (ASP). This processor was selected because of its unique design and the availability of support tools.

CHAPTER 2

DIGITAL FILTER DESIGN

Filter Design Specifications

The pre-sampling filter is an important part of the PCM system. The filter attenuates high frequency noise in sensor signals before the PCM system samples the data for transmission. The high frequency noise must be removed to avoid aliasing. The performance of the existing analog filters is degraded by their sensitivity to environmental stresses. The digital PSF is to approximate the response characteristics of the analog circuit. The nominal specifications of this filter are:

1) The filter transfer function:

$$H(s) = \frac{-G}{\left[\frac{s}{W_c}\right]^3 + 3\left[\frac{s}{W_c}\right]^2 + 3\left[\frac{s}{W_c}\right]^1 + 1}$$

where: $s = jW$

W_c = Corner frequency

G = Gain

2) A balanced differential input.

3) Input impedance: 100k ohms minumum.

4) Output impedance: 20k ohms maximum.

5) Nine bit (eight bit plus sign) signal resolution.

6) Four signal inputs per 2920 chip.

The digital filter design process will be treated in four major steps:

1) Filter hardware design

2) Choosing a sample frequency

3) Development of the Z domain transfer function

4) Implementation of the transfer function

The following is a discussion of the engineering process for designing a digital replacement for the analog third order Butterworth PSF.

The 2920 Support Hardware Design

The 2920 ASP requires several types of external hardware support to meet the PSF nominal specifications. The 2920 input channels require buffering to meet impedance requirements and to interface with the differential input signals. The output signals require filtering to remove noise from the D/A converters.

Figure 2-1 shows the functional blocks necessary to interface a 2920 based filter with the existing PCM

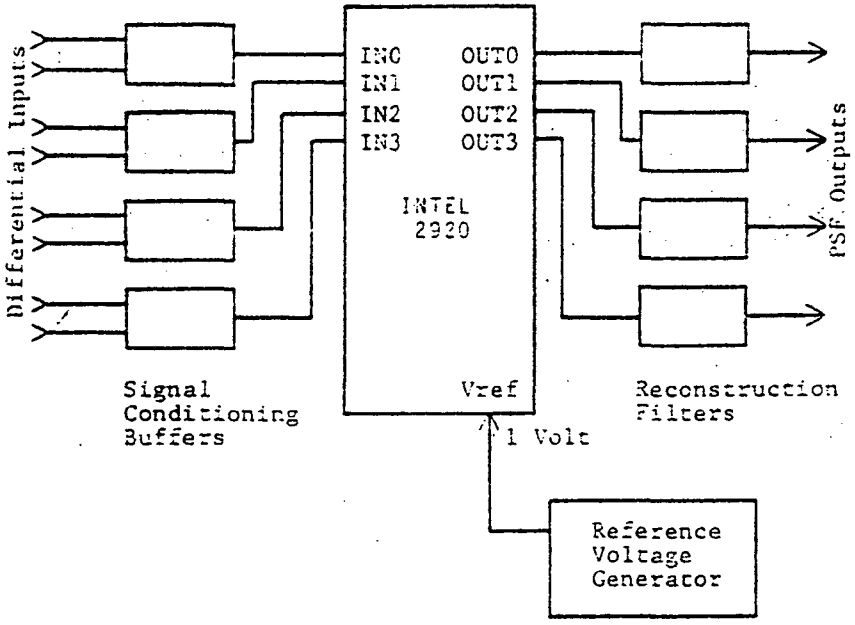


Figure 2-1 Digital PSF Functional Blocks

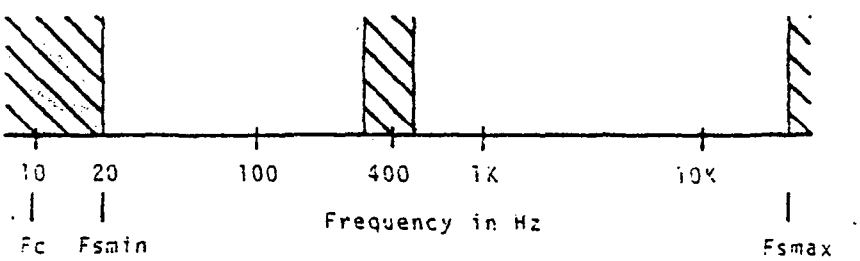
system. The Signal Conditioning Buffers (SCB) are required to meet the nominal filter input specifications. The SCB circuits raise the input impedance of the filter, convert the differential inputs to single line, scale the signal to magnitudes acceptable for the 2920 ADC's and act as anti-aliasing filters for the 2920 by attenuating high frequency signals. The 2920 output signals are passed through reconstruction filters to attenuate noise from the DAC outputs. The reference voltage will be generated local to the processor to reduce reference voltage noise. The circuit shown in Figure B-1 is one possible implementation of these functional blocks. Further information on 2920 support hardware design is contained in Appendix B.

Choosing a Sample Frequency

Choosing the digital filter sampling frequency (F_S) is a fundamental part of the digital filter software design. As shown in Figure 2-2, a broad range of sample frequencies is available for 2920 filter design. The acceptable values of F_S are band limited by the Nyquist frequency[1], noise bands and the speed of the digital processor. The minimum acceptable sample rate is

---

[1]Ferrel G. Stremler, Introduction to Communication Systems, ed. David Cheng, Leonard A. Gould and Fred Manasse, (Phillipines: Addison-Wesley Publishing Company, Inc., 1977), pp. 112-116.



Indicates regions of unacceptable sampling frequencies

-- Fc = 10 Hz

-- Fsmax = 34.7 KHz for 2920-16
= 26.0 KHz for 2920-10

Figure 2-2 Acceptable Sampling Rates for a 2920 Based 10 Hz Filter

determined by the Nyquist criteria applied to the highest signal frequency of interest. This frequency corresponds to the point at which the transfer function of the filter is down 54 dB (the maximum resolution of the 2920 inputs and outputs). This occurs approximately a decade above the corner frequency of the third order Butterworth transfer function (10 F_c). The Nyquist criteria[2] states that the minimum acceptable sample frequency (F_{smin}) is twice the -54 dB frequency or:

$$F_{smin} = 20\ F_c$$

where: F_c = corner frequency of the filter

F_s should be chosen such that it will not cause aliasing of the 400 Hz power supply noise into the passband of the digital filter. The frequency bands which are subject to aliasing are centered around multiples of F_s, and are bounded by:

$$N\ F_s - 10\ F_c$$

and

$$N\ F_s + 10\ F_c$$

where N = 1, 2, 3, 4, . . .

---

[2]Stremler, pp. 112-116.

Any noise occurring in these bands is frequency translated by the sampling process into the passband. Therefore, to avoid frequency aliasing of the 400 Hz power supply noise signal into the passband, F_s must be chosen such that:

$$N F_s - 10 F_c > 400$$

or

$$N F_s + 10 F_c < 400$$

Sample rates which do not meet this condition are considered unacceptable for this application. The maximum sample rate (F_{smax}) is established by the maximum clock frequency of the 2920 processor. This rate as a function of crystal frequency is:

$$F_{smax} = \frac{F_{cry}}{N_{ps}}$$

where: F_{cry} = crystal frequency of the processor

$F_{cry} < 6.6E6$ for 2920-16 processor

$F_{cry} < 5.0E6$ for 2920-18 processor

N_{ps} = number of program steps in the filter code

The program cycle rate (F_{pcr}) is the frequency at which the program space of the 2920 is cycled. The value of F_s is also constrained to be a factor or multiple of F_{pcr}. F_s equals F_{pcr} if a signal sample is processed

every program pass. F_s may be made less than F_{pcr} by a factor N, by enabling the filter input sequence on every Nth program pass. This technique is discussed in greater detail in the 2920 Analog Signal Processor Design Handbook. F_s may be up to four times greater than F_{pcr} by linking duplicate filter code blocks in the 192 step program memory.

In summary, the acceptable values of Fs are limited by the following constraints:

1) $F_s = F_{cpr} * M$

where: $(M=4,3,2,1,\frac{1}{2},\frac{1}{3},\frac{1}{4},....)$

2) $F_s > 20\ F_c$

3) $N\ F_s - 10\ F_c > 400$

or

$N\ F_s + 10\ F_c < 400$

where: F_{pcr} = program cycle rate

$$= \frac{F_{cry}}{4\ N_{ps}}$$

and: F_{cry} = processor crystal frequency,

F_c = filter corner frequency,

N_{ps} = number of program steps in the filter code.

Development of the Z Domain Transfer Function

The development of filter software which will approximate the characteristics of the analog PSF requires that a Z domain equivalent of H(s) be developed. The type of transformation process which will yield the best results depends on the form of H(s) and the target filter structure. For this application, which requires a space efficient implementation, Infinite Impulse Response (IIR) transform techniques will yield the best results. IIR filters approximate analog circuit transfer functions more efficiently than Finite Impulse Response (FIR) filters. Further, studies show that the Bilinear Transform is the preferred IIR transform technique for Butterworth transfer functions[3].

The Bilinear Transform

The Bilinear Transform[4] technique is an approximation method which converts an analog transfer function H(s) to a discrete-time transfer function H(z). This method allows any valid analog transfer function H(s)

---

[3]Bernard Gold and Lawrence R. Rabiner, Theory and Application of Digital Signal Processing, (New Jersey: Prentice-Hall, Inc., 1975), pp. 252-257.

[4]Gold and Rabiner, pp.219-224.

to be expressed as a function of the discrete operator z using the equation:

$$H(z) = H(s) \Big|_{s = 2 F_s \left[\frac{z-1}{z+1}\right]}$$

The Bilinear Transform is an algebraic substitution of a function of the discrete variable z for the laplacean variable s. The derivation of this technique is based on a trapezoidal approximation of an integral, thus the Bilinear Transform technique is not an exact method[5]. The frequency response of H(z) is equal to H(s) if and only if :

$$W_a = 2 F_s \operatorname{Tan}\left[\frac{W_d}{2 F_s}\right]$$

where: W_a = frequency variable (analog)

W_d = frequency variable (discrete)

This equation implies that the frequency response of the Bilinear Transform based discrete time transfer function will be frequency shifted with respect to H(s) as

---

[5]Andreas Antoniou, Digital Filters: Analysis and Design, (United States: McGraw-Hill, Inc., 1979), pp.178-179.

a function of input frequency. This effect is known as "warping"[6].

The warping error is negligible for input frequencies much less than F_s. The warping error increases as the frequency variable W_d approaches one-half of the sample rate. The error may be minimized in the pass band by applying the pre-warping factor (L_w) during the transform process. L_w is defined as:

$$L_w = \frac{W_c}{2F_s \text{ Tan } [\frac{W_c}{2 F_s}]}$$

where: W_c = filter corner frequency.

L_w shifts the frequency response of the digital transfer function so that the amplitude response of the digital transform is exactly equal to the amplitude response of the analog transfer function at W_c. Applying the Bilinear Transform to the transfer function of the general third order Butterworth pre-sampling filter shown

---

[6]Antoniou, pp.182-185

in Figure 1-2 we find:

$$H(s) = \frac{1}{\left[\frac{s}{W_c}\right]^3 + 3\left[\frac{s}{W_c}\right]^2 + 3\left[\frac{s}{W_c}\right]^1 + 1} \Bigg|_{s=2F_s \, L_w \left[\frac{z-1}{z+1}\right]}$$

yields the z transform;

$$H(z) = \frac{1z^3 + 3z^2 + 3z + 1}{K_0 z^3 + K_1 z^2 + K_2 z + K_3}$$

where:
$$K_0 = 8 R_{sc}^3 + 8 R_{sc}^2 + 4 R_{sc} + 1$$
$$K_1 = -24 R_{sc}^3 - 8 R_{sc}^2 + 4 R_{sc} + 3$$
$$K_2 = 24 R_{sc}^3 - 8 R_{sc}^2 - 4 R_{sc} + 3$$
$$K_3 = -8 R_{sc}^3 + 8 R_{sc}^2 - 4 R_{sc} + 1$$

R_{sc} = sample rate to corner frequency ratio

$$= \frac{L_w * F_s}{W_c} = \frac{1}{2 \, \mathrm{Tan}\left[\frac{W_c}{2F_s}\right]}$$

z = discrete-time operator

The analog to digital transfer function transformation for the Butterworth filter is complete once the values of K_0 through K_3 have been calculated. The results of the transformation are compared to the analog response in Figure 2-3. The frequency response of the H(z) was evaluated by the program MAGCK (see appendix E).

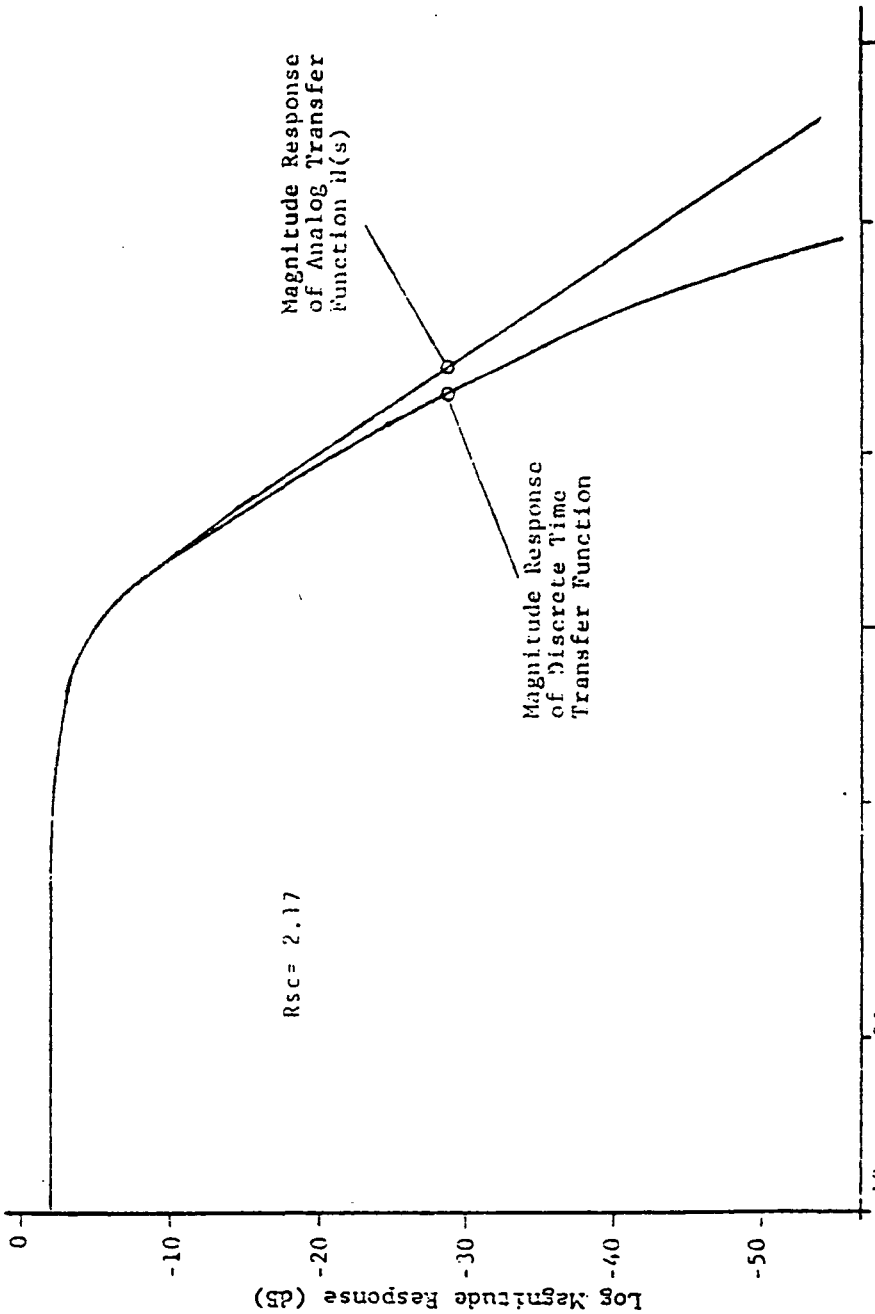


Figure 2-3a Comparison of the Analog Magnitude Response with the Magnitude Response of the Bilinear Transform Resultant H(z)

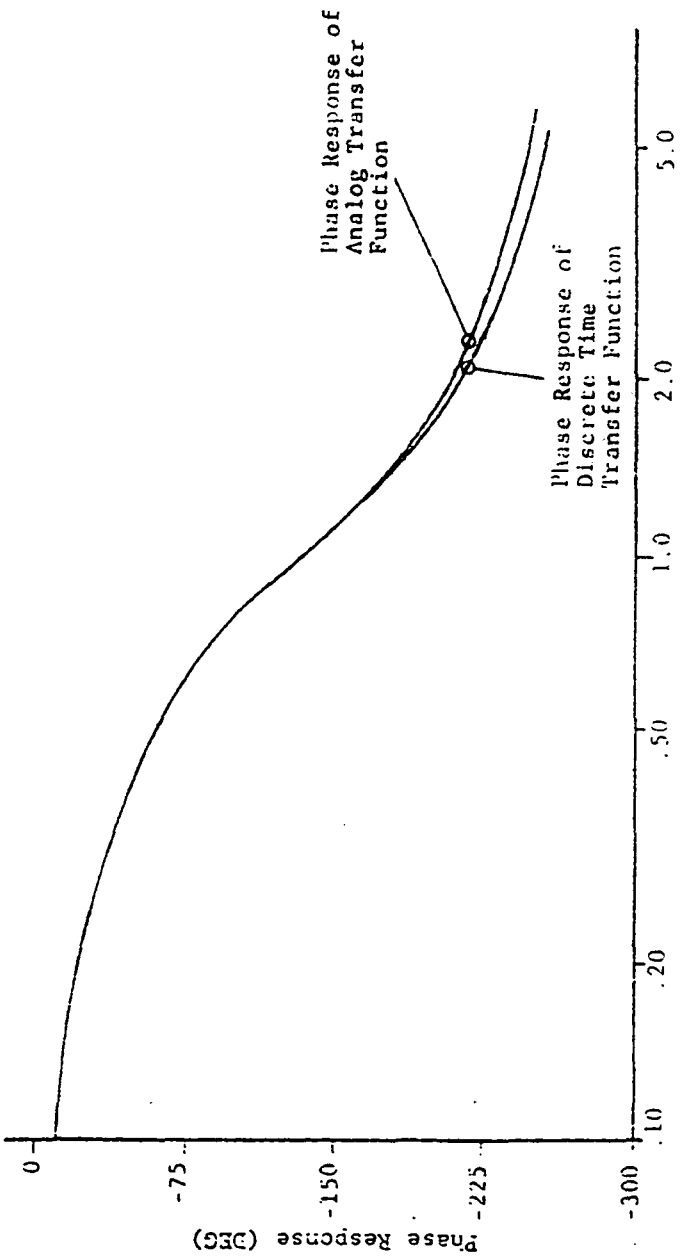


Figure 2-3b Comparison of the Analog Phase Response with the Phase Response of the Bilinear Transform Resultant H(z)

The magnitude response of the transforms is normalized to 0.8 volts/volt to avoid the non-linearities in the 2920-16D ADC's and DAC's. This normalization convention is carried throughout this investigation.

Implementation of H(z)

Implementing H(z) with a 2920 ASP requires that the discrete transform be converted into a sequence of instructions that the 2920 can execute. The structure of the filter algorithms is an important consideration given the program length restrictions of the 2920. Realizing four filters with one 2920 implies that the length of each individual filter algorithm is limited to forty-eight program steps.

Benchmark programs were written based on the ladder, parallel, direct and cascade structures to evaluate the relative efficiencies of the 2920 algorithms. The benchmark filters implement 300 Hz PSF's with a minimum signal resolution of nine bits. The parallel and ladder structures did not perform well in this application. Both these structures failed to meet the signal resolution requirements. The direct and cascade forms were found to be better suited to the 2920 architecture.

Figure 2-4 shows the flow diagrams of the cascade and direct form-2 algorithm structures[7]. The Z^{-1} term represents a memory storage element where data is time delayed one sample. The sections with discrete feedback are called recursive sections, the sections without feedback are called non-recursive sections. The recursive sections implement the denominator (poles) of H(z), and the non-recursive sections implement the numerator (zeros).

The direct form-2 structure implements H(z) with fewer multiplications and memory variables than the cascade form, but the resolution requirements impact the length of the multiplication algorithms to a greater degree than in the cascade form. The direct form requires three full accuracy recursive multiplications and four non-recursive multiplications. The cascade form also requires four full accuracy recursive multiplications, but the gain of the recursive section is divided into two parts, thus reducing the accuracy requirements of these multiplications. Further, the gain of the two non-recursive sections can be distributed so that three of the multiplications are one instruction long. This effectively reduces the number of non-recursive

---

[7]Antoniou, p.76.

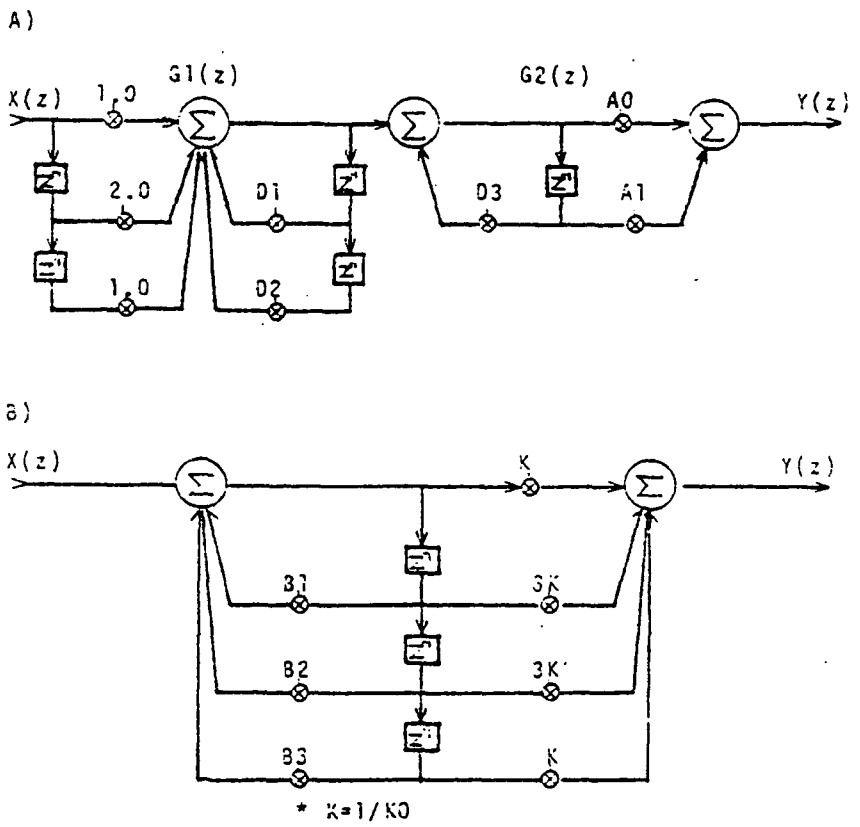


Figure 2-4 Cascade Form (A) and Direct Form 2 (B) Flow Diagrams

multiplications in the cascade structure to two.

The third order recursive section of the direct form is subject to oscillations caused by power up conditions. Under normal circumstances stability is not a problem for these digital Butterworth filters. Occasionally power up conditions leave random values in the delay memories which cause ALU overflows during multiplications. Recursive sections feed the overflow resultants back into the algorithm. Under certain conditions the overflow resultants cause new overflow errors. When this occurs the recursive section goes into an oscillatory mode. The instability is observed only in odd orders of recursive sections. This unstable mode can be prevented by software which detects the mode and damps the oscillations. The additional instructions of the software fix further reduce the efficiency of the direct structure algorithms.

A comparison of the relative efficiencies of the benchmark programs shows that the cascade structure is best suited for the PSF application.

The Cascade Difference Equation

The cascade form of H(z) may be derived directly from the results of the Bilinear Transformation. Applying

the Bilinear Transform to H(s) it was found that:

$$H(z) = \frac{Y(z)}{X(z)} = \frac{1z^3 + 3z^2 + 3z + 1}{K_0 z^3 + K_1 z^2 + K_2 z + K_3}$$

The cascade form of the discrete transfer function is derived by expressing H(z) as the product of second order transfer functions. In this application, H(z) will be expressed as the product of two transfer functions, $G_1(z)$ and $G_2(z)$. The transfer functions $G_1(z)$ and $G_2(z)$ are defined as:

$$H(z) = \frac{Y(z)}{X(z)} = G_1(z) * G_2(z)$$

where:

$$G_1(z) = \frac{G'(z)}{X(z)} = \frac{1 + 2 z^{-1} + 1 z^{-2}}{1 - d_1 z^{-1} - d_2 z^{-2}}$$

and:

$$G_2(z) = \frac{Y(z)}{G'(z)} = \frac{a_0 + a_1 z^{-1}}{1 - d_3 z^{-1}}.$$

Where a_0, a_1 are defined such that:

$$N(z) = \frac{1.}{K_0} [\ 1 + 3 z^{-1} + 3 z^{-2} + 1 z^{-3}\ ]$$

$$= [\ 1 + 2 z^{-1} + 1 z^{-2}\ ] * [\ a_0 + a_1 z^{-1}\ ]$$

and d_1, d_2, d_3 are defined such that:

$$D(z) = 1 - b_1 z^{-1} - b_2 z^{-2} - b_3 z^{-3}$$
$$= [\, 1 - d_1 z^{-1} - d_2 z^{-2} \,] * [\, 1 - d_3 z^{-1} \,]$$

$$\text{with: } b_1 = -K_1/K_0$$
$$b_2 = -K_2/K_0$$
$$b_3 = -K_3/K_0$$

Gain Scaling in The Cascade Algorithm

If the flow diagram of Figure 2-4a was programmed as shown into a 2920, the output Y(nT) would be distorted by ALU overflow errors. The two's complement fixed point 2920 ALU overflows if the result of any calculation exceeds one (some overflow protection is provided, see the 2920 Assembly Language Manual for details). To avoid ALU overflow errors the signal should be scaled as it is processed through the filter algorithm.

The magnitude of a section gain determines the amount of signal scaling required to prevent overflow errors. A gain scale factor, C_i, is carefully introduced into each section such that the gain of the entire filter does not change. This requires that:

$$1 = C_1 * C_2 * C_3 * \ldots . C_i$$

where: i = section number

A further restriction on the value of C_i is made to insure that the scale factors do not significantly lengthen the instruction code. C_i is restricted to be a power of two. This allows C_i to be implemented with one instruction. The binary scale factor (SF_i) is the exponent of two which determines C_i. The values of SF_i are limited to integers between negative thirteen and positive two by the instruction set of the 2920. The two restrictions on the allowable values of C_i are:

1) $C_i = 2^{SF_i}$

where $SF_i = -13,-12,-11,-10,.....0,1,2$

2) $1 = C_1 * C_2 * C_3 * .... C_j$

Signal scaling prevents ALU overflows, but it also increases the accuracy requirements in the algorithms, which increases the required accuracy of the multiplications. The optimum magnitude of C_i will be the largest valid power of two which prevents ALU overflows during signal processing. This implies that:

$$C_i A_i (1+k) < 1$$

$$\text{Solving for } C_i: \quad C_i < \frac{1}{A_i\ (1+k)}$$

where: C_i = gain scale factor

$= 2^{SF_i}$

and: A_i = gain of the section

$$= G(z) \Big|_{z=1}$$

with G(z) = the section transfer function

k = percent overshoot of the section transfer function

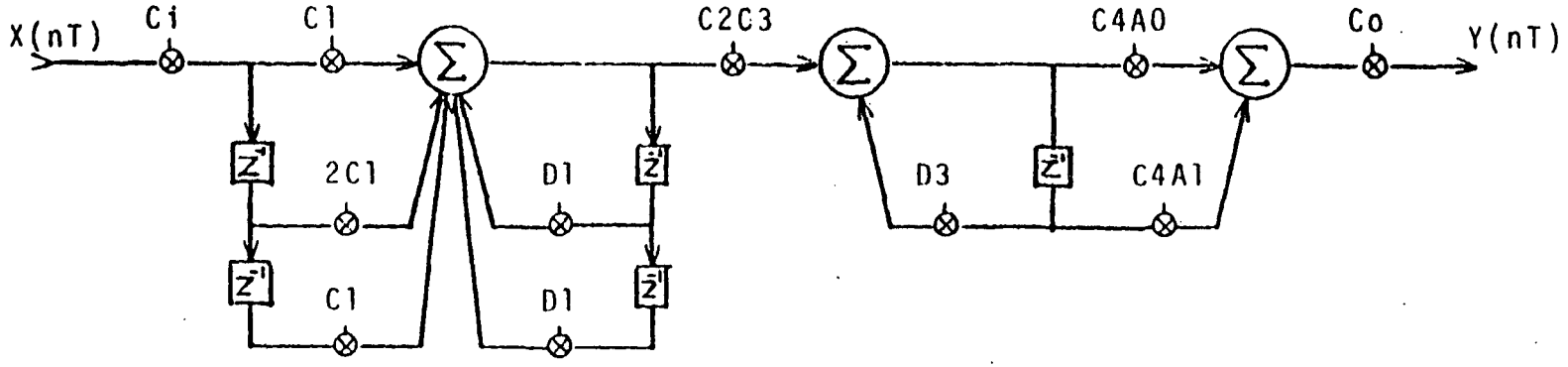
Solving this equation for the binary scale factor SF_i:

$$SF_i < - \frac{\text{Ln } A_i + \text{Ln } (1+k)}{\text{Ln } 2}$$

The binary scale factor may be determined from this equation by finding the largest integer value of SF_i which will satisfy the equation.

The method in which gain scaling is implemented is different for the recursive and non-recursive sections. In recursive sections, the scale factor is implemented as a binary shift applied to the section input. The coefficients are not modified. In non-recursive sections the coefficients and the scale factor are multiplied together to form new non-recursive section coefficients.

Once the gain scale factors are properly placed in the flow diagram (see Figure 2-5), it is ready to be



Where; $$H(z) = \frac{Y(nT)}{X(nT)} = \frac{1 + 2Z^{-1} + Z^{-2}}{1 - D1Z^{-1} - D2Z^{-2}} * \frac{A0 + A1Z^{-1}}{1 - D3Z^{-1}}$$

and, $$1 = Ci * C1 * C2 * C3 * C4 * Co$$

Figure 2-5 The Flow Diagram of the Digital Filter Test Algorithms

converted to 2920 instruction code. C_i and C_o buffer the magnitude of the signal input and the algorithm output.

Accuracy of Multiplications

The flow diagram in Figure 2-5 contains five full accuracy multiplications. 2920 program memory constraints limit the number of instructions in each filter algorithm to forty-eight steps. An average 24 bit multiplication requires eight program steps. Five 24 bit multiplications would consume forty of the forty-eight available program steps. Clearly, if all the filter functions are to be implemented within the forty-eight instruction step limit, the multiplication algorithms must occupy less space. The multiplication algorithms can be made more space efficient by decreasing the accuracy of the multiplications. The amount of multiplication accuracy required to preserve the signal resolution, as it is processed through a section, is a function of the resolution of the input signal and gain of the section.

The binary resolution of a fixed point digital signal is defined as the number of bits in the signal which contain accurate data. The original accuracy of the sampled data is limited to eight bits plus sign by the resolution of the A/D converters. The binary resolution of a signal (R_i) at the input to section i can be

approximated by finding the largest integer which satisfies the equation:

$$R_i < R_s - SF_i - \frac{\text{Ln } A}{\text{Ln } 2}$$

where: R_i = Binary resolution of input to section i (number of bits),

R_s = Binary resolution of signal (number of bits),

A = Product of the filter sections prior to section i.

To establish the relationship between the input resolution, section gain and multiplication accuracy, the effects of finite signal accuracy in the section algorithms must be established.

The value of a finite accuracy signal may be expressed as the value of the infinite accuracy signal plus an error term (E_{rs}).

$$X(nT)_f = X(nT)_{in} + E_{rs}$$

where: $X(nT)_f$ = value of finite accuracy signal

$X(nT)_{in}$ = value of infinite accuracy signal

E_{rs} = finite signal accuracy error

Recursive section difference equations have the form:

$$X(nT)_{in} = Y(nT)_{in} + B_1 X(nT-T)_{in} + B_2 X(nT-2T)_{in} \ldots$$
$$\ldots B_j X(nT-jT)_{in}$$

where: $Y(nT)$ = discrete input to the section

$X(nT-T)$ = section output

Then, assuming finite signal accuracy, the difference equation becomes:

$$X(nT)_f = Y(nT)_{in} + E_{rs} + (B_1 * X(nT-T)_{in}) + (B_1 * E_{rs})$$
$$+ (B_2 * X(nT-2T)_{in}) + (B_2 * E_{rs}) + \ldots$$
$$\ldots + (B_j * X(nT-jT)_{in}) + (B_j * E_{rs})$$

or: $$X(nT)_f = X(nT)_{in} + E_{rs} + (B_1 * E_{rs}) + (B_2 * E_{rs}) + \ldots$$
$$\ldots + (B_j * E_{rs})$$

The finite signal error E_{rs} is bounded by the signal resolution:

$$\left| E_{rs} \right| < 2^{-R_i}$$

Therefore, the maximum error contributed to a signal by the multiplications in a recursive section may

be expressed as:

$$E_{rs|rsec} = E_{rs} + B_1 * E_{rs} + B_2 + E_{rs} \ldots . B_j * E_r$$
$$= E_{rs} * (1 + B_1 + B_2 + B_3 \ldots)$$
$$< 2^{-Ri} * \frac{1}{A_i}$$

where: $E_{rs|rsec}$ = maximum section output error due to finite accuracy signal input for recursive section.

Similarly, for the non-recursive section it can be shown that:

$$E_{rs|nrsec} = D_0 * E_{rs} + D_1 * E_{rs} + D_2 * E_{rs} \ldots . D_j * E_{rs}$$
$$= E_{rs} * (D_0 + D_1 + D_2 + \ldots D_j)$$
$$< 2^{-Ri} * A_i$$

where: $E_{rs|nrsec}$ = maximum section output error due to finite accuracy signal input for recursive section.

and: A_i = gain of section i.

The effects of finite accuracy multiplications are determined by a similar argument. The multiplication constants are expressed as the sum of the infinite

accuracy constant (B_j) plus an error (E_{r_j}):

$$B_{j_f} = B_{j_{in}} + E_{r_j}$$

where: B_{j_f} = the value of the finite accuracy constant,

$B_{j_{in}}$ = The value of the infinite accuracy constant,

E_{r_j} = finite constant error for B_j.

Again, recursive section difference equations have the form:

$$X(nT)_{in} = Y(nT)_{in} + (B_{1_{in}} * X(nT-T)_{in}) + (B_{2_{in}} * X(nT-T)_{in}) \ldots$$

$$\ldots + (B_{j_{in}} * X(nT-jT))$$

The results of the finite accuracy multiplications for recursive sections may then expressed as:

$$X(nT)_f = Y(nT)_{in} + (B_{1_{in}} + E_{r_1}) * X(nT-T) + \ldots$$

$$\ldots + (B_{j_{in}} + E_{r_j}) * X(nT-jT)$$

$$= X(nT)_f + E_{r_1} * X(nT-T) + E_{r_2} * X(nT-2T) + \ldots$$

$$\ldots + E_{r_j} * X(nT-jT)$$

$X(nT)$ is bounded by the fixed point memory such that:

$$|X(nT)| < \frac{1}{A_i}$$

Also, the individual multiplication errors are bounded by the accuracy of the multiplication algorithm:

$$\left| E_{r_j} \right| < 2^{-R_m}$$

where: R_m = accuracy of multiplications (assumed to be uniform in section i).

Therefore the the maximum output error associated with finite accuracy multiplications in recursive sections is:

$$E_{rm|rsec} < N_m * 2^{-R_m} * \frac{1.}{A_i}$$

where: $E_{rm|rsec}$ = maximum output error due to finite multiplications in a recursive sections.

For the non-recursive sections, the error associated with finite multiplications is:

$$E_{rm|nrsec} < N_m * 2^{-R_m} * A_i$$

where: $E_{rm|nrsec}$ = maximum output error due to finite multiplications in a non-recursive sections.

The finite accuracy multiplications will not degrade the section performance if the error associated with the multiplications is less than the maximum output signal resolution. Un-degraded section performance requires that the errors due to the finite multiplications are less than errors due to the finite signal accuracy. The inequalities:

$$E_{rm|rsec} < E_{rs|rsec}$$

and

$$E_{rm|nrsec} < E_{rs|nrsec}$$

state this condition. Solving these inequalities for the required binary multiplication accuracy yields the same relationship between multiplication accuracy, number of multiplications and input signal resolution for recursive and non-recursive sections.

These equations reduce to the inequality:

$$R_m > R_i + \left[\frac{\text{Ln } N_m}{\text{Ln } 2}\right]$$

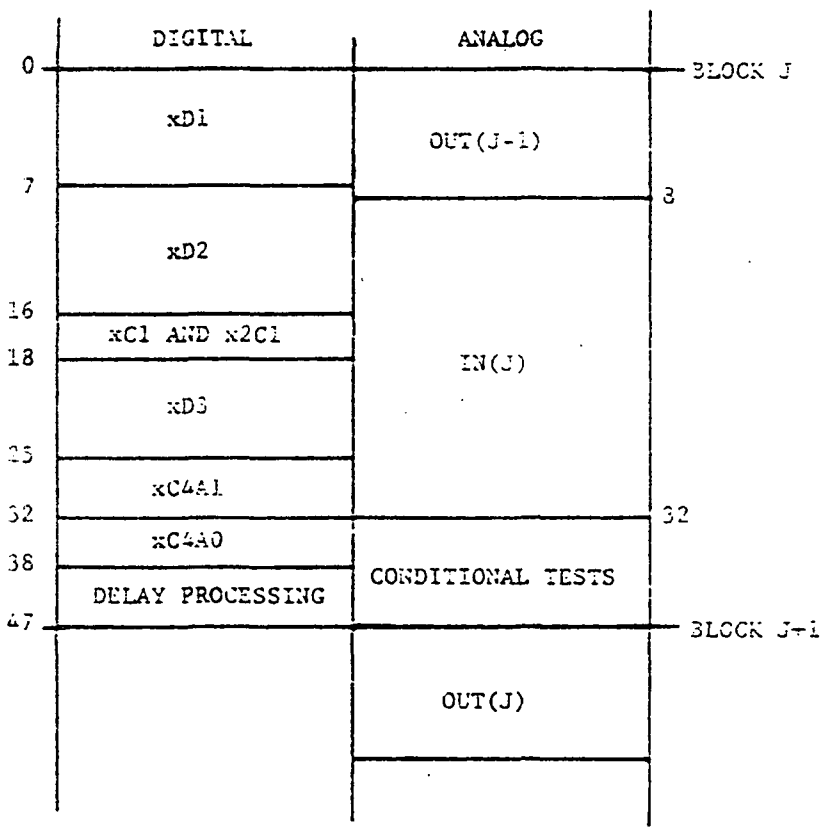
where: R_m = required binary accuracy of section multiplication

R_i = input signal resolution

N_m = number of multiplications in the section

The minimum required accuracy of the multiplications in a section may be determined by finding the smallest integer which will satisfy this equation.

Converting the final flow diagram to 2920 instruction code is a straight forward task. The coefficient multiplications must be coded into efficient algorithms using the section Rm as a standard for minimum accuracy. Figure 2-6 shows a breakdown of the digital and analog function placement in the forty-eight step program block. The code is arranged to minimize the phase delay associated with numerical processing. The code block shown was used in the development of the PSFSS software. The A/D conversion is not complete until late in the processing, so all the multiplications which do not require the input are performed first. The final portion of the code block performs the sample rate division process and manages the time delayed variable storage.



(* xDi indicates multiplication of constant Di)

Figure 2-6 Placement of Functional Blocks in Program Space

CHAPTER 3

RESULTS AND MEASUREMENTS

The evaluation of the 2920 based PSF includes several areas of its design. The filter size, accuracy and limitations are all important considerations. The issue of filter size is investigated in Appendix B. The test results presented in this section demonstrate filter accuracy and provide an indication of the limitations of the 2920 ASP.

The final space requirements of a 2920 based PSF are largely decided by the complexity of the 2920 support hardware. The 2920 occupies 1.5 square inches of the estimated 10 square inches occupied by the PSF circuit shown in Figure B-1. The remainder of the space is consumed by the various 2920 support and PCM interface circuits. The circuit presented in Appendix B was designed using available components in order to provide a rough estimate of the size requirements of a 2920 PSF which meets all the input and output specifications. The design is not space optimal. Several special purpose linear chips available could be used to reduce the size of the circuit considerably.

The accuracy of the 2920 processor and PSF algorithms was determined by measuring filter transfer functions from a 2920 PSF test circuit. Three basic PSF algorithms were written for the 2920, and from these, using sample frequency division techniques, fourteen filter algorithms were developed. These algorithms form the PSFSS library presented in Appendix D.

Test Circuitry and Transfer Function Measurements

A block diagram of the test set which measured the transfer functions is shown in Figure 3-1. The filter input signal is provided by a noise source from the HP 54410A A/D Converter. The output of the filter is processed by the HP 54410A A/D Converter and a HP 54470B Digital Filter. The digitized output signal is then processed by the HP 5420A Digital Signal Analyzer to determine the filter transfer function. A circuit diagram of the 2920 and test support hardware is shown in Figure 3-2. The input SCB's and local reference voltage generator were not included in the test set to eliminate the impact of these circuits on the measurements. The power supply voltages, reference voltage and bread board for the test circuit were supplied by an EL Instruments ELITE 2 Circuit Design Test System. The reconstruction filters on the 2920 outputs were required to minimize interaction between the 2920 DAC's and the sampler in the

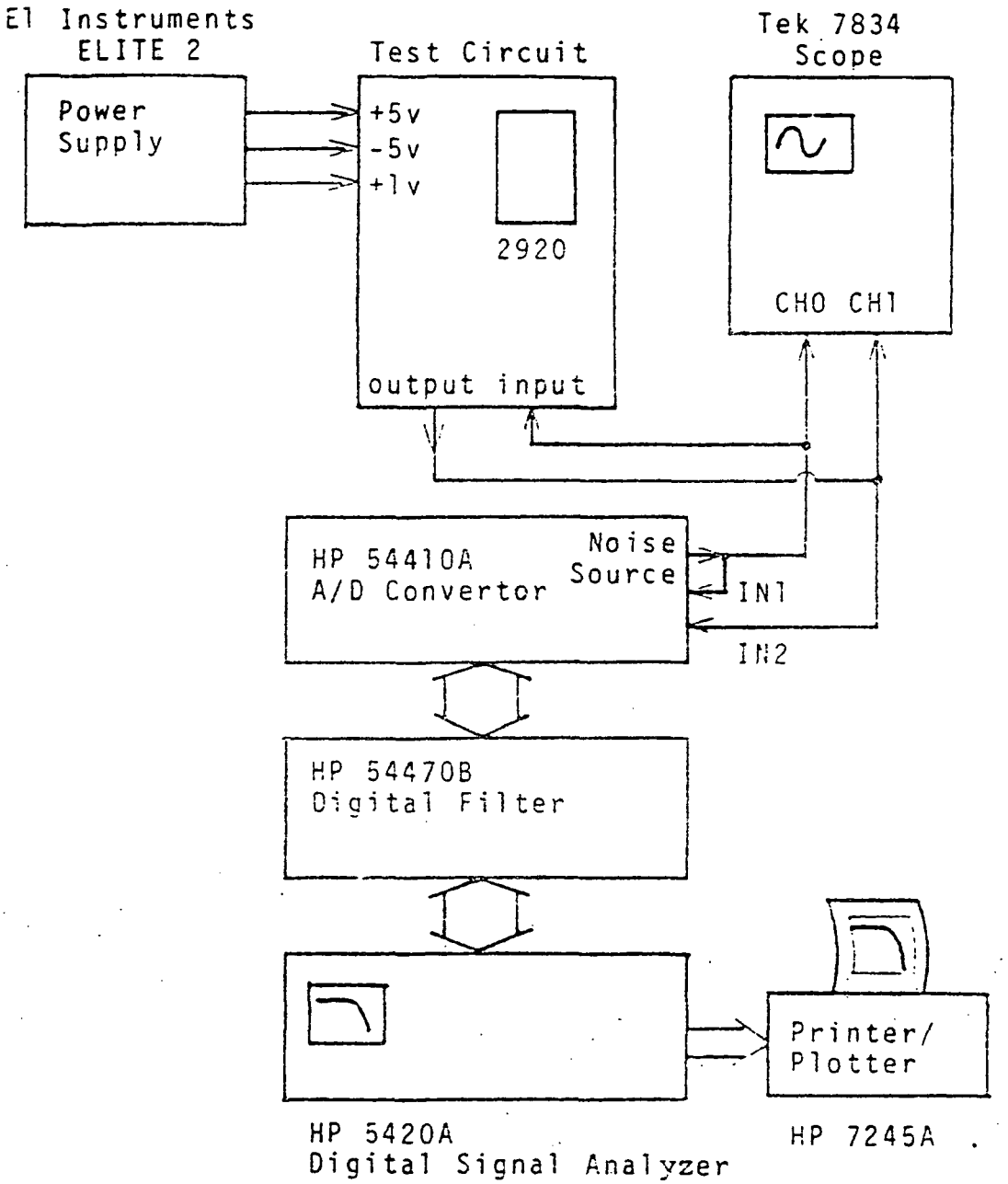


Figure 3-1 Filter Test Set

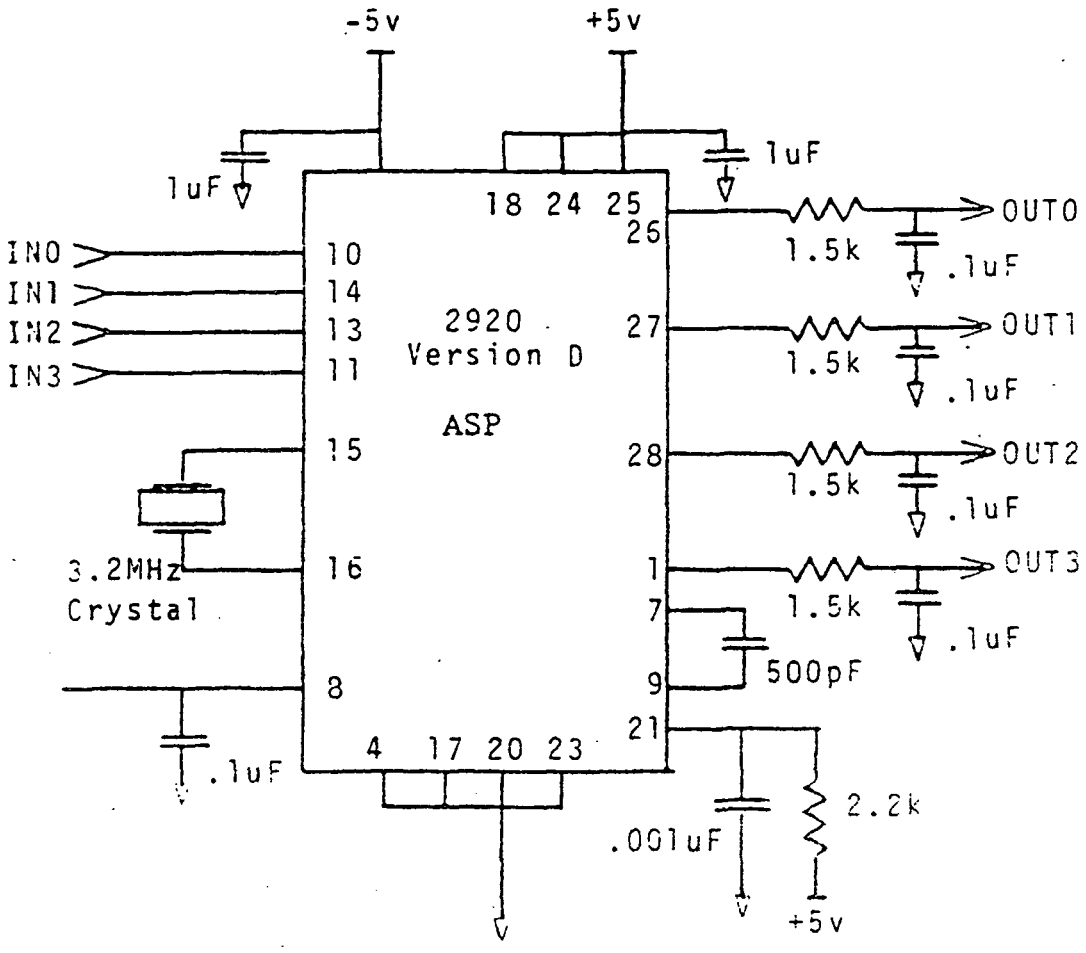


Figure 3-2 2920 PSF Algorithm Test Circuit

HP A/D converter. Without these filters the high frequency noise added by the D/A conversion could cause aliasing errors in the measurements. The transfer function of the test circuit was measured using a 2920 test program which passes the sampled input directly to the output without modification (other than a D-C offset error compensation). The test circuit transfer function (Figure 3-3) illustrates several effects caused by the sampling process[1]. The magnitude response of the circuit shows the magnitude of the output as a function of the input frequency. The frequency of the output signal is always less than the sample rate, regardless of the input frequency. The magnitude minimum and 180 degree phase shift observed at multiples of the sample rate are characteristic of the test set. The slight roll off of the magnitude response is caused by signal skewing during the sample capacitor charging cycle.

2920 Version D Software Fixes

The D version 2920 has several glitches which affect the processor operation[2]. Intel has released new

---

[1] 2920 Analog Signal Processor Design Handbook, (Santa Clara: Intel Corporation, 1980), pp. 2-4.

[2] Carmel Vernia, "Signal Processing Products Status", Signal Processing News, April 1982, p. 2.

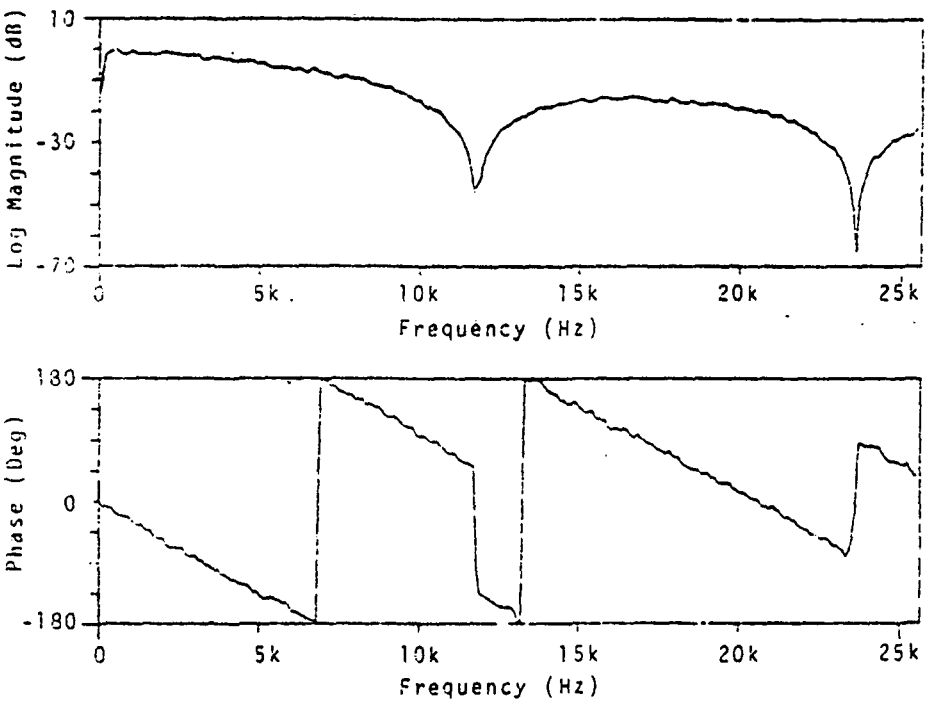


Figure 3-3 Magnitude and Phase Response of the 2920 I/O Test Algorithm

versions of the 2920 which are free from these operational problems[3]. Most of the operational 'glitches' had no impact on the PSF filter operation; others require software fixes to insure proper operation. These fixes appear in the input sequence of all the filter algorithms written for this investigation. The SUB DAR,KM2,R00,CND6 and LDA T,T,R00,CND4 instructions are all version D fixes. These instructions occupy four program steps which could otherwise be used for multiplications. Separate fixes are used to compensate for signal conversion errors.

A test algorithm which samples the analog input, converts it to a digital value, then back to analog was used to measure the 2920 conversion process characteristics. Figure 3-4 shows the results of the measurement. The linear regions for the output converters are from -.95 to 0 volts, and 0 to .9 volts. Beyond these regions, the converters saturate. The gain of the PSF filters has been normalized to .8 volts/volt to prevent DAC saturation. The conversion processes also introduce a negative DC offset. A typical offset can be seen in the linearity curve shown in Figure 3-4. Intel states this offset is typically -16 mV, -70 mV maximum. To partially correct for the offset, a DC value of .062 is added to the

---

[3]Carmel Vernia, "2920 Status", Signal Processing News, January 1982, pp. 6-12.

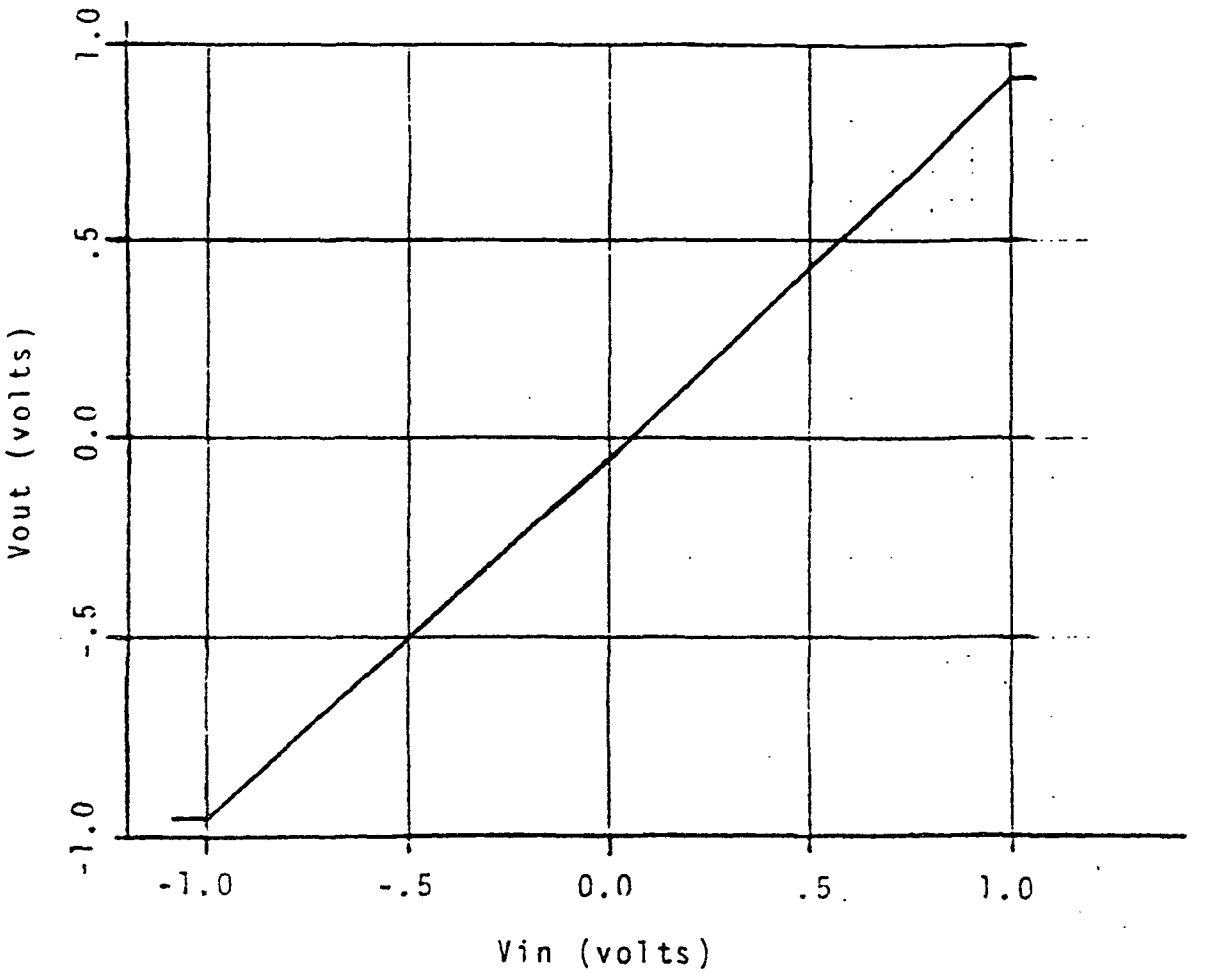


Figure 3-4 2920 Convertor Linearity

filter output before the D/A conversion. These PSF fixes appear in all the filter algorithms presented in this investigation. Five digital and four analog instructions in each forty-eight instruction block are dedicated to the version D fixes.

Test Algorithms

The test algorithms presented represent the extremes of the filter applications. The test set includes 300, 80, 70 and 10 Hz filters. All the algorithms were written using the cascade structure of Figure 2-5, and are part of the PSFSS library (see Appendix D). The 300, 80 and 70 Hz algorithms sample at 4.1K Hz. The 10 Hz algorithm is actually the 80 Hz algorithm using the sample rate division techniques to reduce the sample rate and corner frequency by a factor of eight. The 300 Hz and 10 Hz filters represent the extremes of the PSF corner frequencies used by NASA DFRF. The 70 Hz filter represents the approximate lower limit of the filter corner frequencies without using sample rate division techniques.

Magnitude and Phase Response Curves

The transfer functions of the algorithms as measured and plotted by the test set are shown in Figures 3-5, 3-6, 3-7 and 3-8. The expected transfer functions

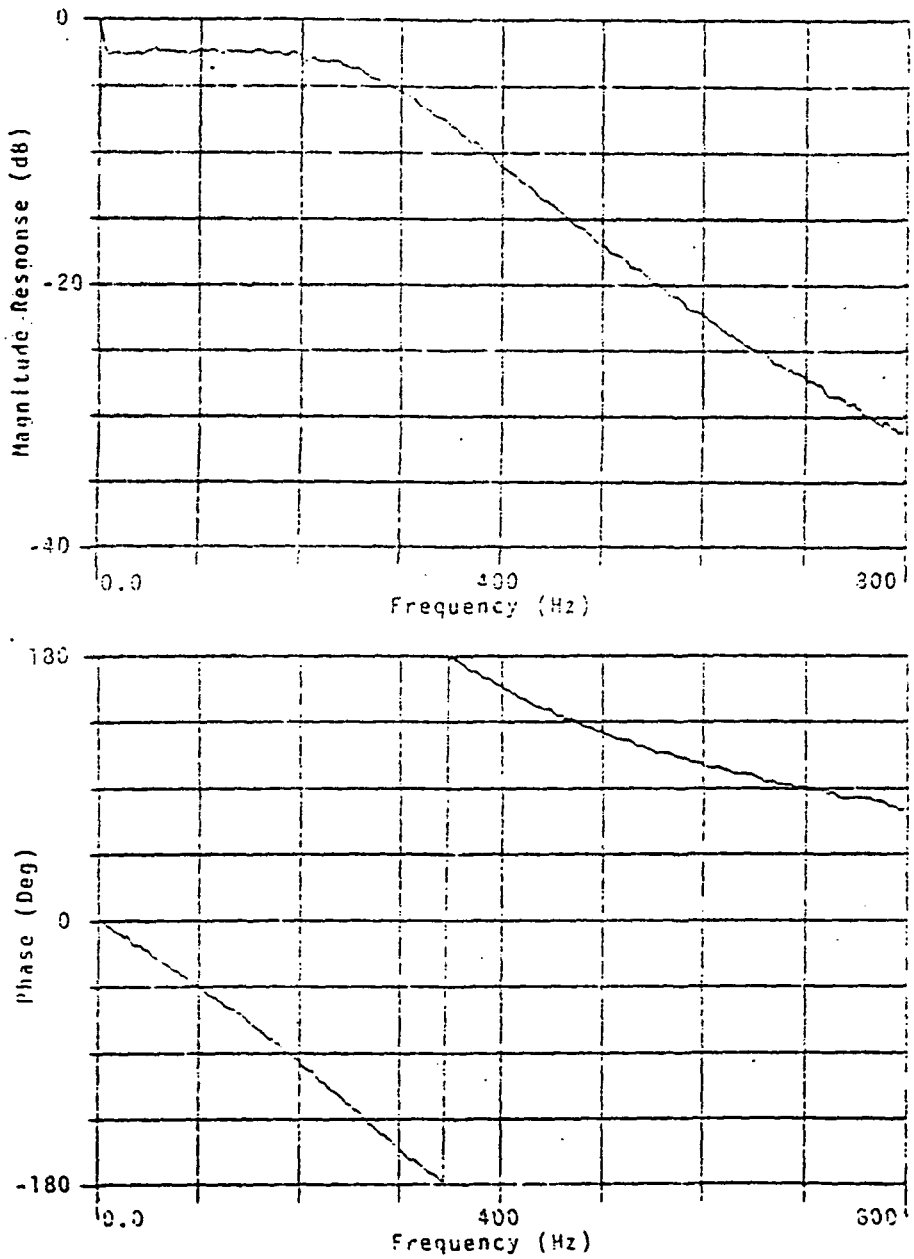


Figure 3-5 Magnitude and Phase Response of the 300 Hz Digital Filter

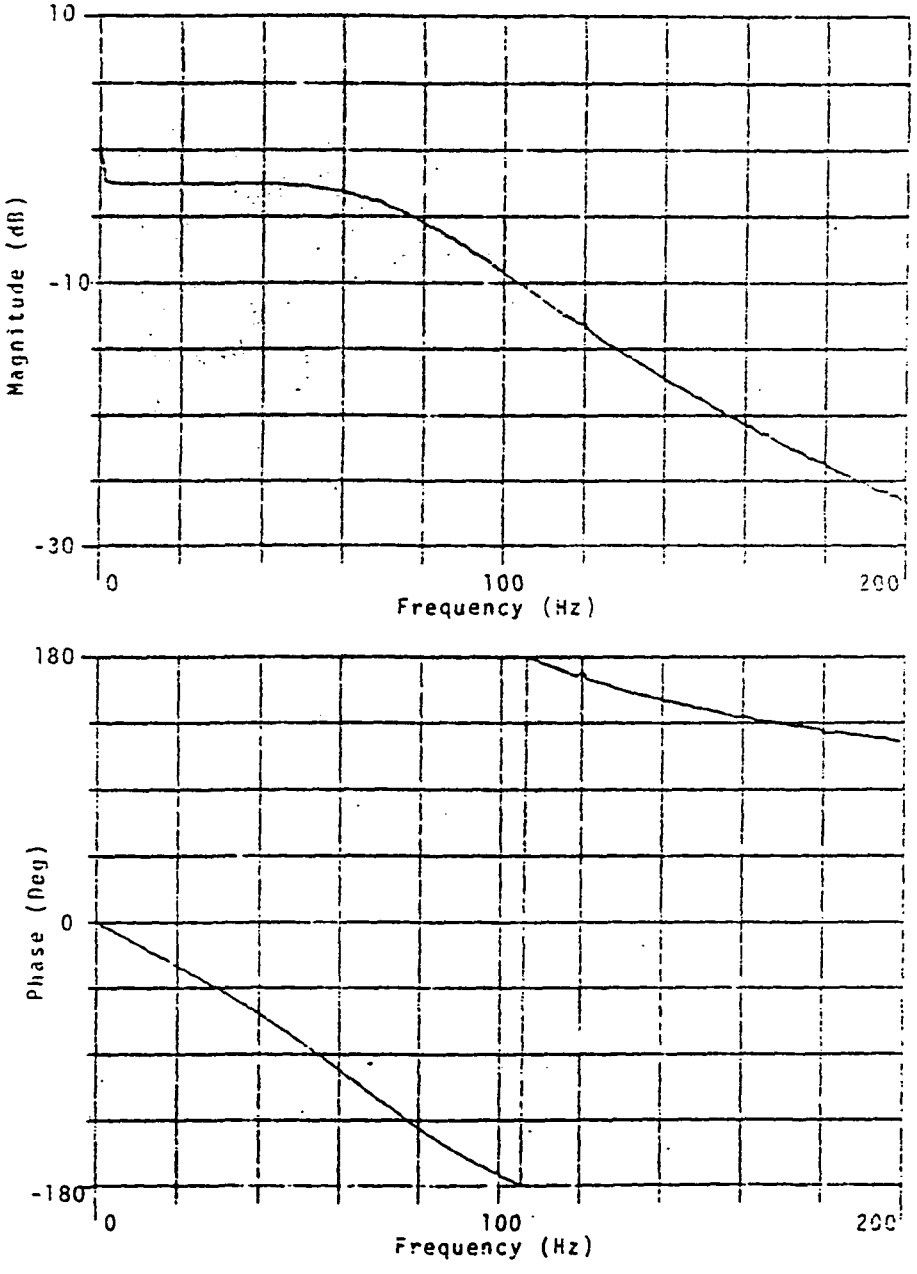


Figure 3-6 Magnitude and Phase Response of the 30 Hz Digital Filter

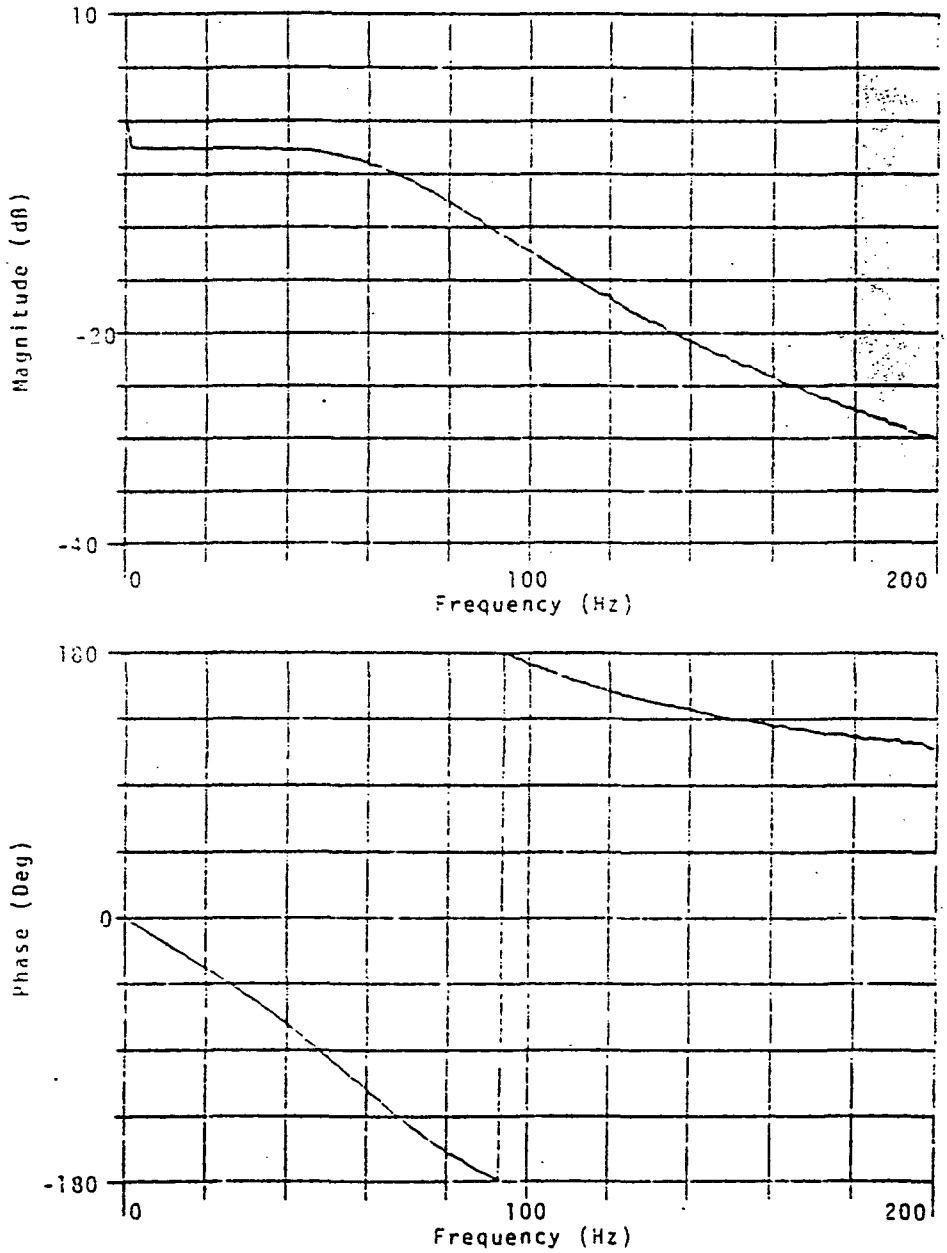


Figure 3-7 Magnitude and Phase Response of the 70 Hz Digital Filter

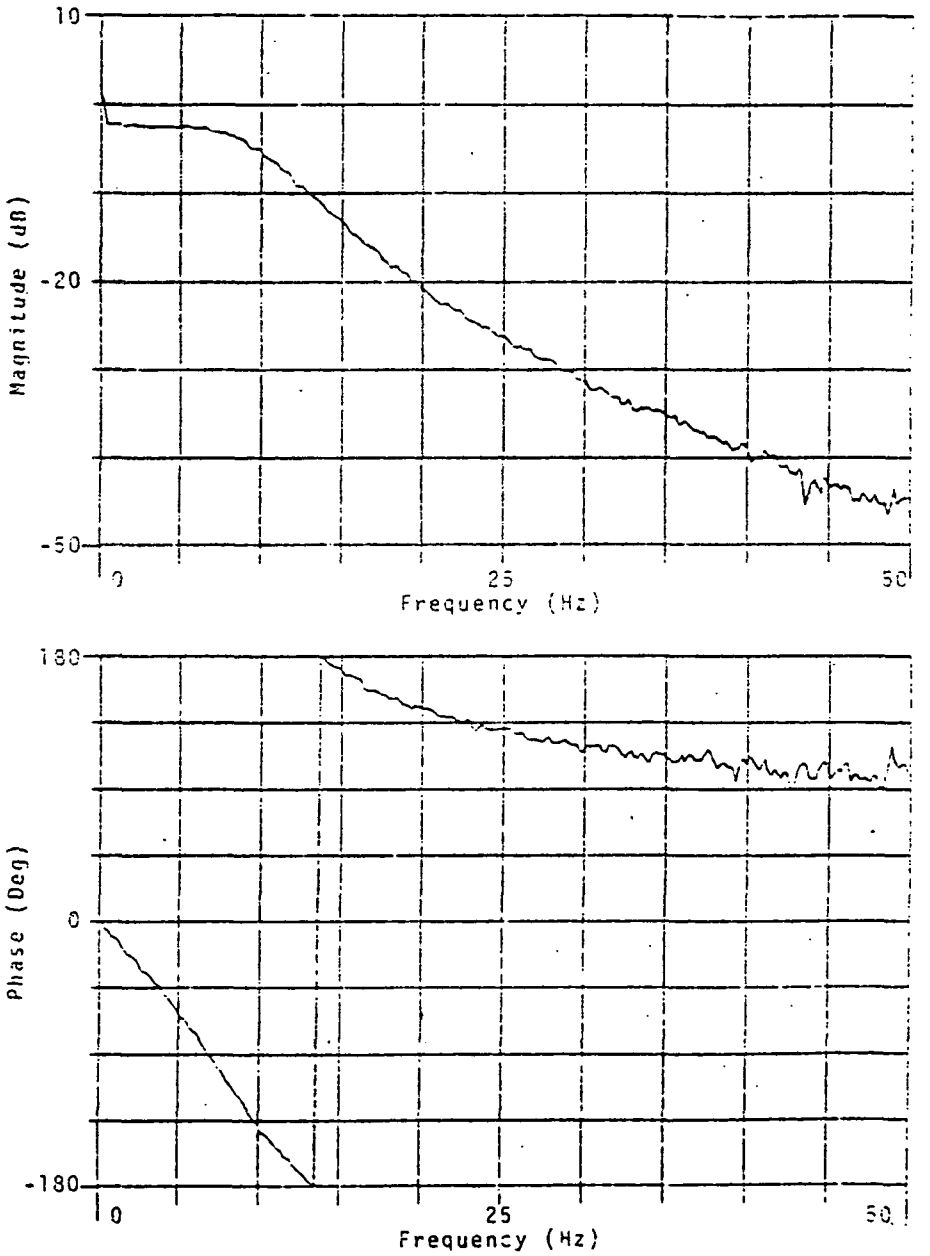


Figure 3-3 Magnitude and Phase Response of the 10 Hz Digital Filter

are shown in Figures 3-9, 3-10, 3-11, 3-12. The measured transfer functions correlate closely with the predicted transfer functions. The 2920 filter resolution is demonstrated in Figure 3-13. This plot shows the high frequency response of the 10 Hz filter. The filter rolls off till the magnitude of the output signal is less than the resolution of the DAC's. The result is the flat area in the stop band above 100 Hz. This plot also shows the effects of sample aliasing. A slight rise in the magnitude response may be observed at 200 Hz. This effect is shown more clearly in Figure 3-14. In this plot, the frequency scale of the 10 Hz filter transfer function has been expanded to show the pass bands centered at multiples of the sample rate. The peaks are caused by ADC frequency aliasing of the analog input signals.

Filter Repeatability

A demonstration of filter repeatability is shown in Figures 3-15 and 3-16. These plots are the transfer functions of two 70 Hz filters implemented on different 2920 chips. The frequency responses of these filters are for all practical purposes identical.

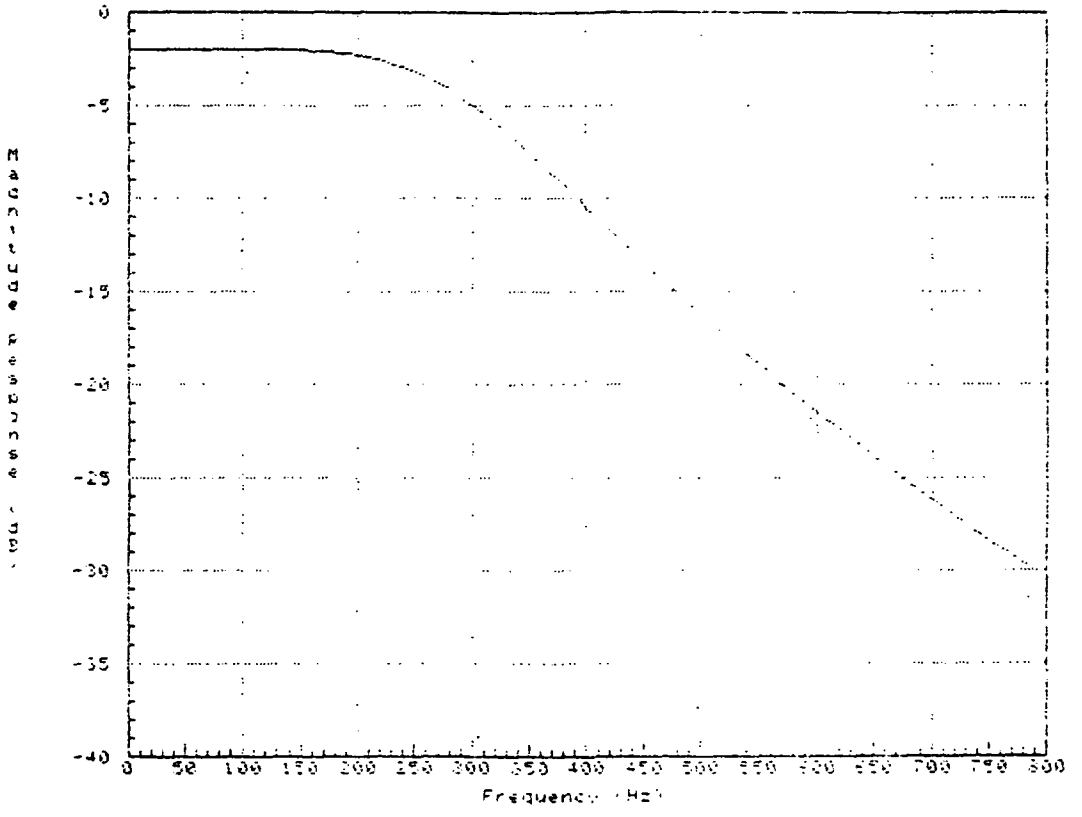


Figure 3-9 Calculated Magnitude Response of the 300 Hz Digital Filter

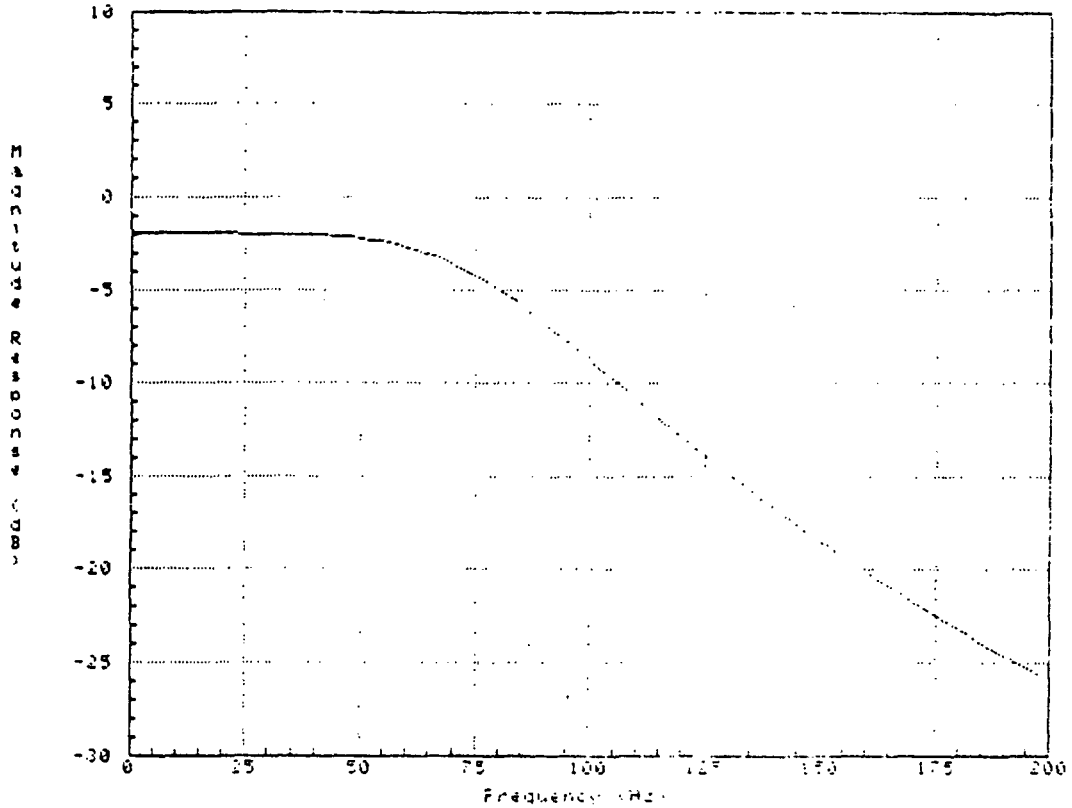


Figure 3-10 Calculated Magnitude Response of the 80 Hz Filter

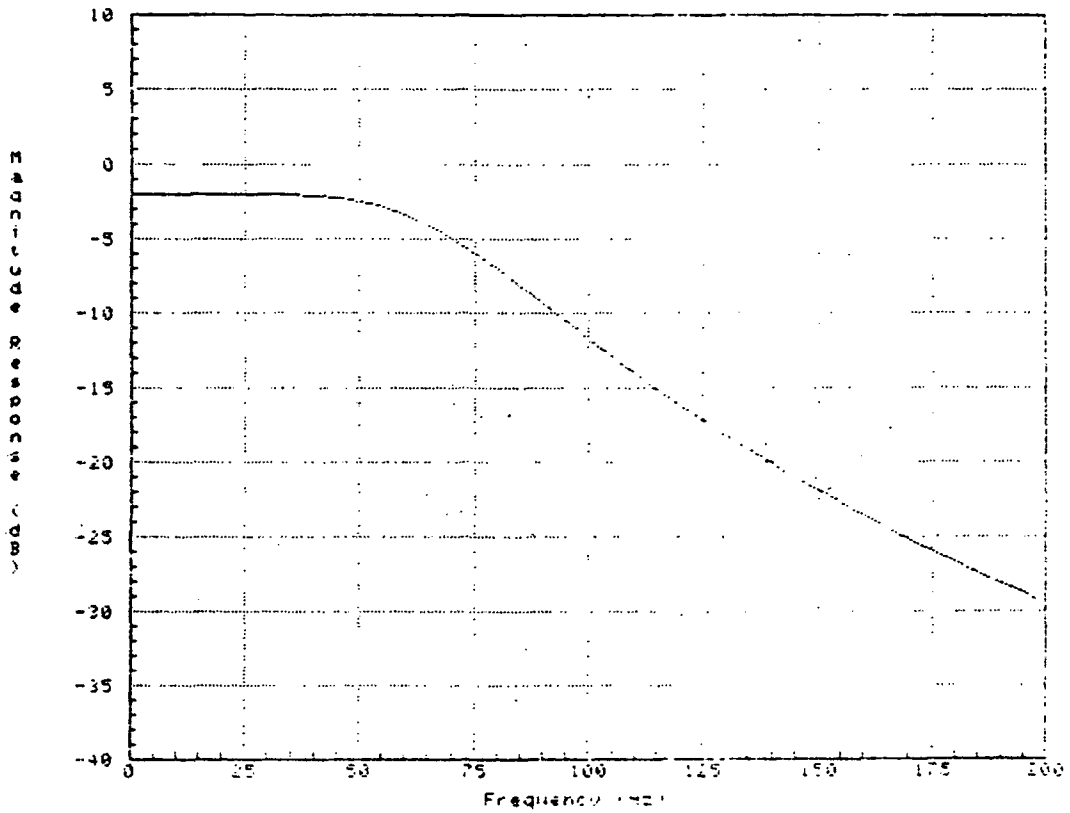


Figure 3-11 Calculated Magnitude Response of the 70 Hz Filter

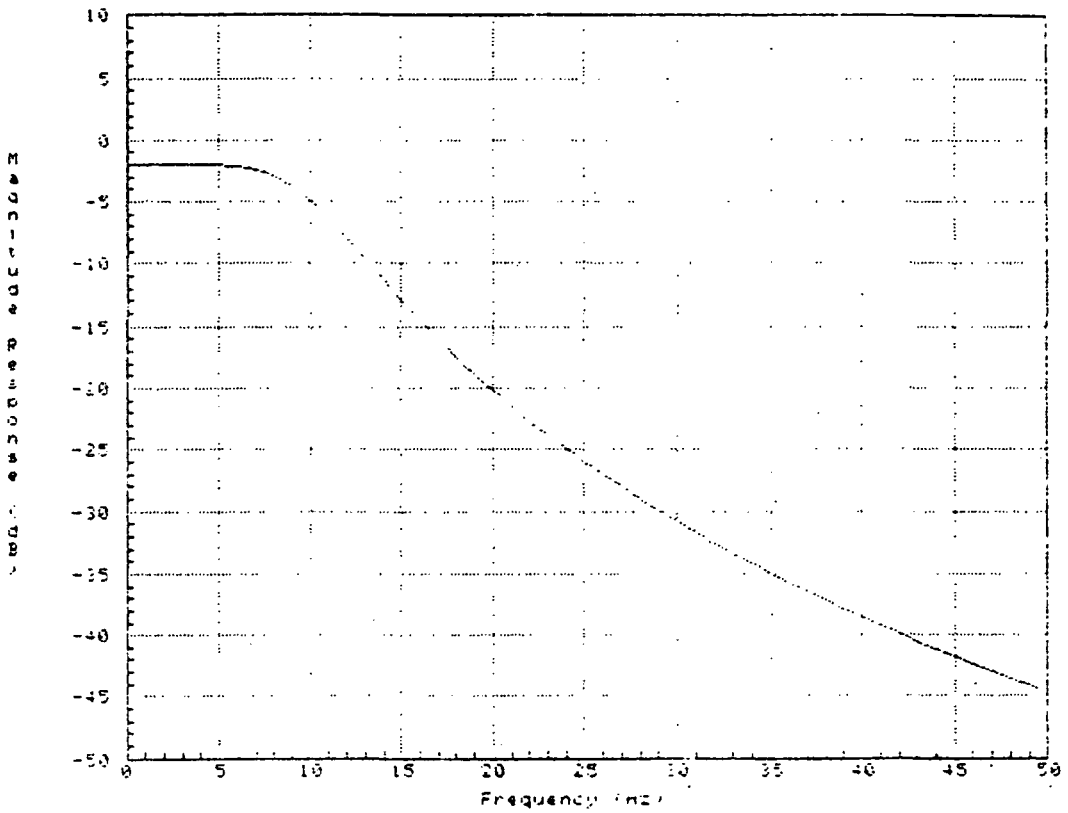


Figure 3-12 Calculated Magnitude Response of the 10 Hz Filter.

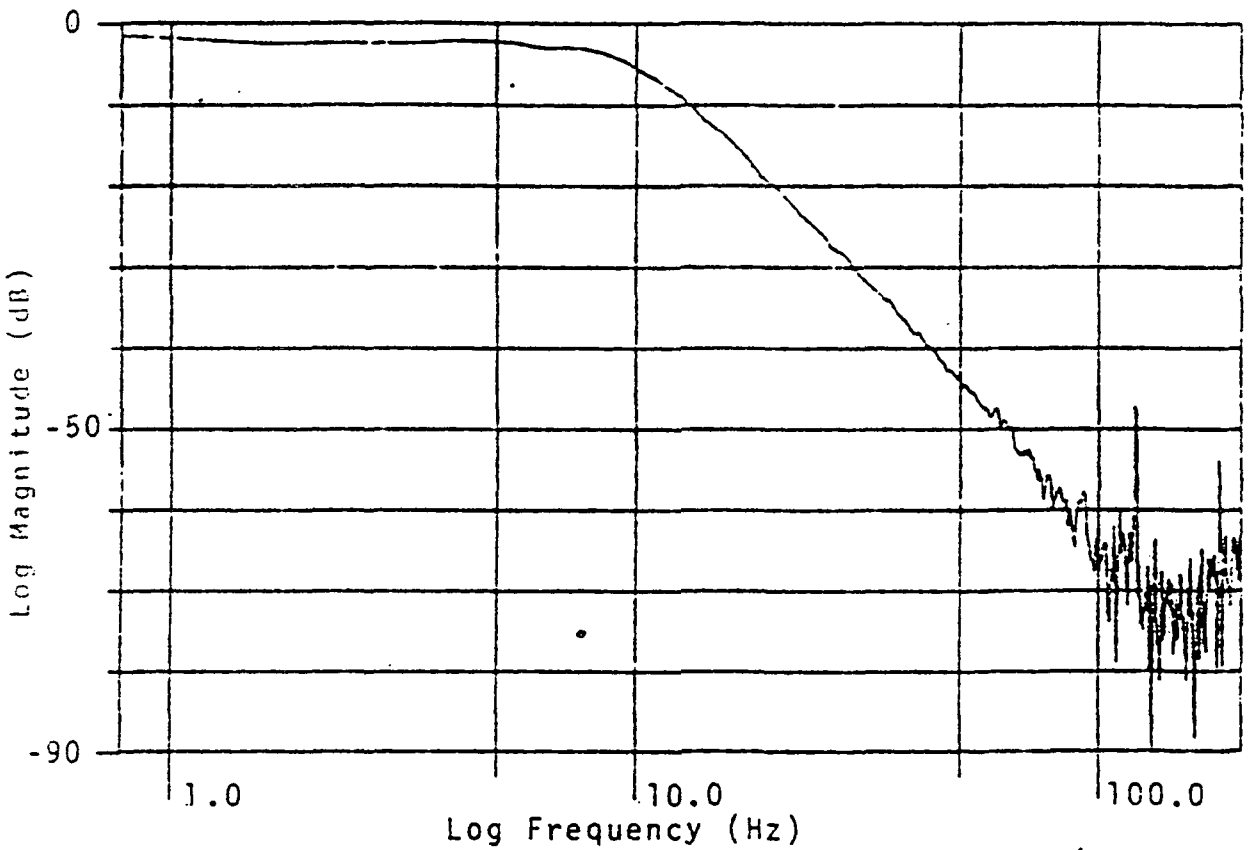


Figure 3-13 Magnitude Response of the 10 Hz Filter

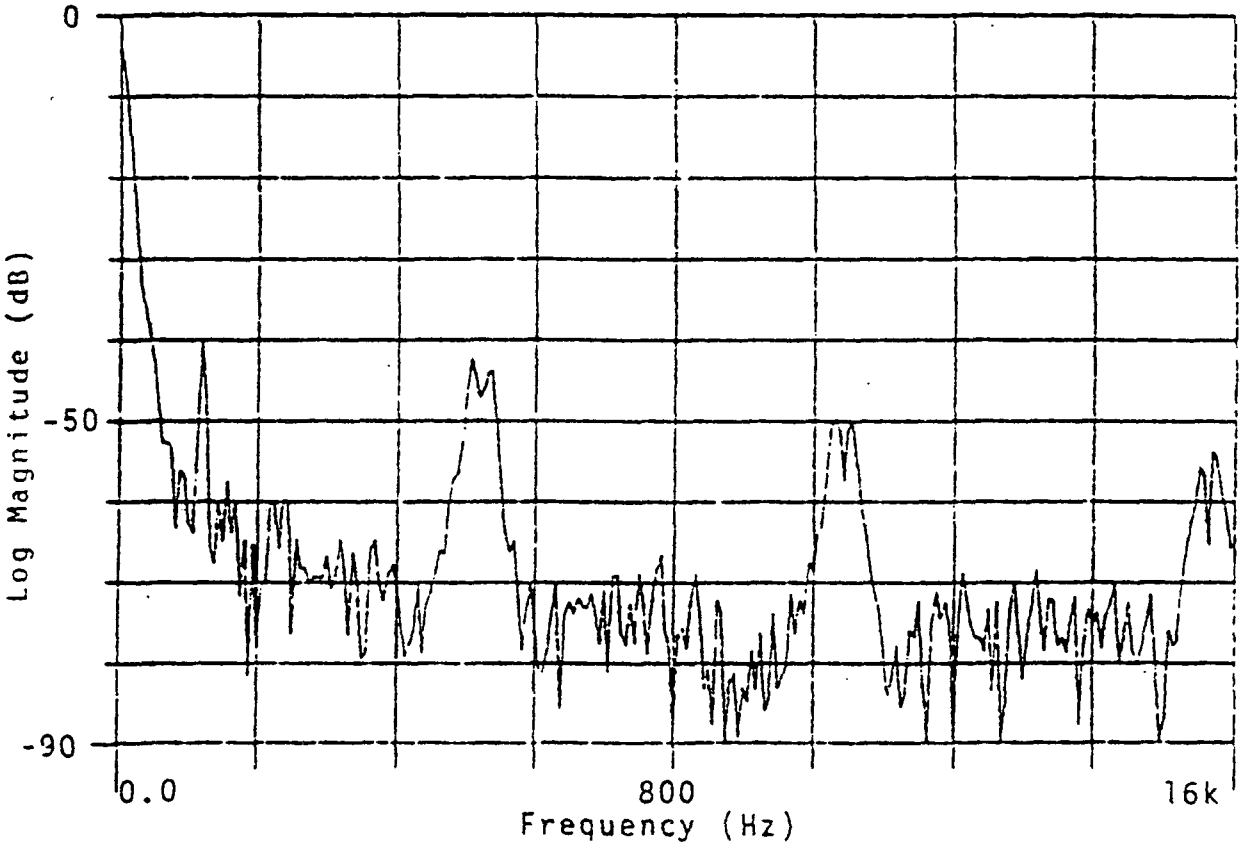


Figure 3-14 Aliasing in the 10 Hz Filter Magnitude Response

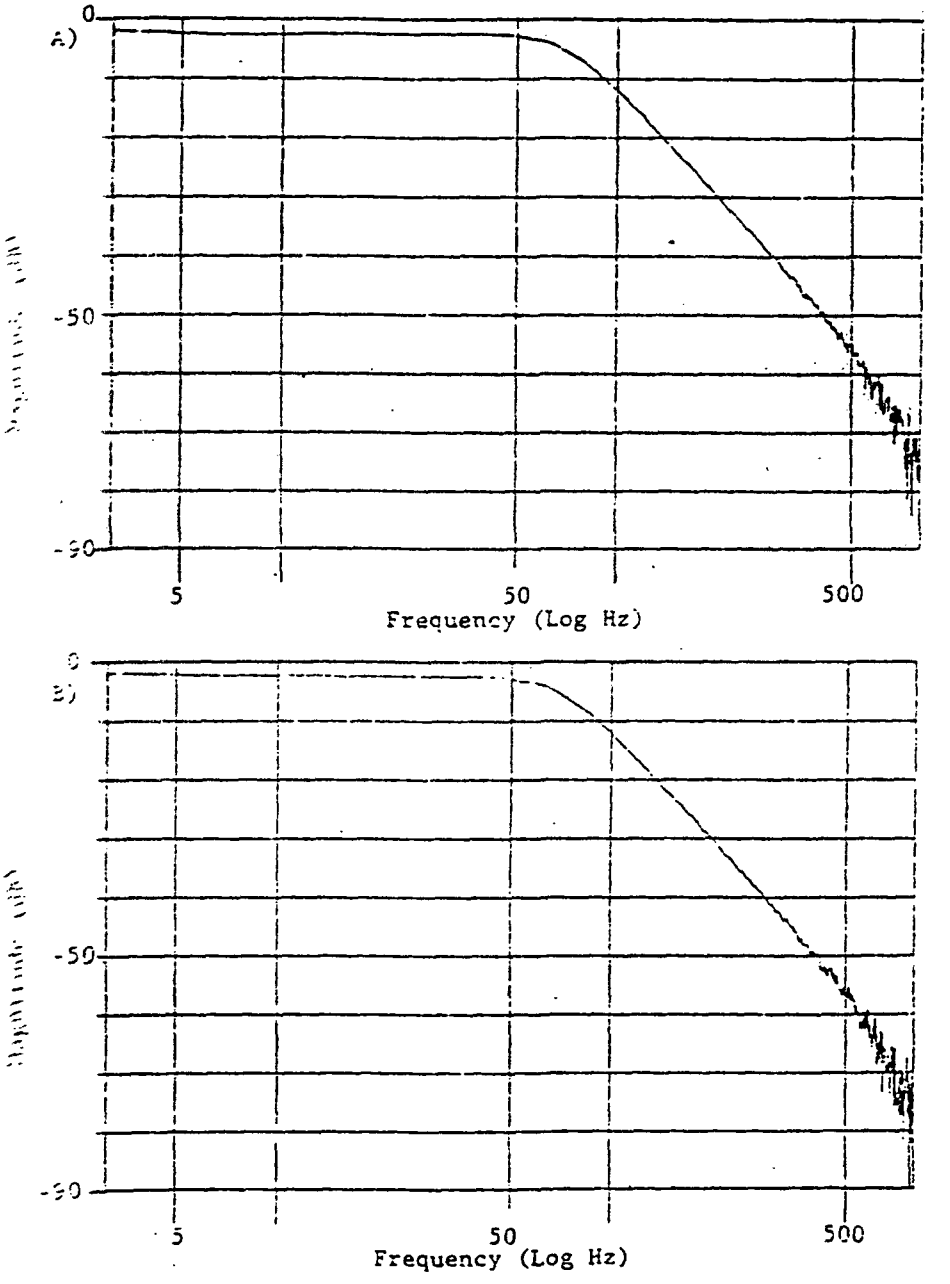


Figure 3-15 Comparison of Magnitude Response Curves for 70 Hz Filters

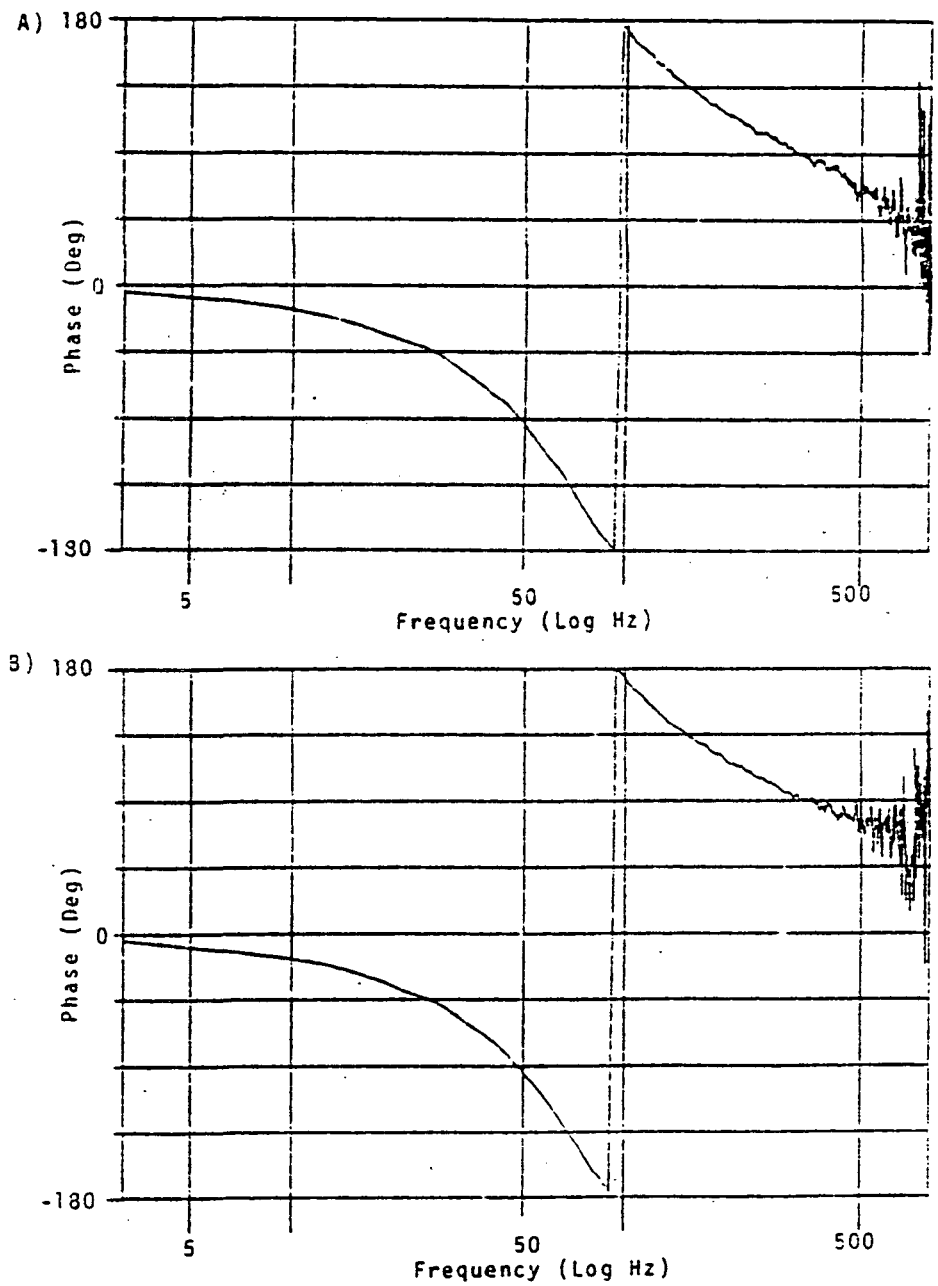


Figure 3-16 Comparison of Phase Response Curves for 70 Hz Filters

CHAPTER 4

CONCLUSION

Summary

The goal of this investigation was to evaluate the feasibility of replacing analog PCM PSF's with a digital equivalent based on Intel's 2920-D-18 Analog Signal Processor. The focus of the investigation was on the operational characteristics and limitations of the 2920 processor functioning as a PSF.

The design procedure established for this investigation starts with the analog PSF transfer function H(s). The Bilinear Transform Technique is applied to H(s) to derive the digital transfer function H(z). H(z) is then factored into its cascade form. Scale factors are assigned on the basis of individual filter section gain to prevent ALU overflows during processing. The resulting modified cascade transfer functions are coded into 2920 assembly language. These filter algorithms are then compiled and loaded into the 2920 EPROM memory. The filter algorithms were ground tested in a 2920 test circuit to determine the operating characteristics of the filter.

The test results presented in the previous chapter demonstrate that the 2920 is capable of performing the PSF function for limited resolution signals. The maximum signal resolution which the 2920 can process without some loss of signal accuracy is determined by the width of the 2920 A/D and D/A converters. This constraint limits the resolution of 2920 output signals to nine bits. There are several design 'tricks' which extend the resolution of the 2920, but the techniques are too hardware and software intensive to be used in this application.

The transfer functions of the digital filters roll off more sharply than the analog filters, and have excellent repeatability characteristics as the filter algorithm is implemented in different chips. The increased roll off rate is due to the "warping effect" of the Bilinear Transform. The frequency scale distortion in the H(s) to H(z) mapping is desirable in this application. The increased noise rejection in the transition band is realized without disturbing the smoothness of the Butterworth shape. The transportability of filter algorithms is a major advantage of the digital filter.

The filter algorithms were written in such a form that a computer program can handle all the code linking and compiling operations normally associated with digital programs. The filter transfer functions are independent

of which 2920 chip or where in memory it is implemented. Therefore, the algorithm development process can be performed in a uniform manner for any 2920 chip. The PSFSS programs are a result of this characteristic of digital processing. The PSFSS tool simplifies the installation of a 2920 PSF modification to a five minute operation. The only information needed to specify a PSF is the desired corner frequency. The power in this type of tool lies both in the simplicity and the flexibility. The PSFSS software manages signal processing modules. The function of the modules is not limited to PSF filters. Modules could be included which perform any desired function (waveform generation, linearization, etc.). This flexibility could allow many of the tasks normally associated with ground processing to be performed real time in flight.

The size of the 2920 based PSF circuitry is estimated to be 2.25 square inches per filter. This is comparable to the size of the analog circuit. This estimate is based on the circuit shown in Figure B-1, and the implementation of four filters per 2920 PSF circuit. The potential exists for significantly reducing the size of the circuit.

2920 Version J

The major design limitation encountered in this investigation was the length of the 2920 program space. Implementing four filters per chip limited the number of instructions for each filter algorithm to forty-eight. This design constraint becomes more and more imposing as the ratio R_{sc} (F_s/W_c) increases. As R_{sc} increases, the accuracy requirements of the multiplication algorithms also increase. More accurate multiplications require more instruction steps. If R_{sc} is large, there simply is not enough room to implement the algorithm within the forty-eight program step limit. The 70 Hz filter in the PSFSS library has a Rsc equal to 9.5 Hz/Rad. The upper limit of R_{sc} which can be implemented within the size and resolution constraints is approximately ten for the 2920-D. The filter algorithms with corner frequencies below 70 Hz ($R_{sc} > 10.0$) were created using the sample rate division techniques. The value of R_{sc} for these filters was too large to allow coding a filter which used the base sample rate (4166.7 Hz). Intel has released a newer version of the 2920 which does not require the software fixes required by the version D 2920 used in this investigation. The I/O sequences version J processor have been fixed thus increasing the number of instructions available for calculations. The impact of the reduced algorithm overhead of the version J is to increase the flexibility of algorithm design.

Areas for Further Study

This investigation demonstrated the characteristics of the 2920 PSF transfer functions. But the effect of environmental stresses on the critical 2920 support circuits has not been established. Further study and tests are required to determine an optimum design for the support hardware and the impact of environmental stress on the 2920 PSF circuitry.

The function of the analog PSF in the PCM system is to remove high frequency noise in signals before sampling. The introducing of a digital equivalent for the analog filters creates an interesting anomaly in the system. Analog signals are processed by the 2920 circuit SCB's to remove any high frequency noise, then sampled by the 2920. The 2920 processes the digitized signal, then converts the signal back to analog so it can be sampled by the PCM system. The PCM sampler has a digital PSF (the 2920) which has its own PSF (the SCB's). Further, the signal is converted from an analog to a digital twice. The extra PSF and conversion could be eliminated by using the SCB circuits as PSF's for the PCM sampler and performing the digital filter processing function on the output of the PCM sampler. Further study is needed to determine the feasibility of eliminating the extra PSF circuit and A/D, D/A conversion process present in the 2920 based PSF scheme.

BIBLIOGRAPHY

Manuals

2920 Assembly Language Manual. Ed. Literature Department. Santa Clara: Intel Corporation, 1979.

2920 Simulator User's Guide. Ed. Literature Department. Santa Clara: Intel Corporation, 1979.

Technical Reference

The TTL Data Book. Dallas: Texas Instruments, Inc., 1976.

2920-10 Signal Processor. Ed. Literature Department. Santa Clara: Intel Corporation, October 1979.

2920 Support Package. Ed. Literature Department. Santa Clara: Intel Corporation, October 1979.

2920 Analog Signal Processor Design Handbook. Ed. Literature Department. Santa Clara: Intel Corporation, August 1980.

2920 Software Support Package. Ed. Literature Department. Santa Clara: Intel Corporation, September 1980.

2920-16 Signal Processor. Ed. Literature Department. Santa Clara: Intel Corporation, October 1980.

Periodicals

Holm, Robert E. "Implement Complex Analog Filters with a Real-Time Signal Processor." EDN, November 1979.

---------- and Charles T. Yaeger "First Single-chip Processor Simplified Analog Design Problems." Electronic Design, September 1979.

Vernia, Carmel "2920 Status." Signal Processing News, January 1982.

---------- "Signal Processing Products Status." Signal Processing News, April 1982.

---------- "Signal Processing Status." Signal Processing News, August 1982.

Books

Antoniou, Andreas. Digital Filters: Analysis and Design. United States: McGraw-Hill, Inc., 1979.

Budak, Aram. Passive and Active Network Analysis and Synthesis. Boston: Houghton Mifflin Company, 1974.

Dorf, Richard C. Modern Control Systems. 2nd Revised Ed. Philippines: Addison-Wesley Publishing Company, Inc., 1974.

Franklin, Gene F. and Powell, David. Digital Control of Dynamic Systems. Philippines: Addison-Wesley Publishing Company, Inc., 1980.

Gold, Bernard, and Rabiner, Lawrence R. Theory and Application of Digital Signal Processing. New Jersey: Prentice Hall, Inc., 1975.

Rabiner, L.R. and Schafer, R. W. Digital Processing of Speech Signals. Ed. Alan V. Oppenheim. United States: Prentice Hall, Inc., 1978.

Stremler, Ferrel G. Introduction to Communication Systems. Ed. Cheng, David; Gould, Leonard; and Fred Manasse. Philippines: Addison-Wesley Publishing Company, Inc., 1977.

APPENDIX A

TRANSFER FUNCTION GRAPHS

Appendix A contains the measured transfer functions of the 2920 algorithms contained in the PSFSS filter library. The transfer functions were measured and plotted using the test set described in Chapter 3. Figures A-1 through A-14 are the magnitude and phase response plots of filter algorithms created by the PSFSS support tool to implement the desired corner frequencies.

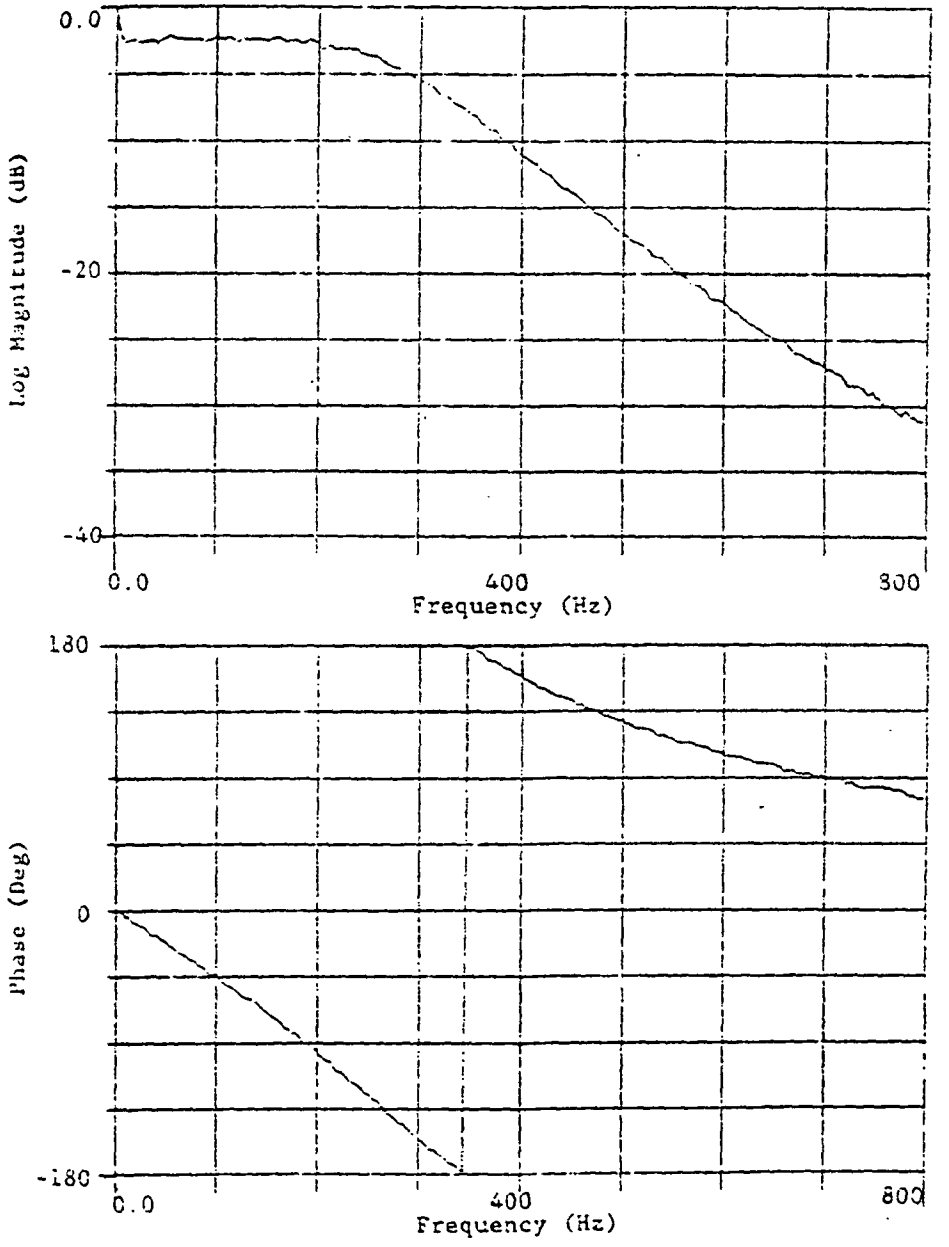


Figure A-1 Magnitude and Phase Response of the 300 Hz Digital Filter

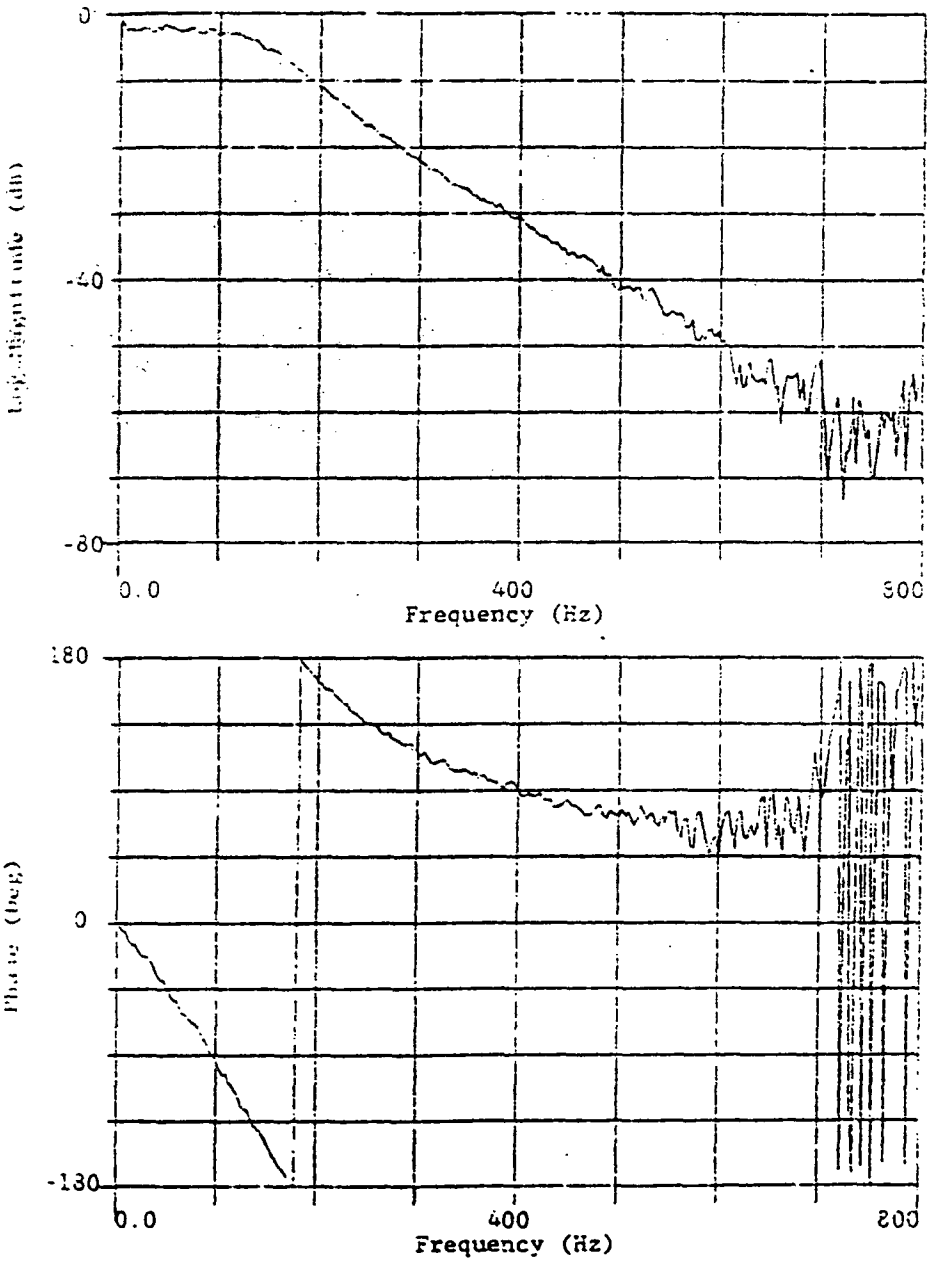


Figure A-2 Magnitude and Phase Response of the 150 Hz Digital Filter

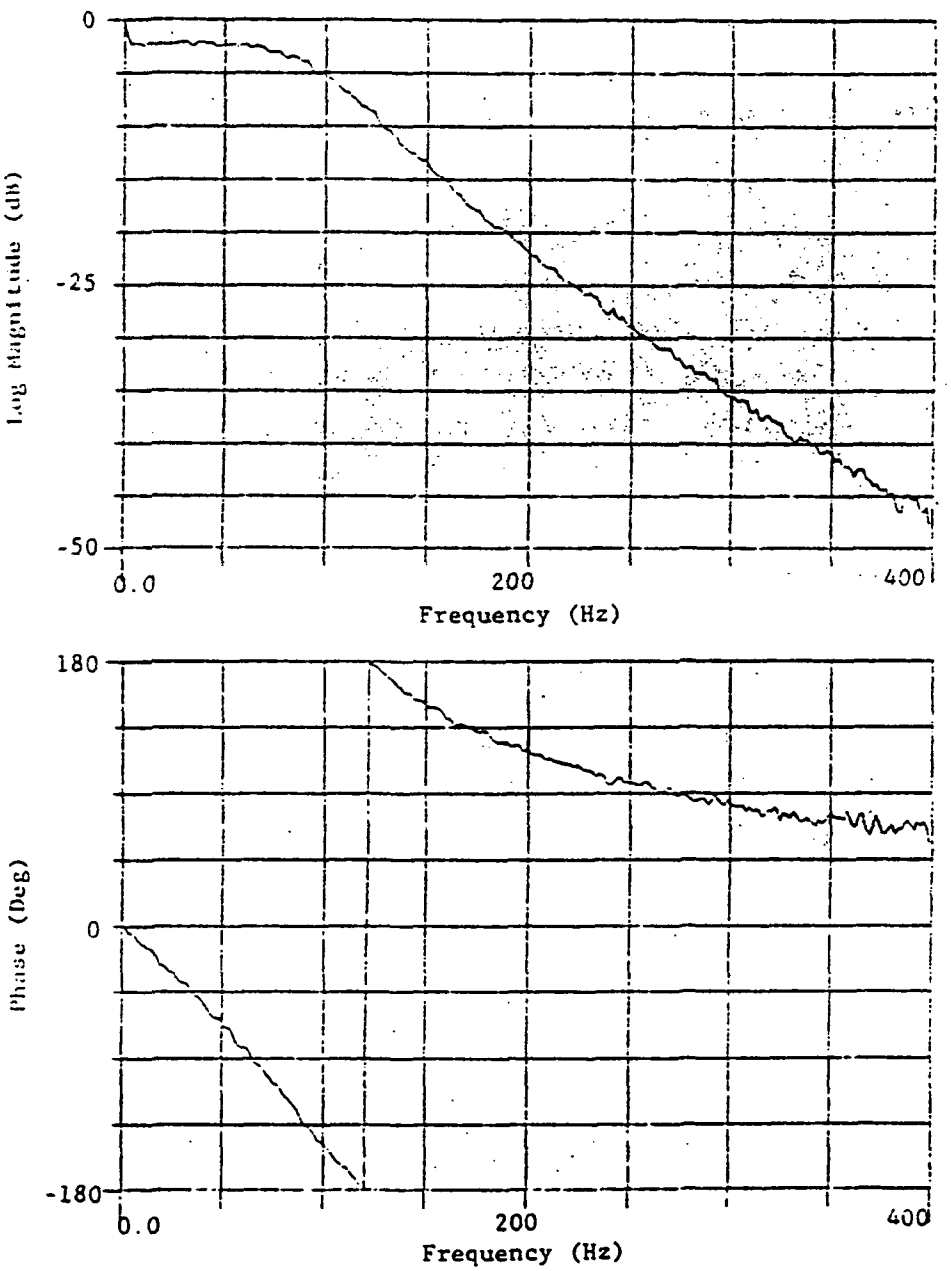


Figure A-3 Magnitude and Phase Response of the 100 Hz Digital Filter

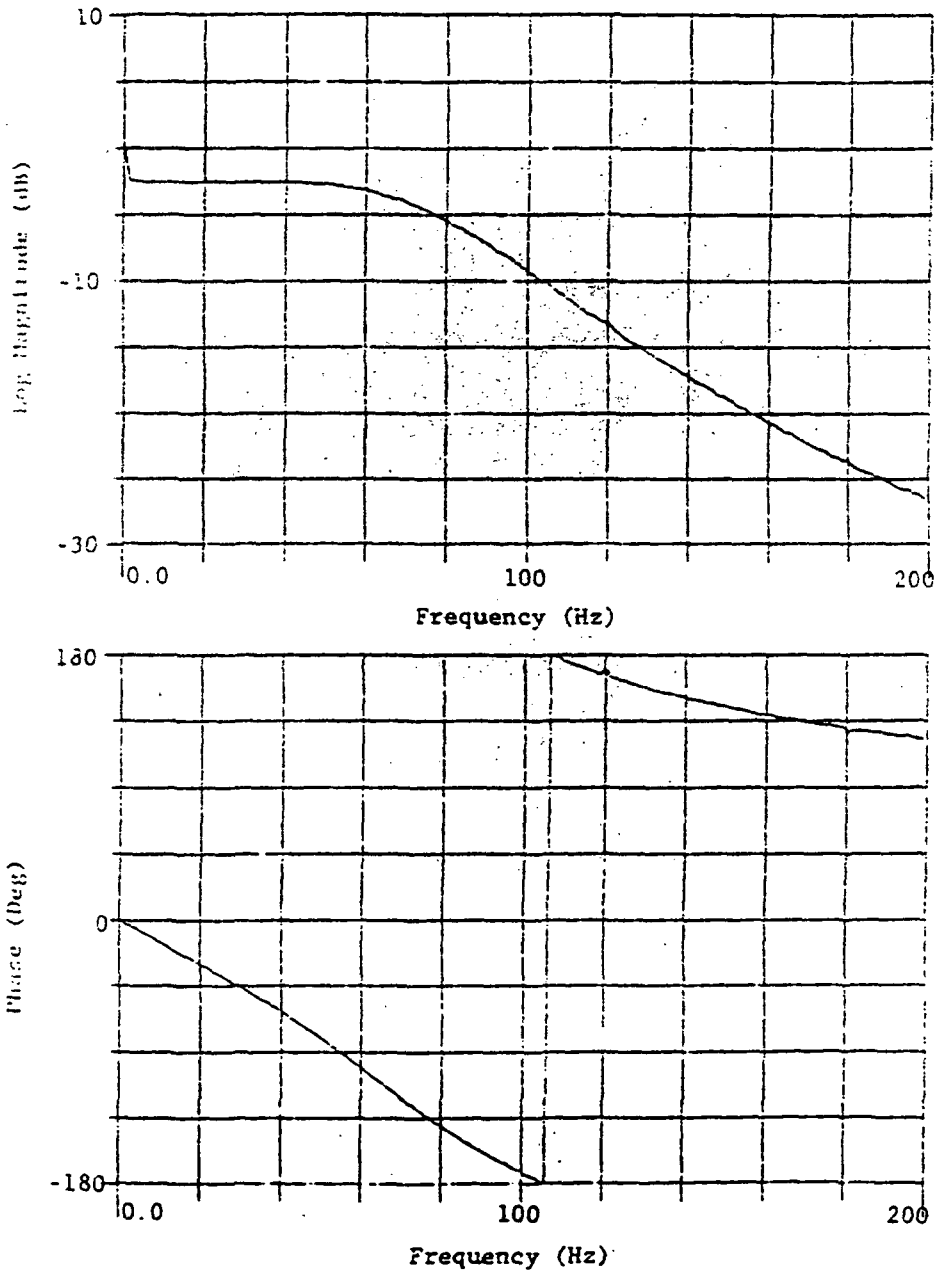


Figure A-4 Magnitude and Phase Response of the 80 Hz Digital Filter

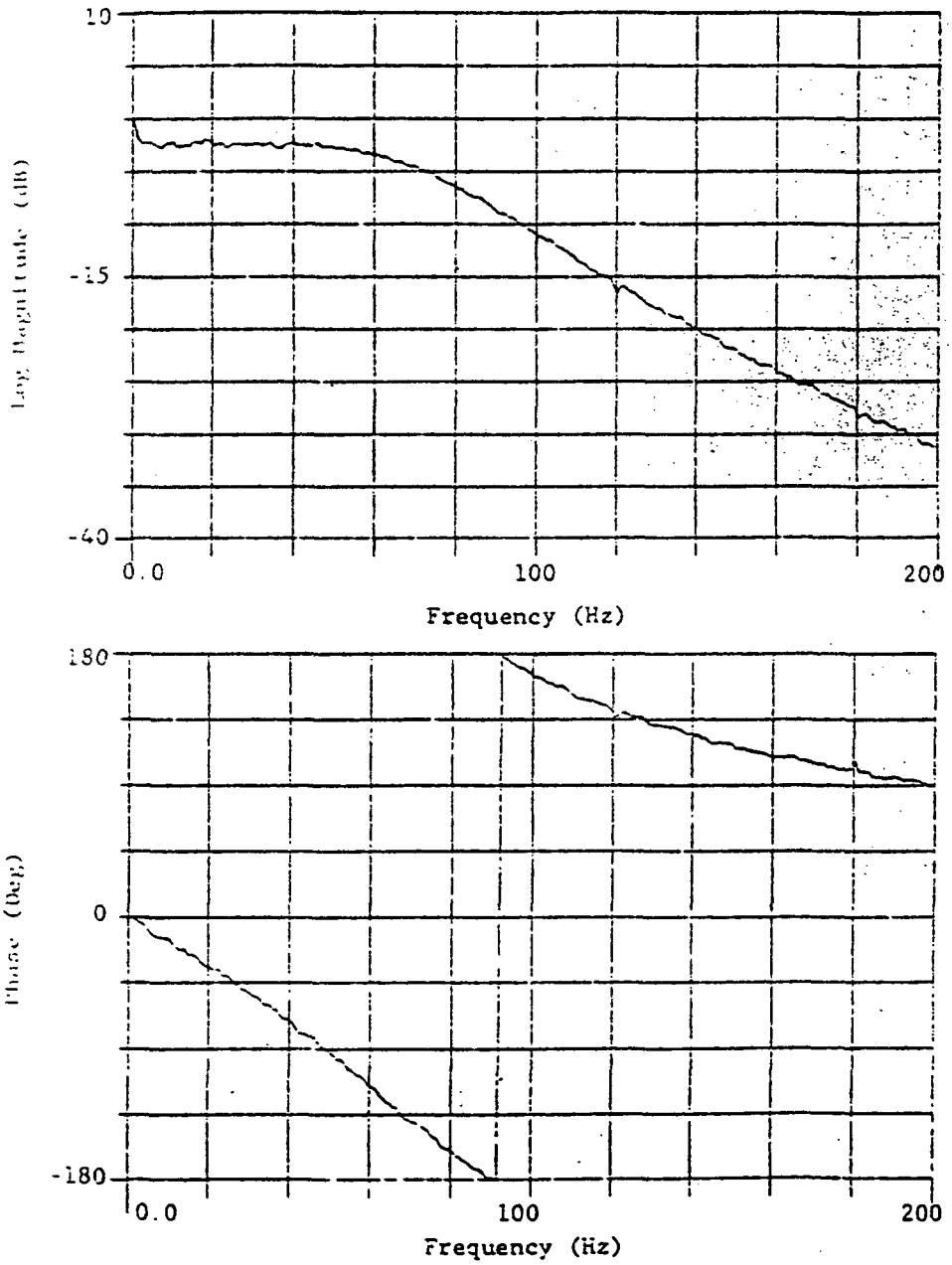


Figure A-5 Magnitude and Phase Response of the 75 Hz Digital Filter

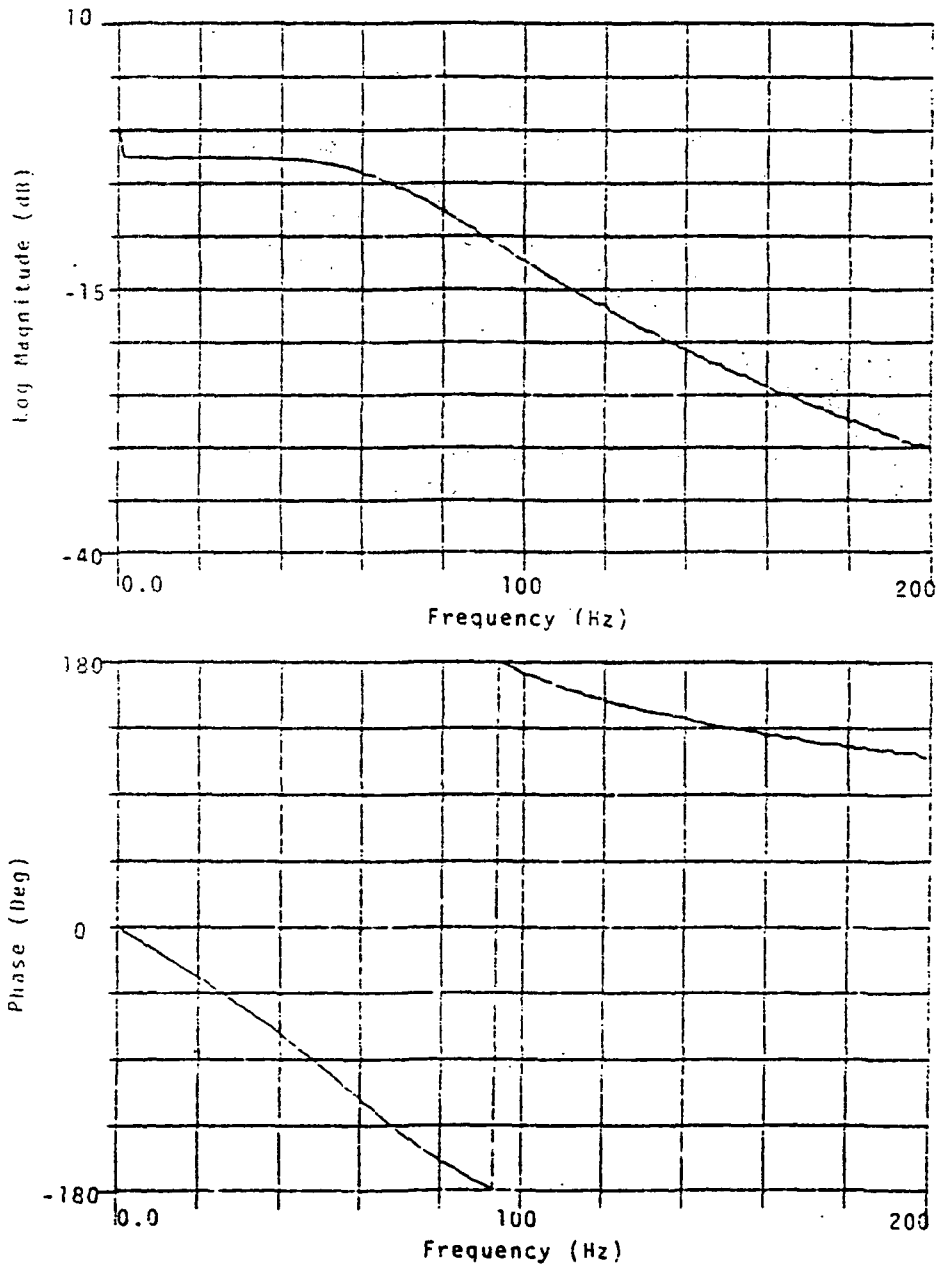


Figure A-6 Magnitude and Phase Response of the 70 Hz Digital Filter

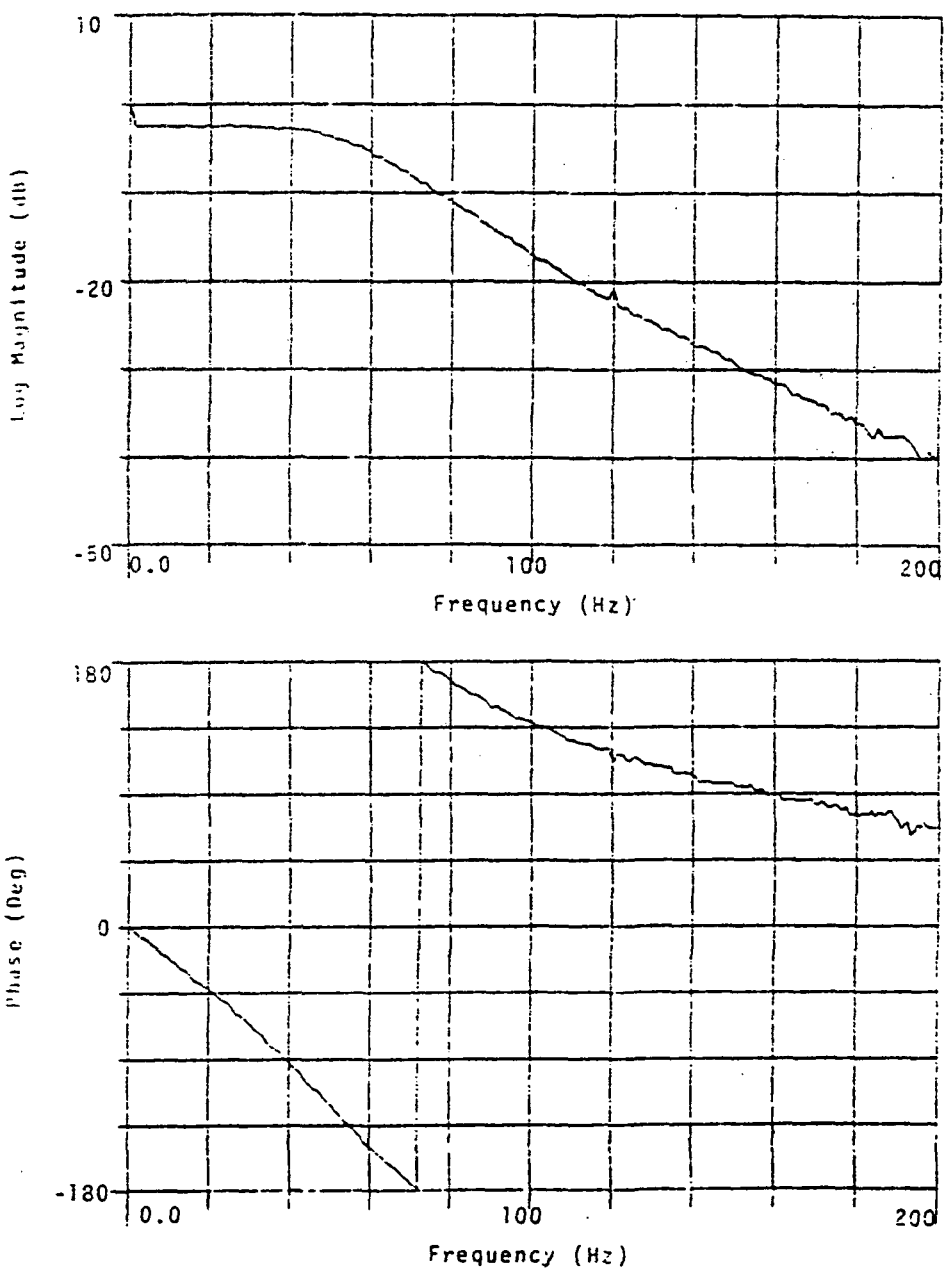


Figure A-7 Magnitude and Phase Response of the 60 Hz Digital Filter

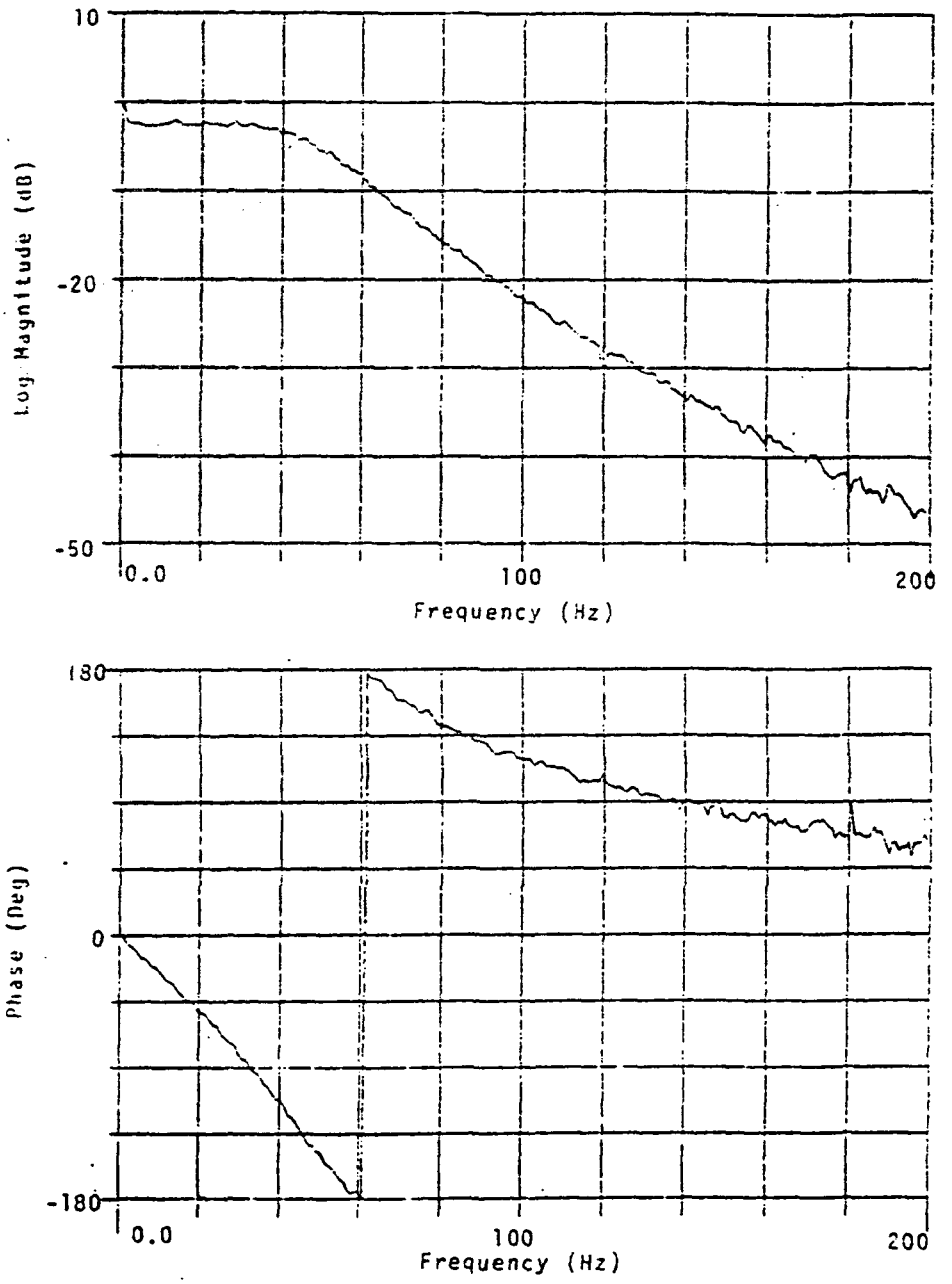


Figure A-8 Magnitude and Phase Response of the 50 Hz Digital Filter

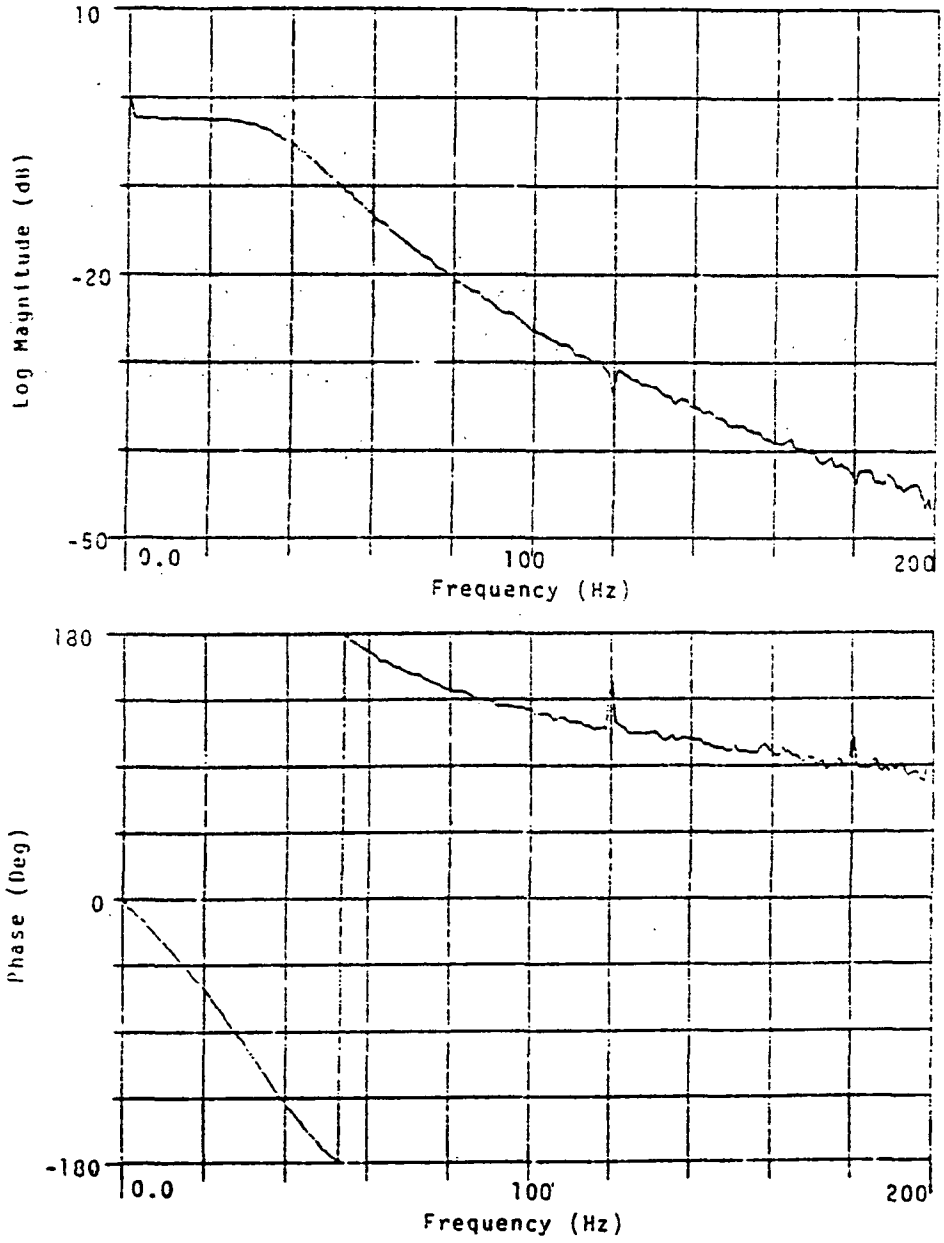


Figure A-9 Magnitude and Phase Response of the 40 Hz Digital Filter

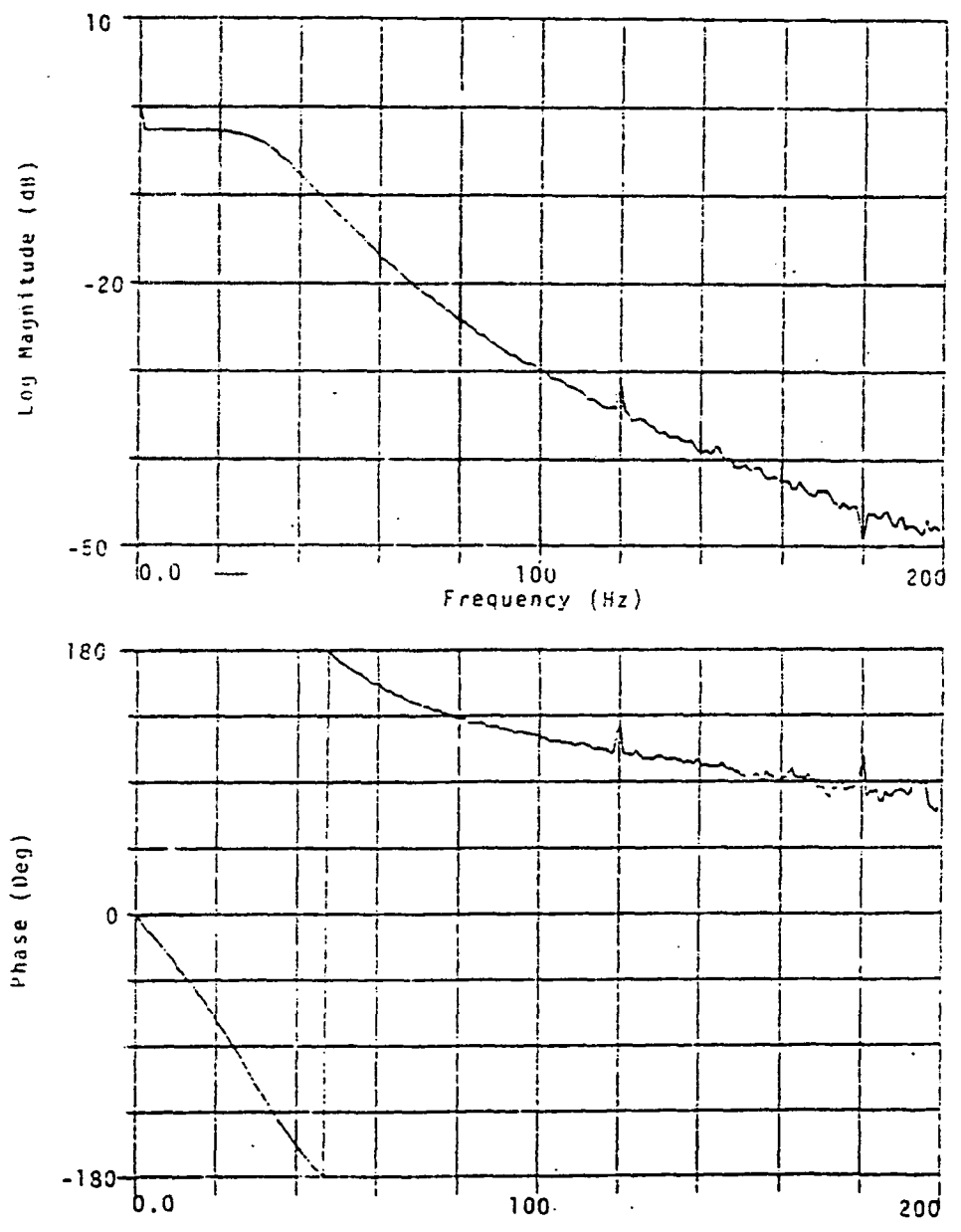


Figure A-10 Magnitude and Phase Response of the 35 Hz Digital Filter

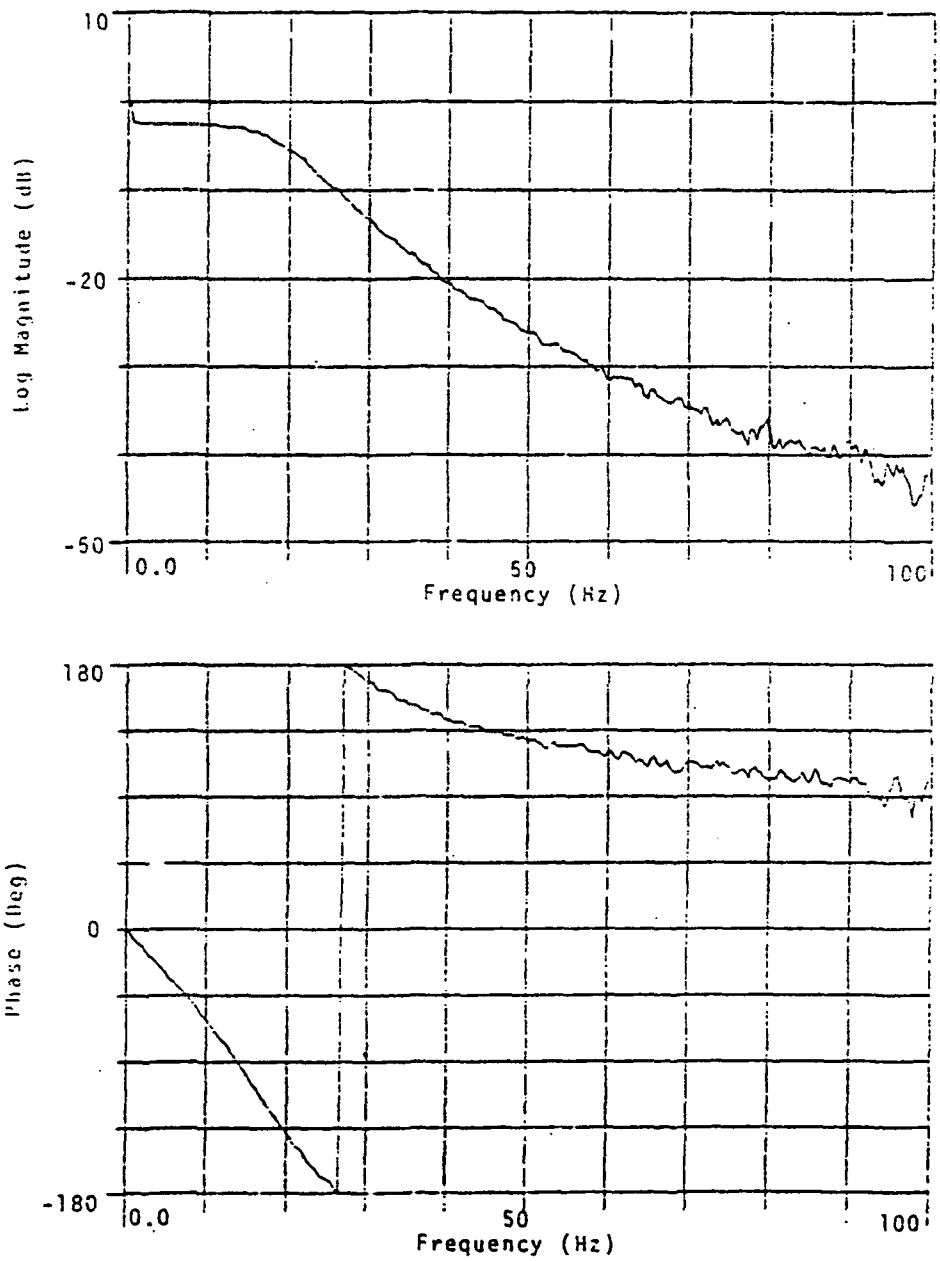


Figure A-11 Magnitude and Phase Response of the 20 Hz Digital Filter

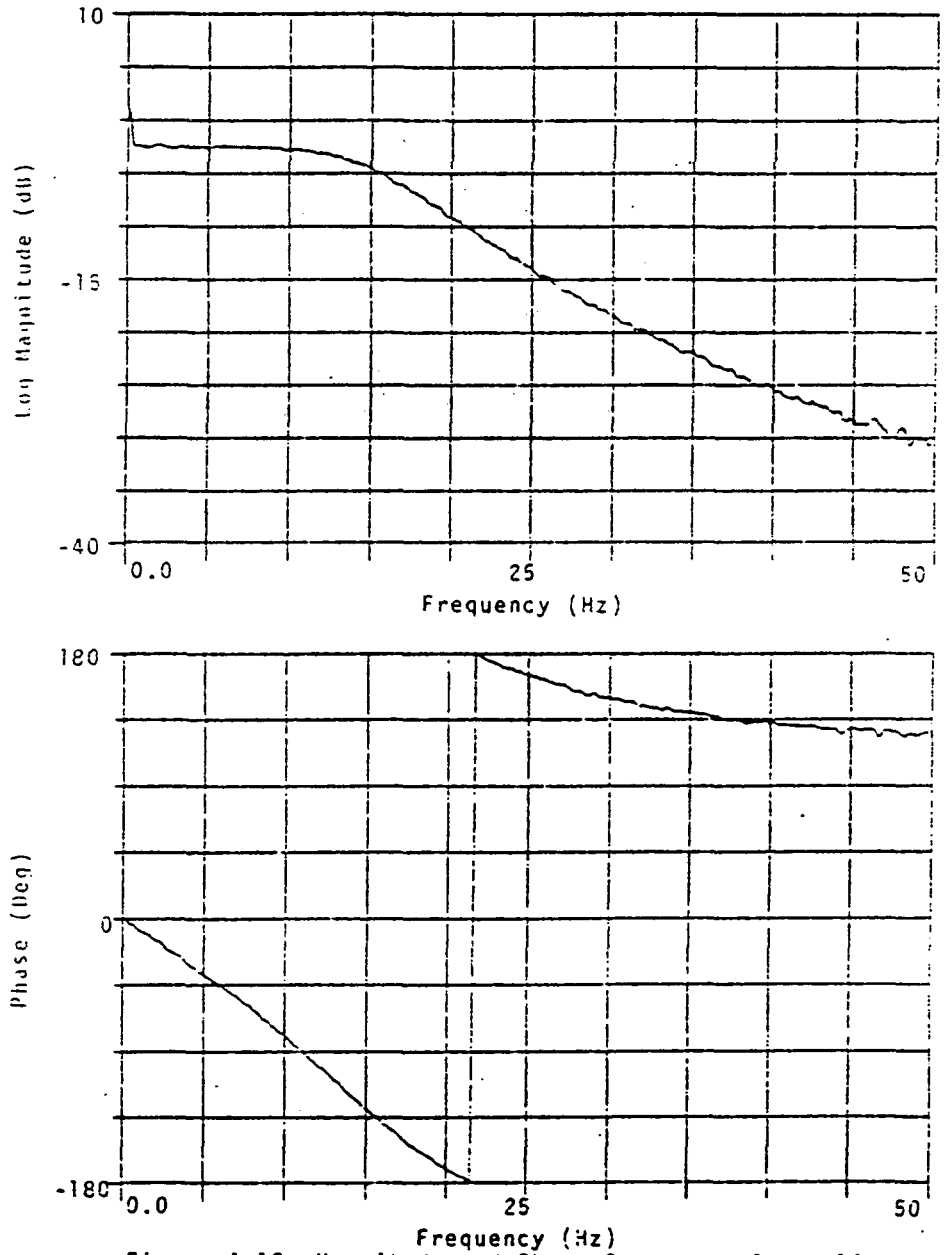
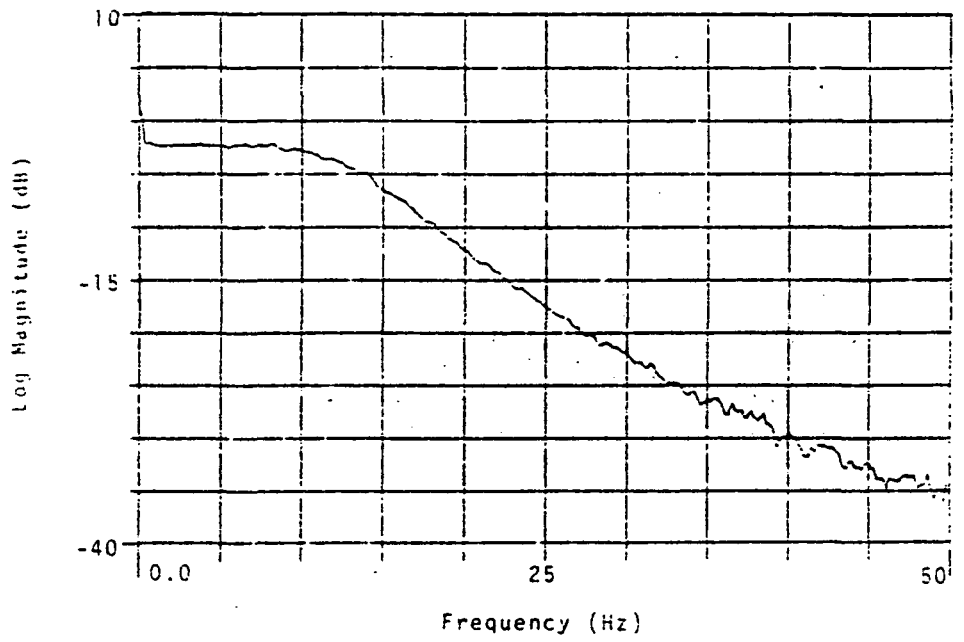


Figure A-12 Magnitude and Phase Response of the 16 Hz Digital Filter



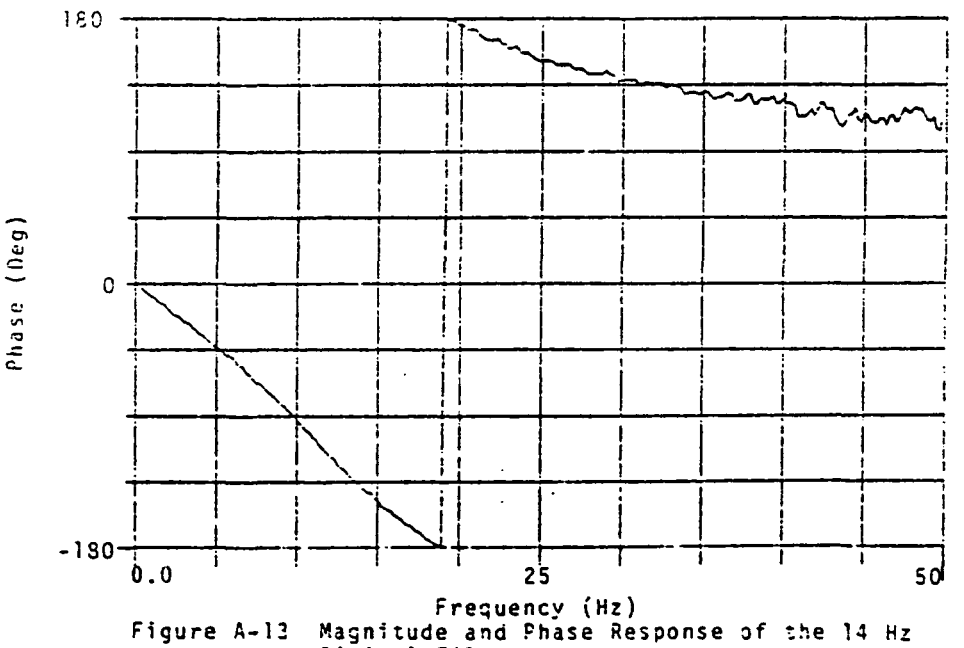


Figure A-13 Magnitude and Phase Response of the 14 Hz Digital Filter

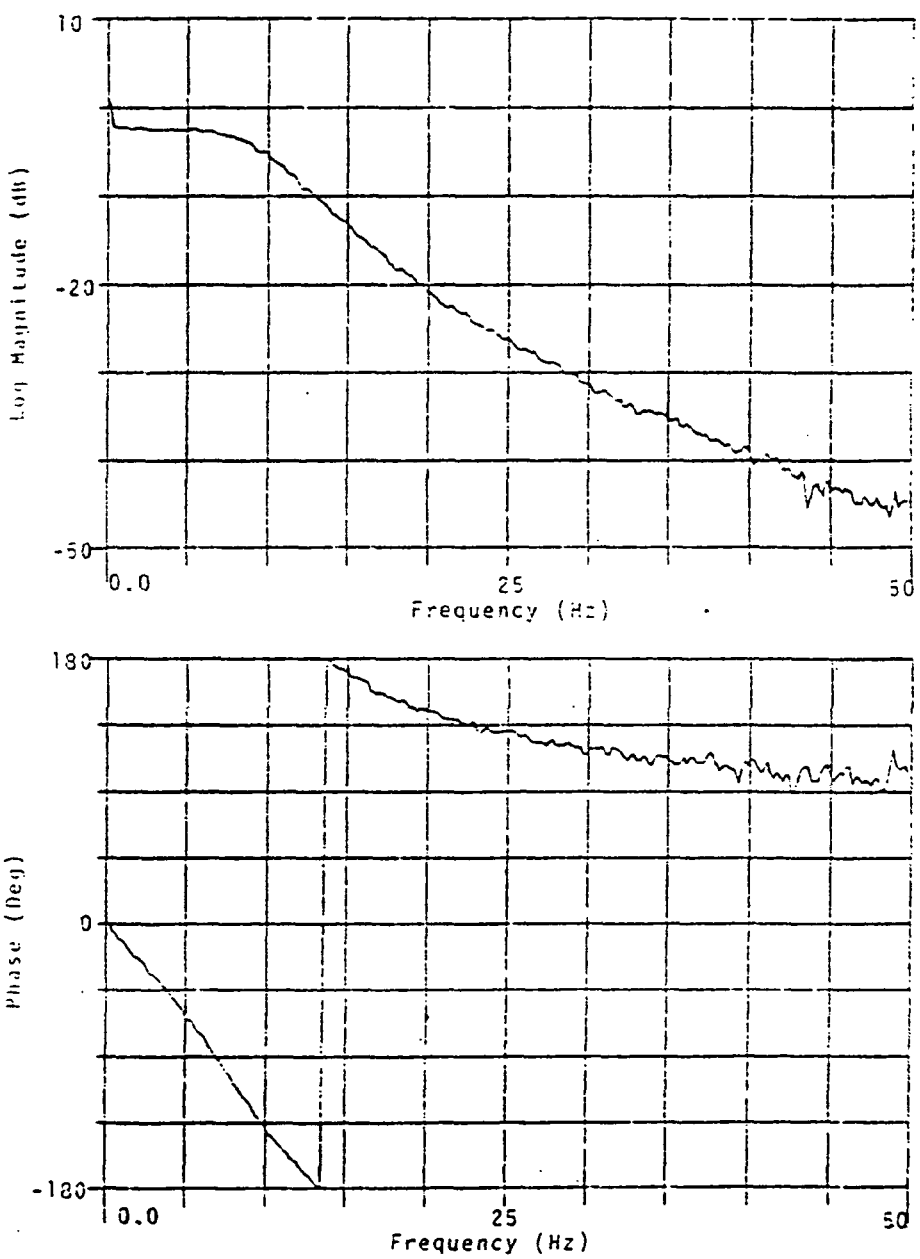


Figure A-14 Magnitude and Phase Response of the 10 Hz Digital Filter

APPENDIX B

2920 HARDWARE SUPPORT CIRCUIT

The nominal specifications of the digital pre-sampling filter are:

1) The filter transfer function is to approximate the transfer function of the analog PSF.
2) Balanced differential input
3) 100k ohms minimum input impedance
4) 20k ohms maximum output impedance
5) Nine bit signal resolution
6) Four filters per 2920 chip

The design specifications for the 2920 PSF filters will impact the filter implementation in two basic areas. Items 1, 5 and 6 of the specification list are primarily determined by the software (within the constraints of the 2920 chip hardware) the 2920 will execute. The structure, the filter algorithm, the accuracy of the multiplications and other software techniques determine these parameters. The I/O specifications of the filters are determined by the hardware. The 2920 will require hardware support to meet the input specifications and to perform post filtering of the output signal from the DAC's. Figure 2-1

shows a block diagram of the extra support circuits required for the PSF application.

The SCB circuits perform several functions. They act as buffers between sensor signals and the 2920 ADC's. The circuits convert the differential input signals to single line, and act as an anti-aliasing filter and input buffer. Finally, the circuit scales the input signal to levels acceptable for the 2920.

The SCB circuits shown in Figure B-1 perform these functions. The circuit was designed for differential signals magnitude limited to ± 5 volts. The corner frequency of the circuit is $1/R_2C$. The sample rate of the filter is known to be less than 4.166 kHz, therefore the corner frequency of the SCB circuit is 2.5 kHz. The gain of the SCB's is 0.5 volts/volt. The input impedance of the circuit 100k ohms.

The reconstruction (or post) filters remove the high frequency DAC noise prior to resampling by the PCM system. The corner frequency of these filters is 6.6k ohms.

The one volt reference voltage should be generated locally to the processor to reduce reference noise. Figure B-1 is the circuit diagram of a circuit which meets

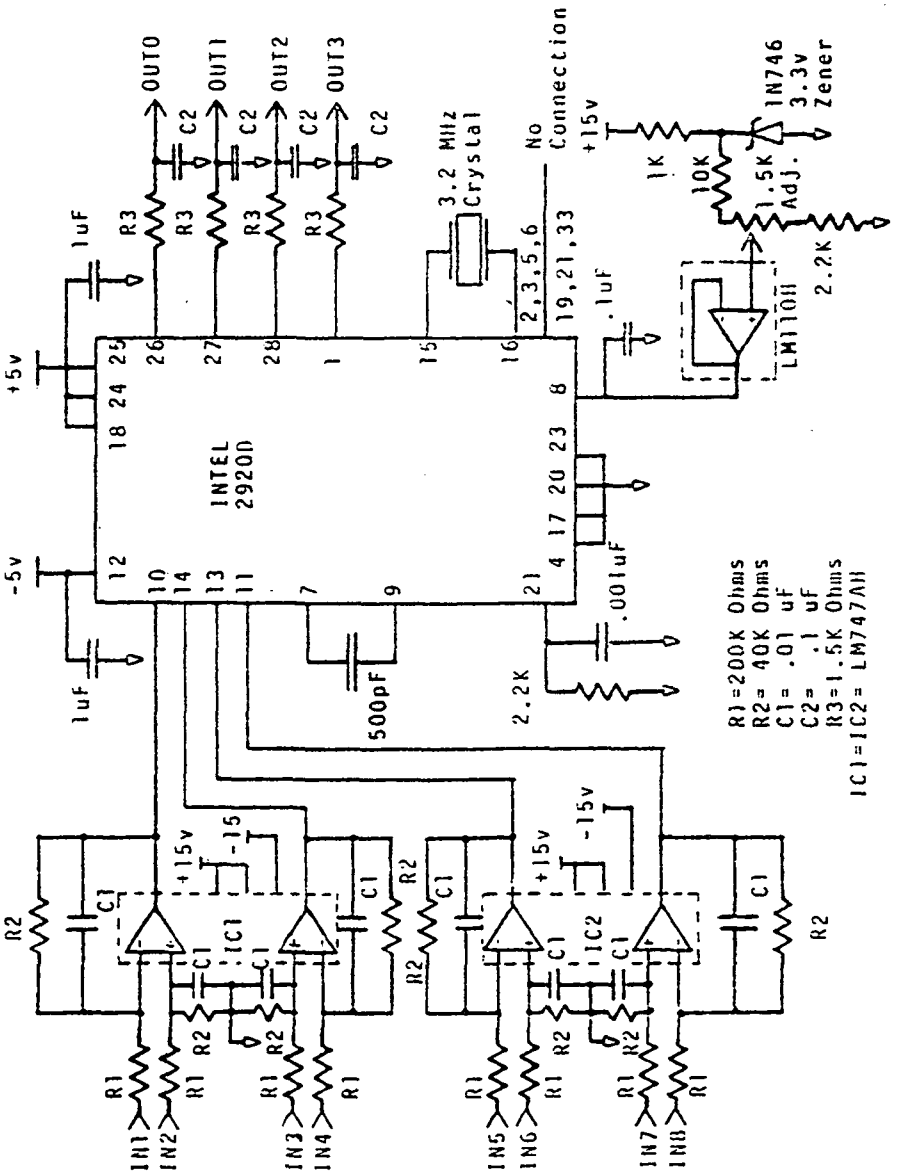


FIGURE B-1 PSF Hardware Support Circuitry for the 2920

the circuit specifications. This circuit was designed using available hardware. It is not intended to be a space optimal design. The circuit will serve as a model for a rough estimate of the space requirements of the 2920 PSF circuit. Table B-1 contains a breakdown of the various parts and estimated size. The space requirement for the circuit of Figure B-1 is estimated to be 10 square inches, or 2.58 square inches per filter.

Table B-1 Estimation of the 2920 PSF Circuit Size

Part	Description	Size (in^2)
2920-D	Signal Processor	.9
IC1	LM747AH, SCB op-amp	.09
IC2	LM747AH, SCB op-amp	.09
LM110H	Reference Voltage Buffer	.09
1N746	3.3 volt Zener Diode	.05
Crystal	3.2 MHz Crystal	.57
1.5k Adj	1.5 Adjustable potentiometer	.12
Misc	24 Misc Resistors @ .12/resistor	2.88
	17 Misc Capacitors @ .12/cap	2.04
	Subtotal	6.83
	circuit layout factor x .5	3.42
	Total estimated size (four filters)	10.25
	Estimated size per filter	2.58

APPENDIX C

2920 PROCESSOR OVERVIEW

The Intel 2920 digital signal processor is specifically designed for signal processing applications. The unique architecture of the 2920 gives it the capability and the speed necessary to perform real time digital signal processing functions.

2920 Architecture

As shown in Figure C-1, this unusual stand alone microprocessor may be divided into two major sections, analog and digital. The architecture of the 2920 allows analog and digital functions to be performed simultaneously.

The analog section, which is controlled by the digital section, contains all the components necessary for analog I/O. This section's circuitry is composed of four analog input channels, an input channel multiplexer, sample and hold circuitry, eight bit A/D and D/A converters and eight analog or digital output channels.

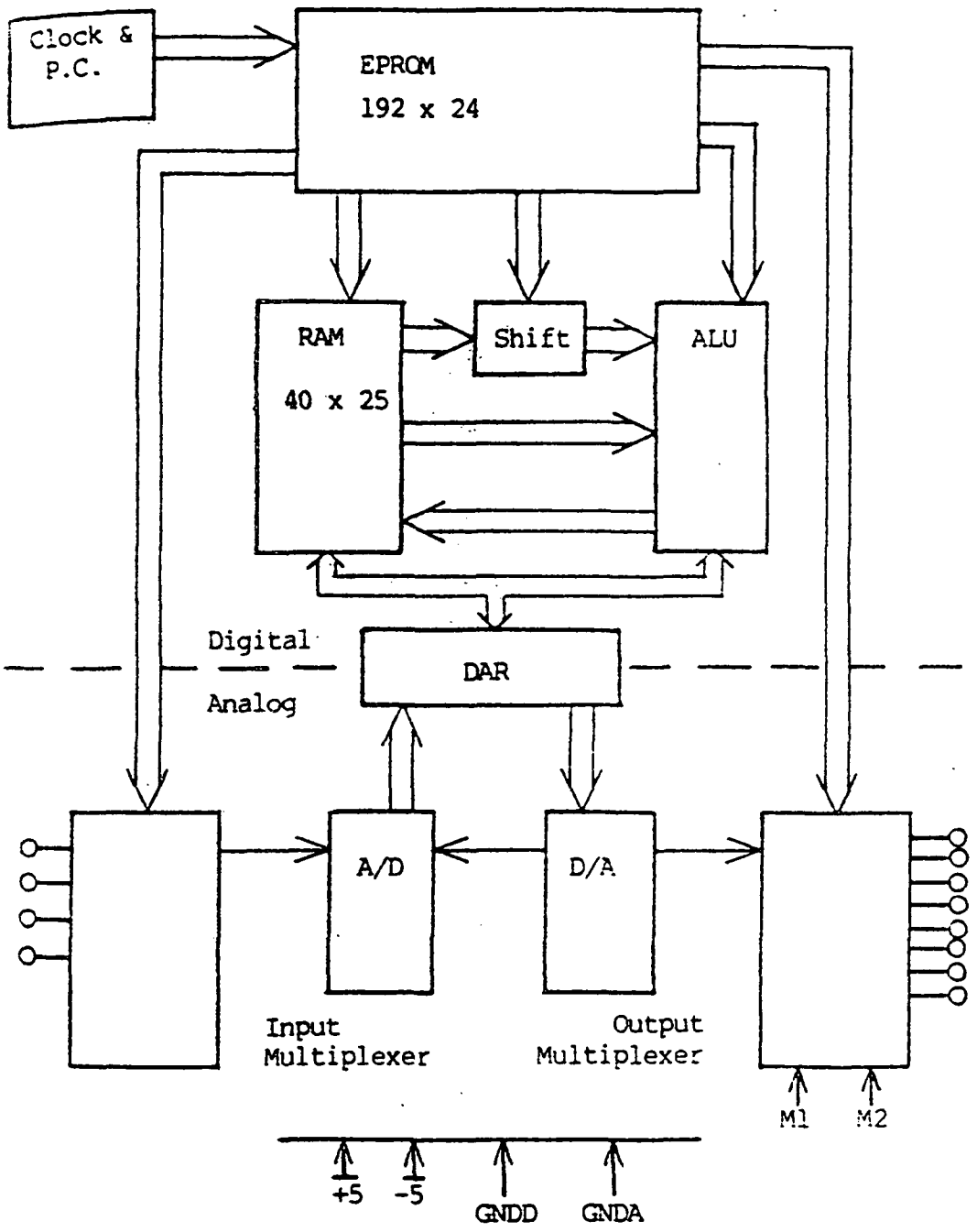


Figure C-1 2920 Signal Processor Block Diagram

Data is passed between the digital and analog sections by the nine bit DAR (Digital Analog Register).

The digital section of the processor contains 192 x 24 bits of EPROM program storage and 40 x 25 bits of RAM storage. The 2920 supports a maximum of 192 instructions, and up to 40 variables. Arithmetic operations are simplified by the capabilities of the shift register, which scales the A register operand by 2**N, where: -14 < N < 3. If an arithmetic operation causes an ALU overflow, the largest valid magnitude of the result and the proper sign bit is retained as the resultant.

The DAR is a nine bit 2's complement binary register which passes data between the digital and analog sections. After an A/D conversion, the result is converted from a 1's complement to a 2's complement number for the DAR. When the twenty five bit digital section reads data from the DAR, the lower sixteen bits of the DAR are read as 1's. This partially compensates for A/D offset errors.

2920 Instruction Set

The assembly language used by the 2920 is as unusual as the 2920 architecture. Each assembly language instruction is comprised of five instruction fields (see

Table C-1). The contents of these fields specify the ALU instruction, the B register, the A register, the shift factor and an analog instruction. Table C-1 lists the 2920 instruction set. The execution time of all the instructions is exactly equal.

Table C-1 The 2920 Instruction Set

(1) ALU instruction	(2) B address	(3) A address	(4) Shift code	(5) Analog instruction

Mnemonics	Operation
ALU Instructions	
ADD	(A x 2^n) + B ---> B
SUB	B - (A x 2^n) ---> B
LDA	(A x 2^n) -------> B
XOR	(A x 2^n) ⊕ B ---> B
AND	(A x 2^n) * B ---> B
LIM	Sign(A) ---------> B
Analog Instructions	
IN(k)	Sample input channel k
OUT(k)	D/A to output channel k
CVTS	Convert sign bit
CVT(k)	Perform A/D on bit k
CNDS	Select sign bit for conditional ALU instructions
CND(k)	Select bit k for conditional ALU instructions
EOP	Reset program counter
NOP	No operation
Shift Codes	
Rnn	Shift A right nn bits (scale factor = 2^-nn)
	nn = 0,1,2,3, ... 12,13
Lnn	Shift A left nn bits (scale factor = 2^nn)
	nn = 1,2

I/O Timing Requirements

The architecture of the 2920 places several restraints on analog I/O instruction sequences. A series of IN(k) instruction causes the input multiplexer to select channel k for input. The execution time of this series of IN(k) instructions must be at least six times the RC time constant of the sample and hold circuitry. This sampling time allows the hold capacitor to fully charge to the value of the input signal. The instruction sequence shown in Table C-2 executes an A/D conversion on input channel zero. The three IN(0) instructions (3.75E-6 seconds at 1.25E-6 seconds per instruction) provide the settling time required by the sample and hold capacitor.

The A/D converters require time to settle before each CVT(n) instruction. A series of analog NOP's whose execution time is no less than 1.2E-6 seconds should precede each CVT(n) instruction. The entire A/D conversion sequence (Fcry = 3.2 MHz) requires 31.25 E-6 seconds or 25 instruction cycles (at 1.25E-6 sec/instruction).

Table C-2 Nine Bit A/D Conversion Sequence

	Digital	Analog
1	SUB DAR,DAR,R00,	IN0
2	*	IN0
3	*	NOP
4	*	NOP
5	*	NOP
6	*	CVTS
7	ADD DAR,KM2,R00,	CND6
8	*	NOP
9	*	CVT7
10	*	NOP
11	*	CVT6
12	*	NOP
12	*	CVT5
13	*	NOP
14	*	CVT4
15	*	NOP
16	*	CVT3
17	LDA T,T,R00,	CND4
18	*	CVT2
19	LDA T,T,R00,	CND4
20	*	CVT1
21	LDA T,T,R00,	CND4
22	*	CVT0
23	LDA INP,DAR,R00,	*

* --> Indicates available instruction
--> F_{cry} = 3.2 MHz
--> C_{hold} = 500 pF

The sequence of instructions for D/A conversions has similar timing restrictions (see Table C-3). Once the DAR is loaded with output data, the D/A circuitry requires 6E-6 seconds settling time before an output instruction may be executed. Crosstalk between output channels is reduced by executing an OUT(k) instruction, where k is an unused channel number. Immediately following this output, the signal should be routed by a series of OUT(v)

Table C-3 Instruction Sequence for D/A Conversions

	Digital	Analog	
1	LDA DAR,OUTP,R00,	NOP	
2	ADD DAR,KP1,R01,	NOP	**
3	*	NOP	
4	*	NOP	
5	*	OUT4	
6	*	OUT0	
7	*	OUT0	

* --> Indicates available instruction
** --> DC offset correction
--> F_{cry} = 3.2 MHz
--> Output channel four not used

instructions to the appropriate output channel v. The execution time of the OUT v instruction series must be no less than 1E-6 seconds.

Multiplication in the 2920

Time delays, additions and multiplications are the three basic functions which form the basis for implementing digital filters. Time delay and addition functions are supported directly by the 2920 assembler.

Multiplication is expressed as a series of LDA, ADD or SUB instructions. In a typical filtering application, the algorithms which multiply variables by constants comprise 75% of the filter program code. The efficiency of these multiplications determines the space efficiency of the filter code as a whole.

There are several ways to code a multiplication between a variable and constant. For example, consider the multiplication:

REG0 = REG1 * 1.8727

where: REG0 = variable A in a filtering program,

REG1 = variable B in a filtering program.

The first task in the process of coding a multiplication algorithm is to express the constant as a 25 bit binary number. The 2920 has a 25 bit (24 bit plus sign) fixed point accumulator; therefore, the binary representation of the constant need not be greater than 24 bits.

Table C-4 illustrates that there are many ways to code a multiplication, some ways more efficient than others. Sequence A requires twenty instructions to perform the twenty five bit multiplication. Sequence uses several coding tricks to shorten the sequence to nine instructions. If twenty five bit accuracy is not required, the multiplication algorithm can be shortened considerably by rounding the constant to an appropriate value. See the 2920 Assembly Language Manual for further details on 2920 programming techniques.

Table C-4 Examples of Algorithms Which Multiply REG0 by 1.8727

A) Instruction	Binary Result
LDA REG1, REG0, R01	0.100 000 000 000 000 000 000 000
ADD REG1, REG0, R02	0.110 000 000 000 000 000 000 000
ADD REG1, REG0, R04	0.110 100 000 000 000 000 000 000
ADD REG1, REG0, R05	0.110 110 000 000 000 000 000 000
ADD REG1, REG0, R06	0.110 111 000 000 000 000 000 000
ADD REG1, REG0, R08	0.110 111 010 000 000 000 000 000
ADD REG1, REG0, R09	0.110 111 011 000 000 000 000 000
ADD REG1, REG0, R11	0.110 111 011 010 000 000 000 000
ADD REG1, REG0, R12	0.110 111 011 011 000 000 000 000
LDA REG0, REG1, R12	0.000 000 000 000 110 111 011 011
ADD REG1, REG0, R00	1.000 000 000 000 110 111 011 011
ADD REG1, REG0, R01	1.100 000 000 000 110 111 011 011
ADD REG1, REG0, R02	1.110 000 000 000 110 111 011 011
ADD REG1, REG0, R04	1.110 100 000 000 110 111 011 011
ADD REG1, REG0, R05	1.110 110 000 000 110 111 011 011
ADD REG1, REG0, R06	1.110 111 000 000 110 111 011 011
ADD REG1, REG0, R07	1.110 111 100 000 110 111 011 011
ADD REG1, REG0, R08	1.110 111 110 000 110 111 011 011
ADD REG1, REG0, R10	1.110 111 110 100 110 111 011 011
ADD REG1, REG0, R11	1.110 111 110 110 110 111 011 011

B)	
LDA REG1, REG0, R09	0.000 000 001 000 000 000 000 000
ADD REG1, REG0, R12	0.000 000 001 001 000 000 000 000
SUB REG1, REG0, R02	0.000 000 000 110 110 000 000 000
ADD REG1, REG0, R05	0.000 010 000 110 110 000 000 000
SUB REG1, REG0, R08	0.000 001 110 110 110 000 000 000
ADD REG1, REG0, R10	0.000 001 110 110 110 111 011 011
ADD REG1, REG0, L01	10.000 001 110 110 110 111 011 011 *
SUB REG1, REG0, R03	1.111 001 110 110 110 111 011 011
SUB REG1, REG0, R05	1.110 111 110 110 110 111 011 011

Note: 1.8727_{10} = 1.110 111 110 110 110 111 011 011_2 + err

* May cause accumulator overflow

Sample Rate Division

Table C-5 shows a sequence of instructions which may be used to modify the sample rate of the filter. The variable OSC is a counter which is decremented by the constant KPn (n ranges from 0 to 9) each time the code is executed. The filter is kept inactive by disabling the

Table C-5 Sample Rate Division Instruction Sequence for Sample Rate Divisor of Three

Instruction	Comments
LDA DAR,OSC,R00,NOP	DAR = Current Value of OSC
* ,CNDS	The time delay and input
* ,CNDS	instructions execute only
* ,CNDS	if DAR (OSC) is negative
LDA OSC,KP2,R05,NOP	Reset OSC
SUB OSC,KP1,R05,NOP	Decrement OSC

time delay and I/O functions while the counter variable is greater than zero. When OSC decrements to a negative value then the filter input and output values are updated and the time delay function is executed. OSC is reloaded with a positive constant KPm and the sequence repeats. This sequence effectively divides the original sample rate of the filter by the ratio of the positive constant to the decremental value. The corner frequency of the filter is divided by the same amount. The new corner frequency of a filter using the sample rate division may be calculated

using the relationship:

$$F_C' = \left[\frac{F_C}{\text{INT}\left[\frac{KPm+1}{KPn}\right]}\right]$$

where F_C' = new sample rate

F_C = basic sample rate

KPm = positive initial counter value

KPn = decremental value

APPENDIX D

PSFSS

Pre-Sampling Filter Software Support

The PSFSS software package reduces the process of writing a program which implements four pre-sampling filters with the 2920 to a single instruction command. The PSFSS command:

```
DO FILTER(F1,F2,F3,F4,[$LIST])
```

links the four filter modules F1,F2,F3,F4 and assembles the code. It also produces an optional program listing (if specified by the $LIST option) and loads the finished quad filter code into the EPROM of the 2920. The chip is then ready to be plugged into the hardware for operation.

The PSFSS package maps the 2920 EPROM program memory into four blocks. Each memory block is forty eight program steps long. Filter modules are assigned a memory block during the linking process. The filter modules are written to handle all the necessary filter functions within these forty eight program steps, with the exception of output.

Phase delay due to digital calculations is minimized by having each filter module output its results immediately after the calculations are complete. As module A finishes its calculations, module B (the next in its program space) starts its calculations. While the digital section is executing calculations for module B, the analog section is executing the output sequence for module A. Thus, the output of module F1 is handled by F2, the output of F2 is done by F3 and so on.

Digital filtering programs must meet several requirements before they can be included in the PSFSS library. During the module linking process, PSFSS searches each of the modules specified in the DO FILTER command for several special characters. These special characters aid the linking process by specifying the location of various module and position dependent instructions. Listed below are these special characters and their corresponding PSFSS operation:

'_' ------> PSFSS replaces the underline character with the block number assigned to the filter module used in the code to create unique variable names.

'#' ------> The # character is replaced with the block number of the previous filter module. Used

to identify the variable name of the previous modules output.

'OUT+' ----> OUT+ is replaced by OUTk, where k is the output channel assigned to the previous filter module.

'IN+' -----> IN+ is replaced by INv, where v is the input channel assigned to the current filter module.

'@' -------> If the module this appears in is in block 4, it is replaced with an EOP instruction to indicate end of program, otherwise it is removed from the code.

PSFSS Commands

The valid PSFSS commands are:

COPY FILTER.LST TO :nn:

COPY FILTER.LST copies the assembled list file to the line printer (:nn:=:LP:), or console (:nn:=:CO:). FILTER.LST contains the filter printout and the assembled code listing if specified by the $LIST option.

COPY HELP.??? TO :nn:

COPY HELP.??? copies of all the help files to the line printer (:nn:=:LP:), or to the console (:nn:=:co:).

DO FILTER(f1,f2,f3,f4[,$LIST])

The DO FILTER command assembles four third order Butterworth filters which have corner frequencies specified by f1,f2,f3 and f4. If the $LIST option is specified, the assembled code listing is included in the list file. F1,f2,f3 and f4 are selected from the set of filter modules listed by the HELP FILters command.

DO SIM(FILTER)

DO SIM(FILTER) sets up the 2920 simulator program for simulation of assembled filter code. Consult the 2920 Simulator User's Guide for further information.

HELP COM

HELP COMmand lists a short description of the common PSFSS commands.

HELP COP

HELP COPy lists commands used to copy files to the console or line printer.

HELP EXA

HELP EXAmple lists a short PSFSS programming example.

HELP FRE

HELP FREquencies displays a listing of the available filter module corner frequencies and their respective sampling rates.

HELP HEL

HELP HELp lists the available help commands.

HELP PRO

HELP PROcedure outlines the proper sequence for using the DO FILTER command.

PSFSS Software

The PSFSS software is written to operate in an Intel ISIS-II operating system environment. The DO FILTER command which initiates the creation of a PSF filter group, prompts the ISIS-II system program DO (renamed from SUBMIT), to process the commands in the command file FILTER.CSD (listing D-1). The FILTER.CSD file directs the ISIS-II editor to concatenate five files into a temporary file for processing. The five files include the HEADER.TXT file (Listing D-2), and any four filter source files (Listings D-3 through D-16). The temporary file is edited to link the filter modules, then is assembled by the AS2920 assembler, again under the control of the FILTER.CSD command file. When the assembly is complete, the program BEEP (Listing D-17) is called to notify the

user that assembly is complete. When the user responds, the command file directs the loading of the object code created by the assembler in the 2920.

The HELP information is processed by the program HELP (listing D-18). HELP dumps the contents of the HELP.XXX (where XXX is specified by the HELP XXX command) file to the CRT. The contents of the HELP.XXX files are shown in Listings D-19 through D25.

```
CREDIT TIME
OR HEADER.TXT;B;R24;CR;JTT;S/ELIST/%4/
JTT;2<S/T1/%0/;S/T2/%1/;S/T3/%2/;S/T4/%3/>;JTE;L-1;TS1;JTE
OR F%0.SRC;B;RB0;CR
JTE;L-1;TS2;JT1;S/_/F0 ===>//;!<JT1;S/_/F0/>;!<JT1;S/#/3/>
!<JT1;S/+/0/>;JT1;S/@//;JTE;OR F%1.SRC;B;RB0;CR;JTE;L-1;TS3;JT2
S/_/F1 ===>//;!<JT2;S/_/F1/>;!<JT2;S/#/0/>
!<JT2;S/+/1/>;JT2;S/@//;JTE;OR F%2.SRC;B;RB0;CR;JTE;L-1;TS4;JT3
S/_/F2 ===>//;!<JT3;S/_/F2/>;!<JT3;S/#/1/>
!<JT3;S/+/2/>;JT3;S/@//;JTE;OR F%3.SRC;B;RB0;CR;JT4
S/_/F3 ===>//;!<JT4;S/_/F3/>;!<JT4;S/#/2/>
!<JT4;S/+/3/>;JT3;S/@/,EOF/;JTE;I/END
/
EX FILTER.SRC

CLR
;SOURCE FILE COMPLETE
AS2920 FILTER.SRC  NOSYMBOLS

;PROGRAM FILTER
;READY TO PROGRAM EPROM
BEEP1
COPY FILTER.LST TO :LP:
UPM
1702A
SOCKET=2
TRANSFER FROM 0 TO 7FFH
SOCKET=1
COMPARE FROM 0 TO 47FH
READ FILE FILTER.HEX INTO 0
PROGRAM FROM 0 TO 47FH START 0
COMPARE FROM 0 TO 460H
EX1
```

Listing D-1 FILTER.CSD Command File Listing

```
; FILTER

; FILTER IS A COMBINATION OF FOUR PRE-SAMPLING FILTER
; ALGORITHMS WRITTEN TO BE IMPLEMENTED ON A 2920.
; THE CRYSTAL FREQUENCY IS ASSUMED TO BE 3.2 MHZ.

; INPUT    #0   (PIN 10) = F1 HZ FILTER INPUT
;          #1   (PIN 14) = F2 HZ
;          #2   (PIN 13) = F3 HZ
;          #3   (PIN 11) = F4 HZ

; OUTPUT   #0   (PIN 26) = F1 HZ FILTER OUTPUT
;          #1   (PIN 27) = F2 HZ
;          #2   (PIN 28) = F3 HZ
;          #3   (PIN  1) = F4 HZ

 $NOLIST
 #LIST
;*****************************************************************
 $TITLE ('FILTER ==> FOUR CASCADE FORM DIGITAL FILTERS')
```

Listing D-2 HEADER.TXT File Listing

```
;*************************************************************
; FILTER _                          Fc=300Hz    SF 300

 LDA DAR,OUTP$,R00
 ADD DAR,RP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.47617977e= 01. 011 110 011 110 011 011 101 011

 LDA SUM1,_P1,R00
 ADD SUM1,_P1,R01
 SUB SUM1,_P1,R05
 ADD SUM1,_P1,R07,OUT4
 SUB SUM1,_P1,R11,OUT$
 OUT$

 SUB DAR,DAR,R00,IN+

;B2=-.641284365= -00. 101 001 000 010 101 100 110 110

 LDA TEMP,_T2,R01,IN+
 ADD TEMP,_T2,R03
 ADD TEMP,_T2,R06
 ADD TEMP,_T2,R11
 ADD TEMP,_T2,R13
 SUB SUM1,TEMP,R00,CVT5

 ADD DAR,RM2,R00,CND6  ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R02
 ADD SUM1,_Z2,R03,CVT7

;B1=.623840501= 00.101 000 000 011 011 100 010 101

 LDA SUM2,_ZP,R01
 ADD SUM2,_ZP,R03,CVT6
 ADD SUM2,_ZP,R10
 SUB SUM2,_ZP,R13,CVT5
 NOP
 CVT+
 NOP

;1/A0=.790725756= 00.110 010 100 110 110 100 000 001

 LDA TEMP,_ZP,R01,CVT3
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R02,CVT2
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R05,CVT1
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R07,CVT0

;READY FOR INPUT

 LDA INP,DAR,R02
 ADD SUM1,INP,R03
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R02
 ADD TEMP,SUM2,R05
 ADD TEMP,SUM2,R07

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP0,R05,CND5
 LDA _P1,_P1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _T2,_P1,R00,CND5
 SUB _OSC,KP1,R050
 LDA _P1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP+,TEMP,L01,CND5
```

D-3 F300.SRC Listing

```
;*************************************************************
; FILTER _                          Fc=150Hz    BF 300

 LDA DAR,OUTP*,R00
 ADD DAR,RP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.476179776= 01. 011 110 011 110 011 011 101 011

 LDA SUM1,_P1,R00
 ADD SUM1,_P1,R01
 SUB SUM1,_P1,R05
 ADD SUM1,_P1,R07,OUT4
 SUB SUM1,_P1,R11,OUT5
 OUT6

 SUB DAR,DAR,R00,IN+

;B2=-.641284365= -00. 101 001 000 010 101 100 110 110

 LDA TEMP,_I2,R01,IN+
 ADD TEMP,_I2,R03
 ADD TEMP,_I2,R06
 ADD TEMP,_I2,R11
 ADD TEMP,_I2,R13
 SUB SUM1,TEMP,R00,CVT5

 ADD DAR,AM2,R00,CN06  ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_I1,R02
 ADD SUM1,_I2,R03,CVT7

;A1=.625840501= 00.101 000 000 011 011 100 010 101

 LDA SUM2,_ZP,R01
 ADD SUM2,_ZP,R03,CVT6
 ADD SUM2,_ZP,R10
 SUB SUM2,_ZP,R13,CVT5
 NOP
 OUT4
 NOP

;A1/A0=.790725756= 00.110 010 100 110 110 100 000 001

 LDA TEMP,_ZP,R01,CVT3
 LDA I/I,R00,CN04
 ADD TEMP,_ZP,R02,CVT2
 LDA I/I,R00,CN04
 ADD TEMP,_ZP,R05,CVT1
 LDA I/I,R00,CN04
 ADD TEMP,_ZP,R07,CVT0

;READY FOR INPUT

 LDA INP,DAR,R03
 ADD SUM1,INP,R03
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R04
 ADD TEMP,SUM2,R05
 ADD TEMP,SUM2,R07

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,RP1,R05,CN06
 LDA _ZP,_Z1,R00,CN05
 LDA _Z1,INP,R00),CN05
 LDA I/I,_C1,R00),CN05
 SUB _OSC,RP1,R05@
 LDA _P1,SUM1,R00),CN05
 LDA _ZP,SUM2,R00,CN05
 LDA OUTP*,TEMP,L01,CN05
```

D-4 F150.SRC Listing

```
;-----------------------------------------------------------------
; FILTER _                          Fc=100Hz     BF 300

 LDA DAR,OUTF$,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.476177776= 01. 011 110 011 110 011 011 101 011

 LDA SUM1,_P1,R00
 ADD SUM1,_P1,R01
 SUB SUM1,_P1,R05
 ADD SUM1,_P1,R07,OUT4
 SUB SUM1,_P1,R11,OUT8
 OUT$

 SUB DAR,DAR,R00,IN+

;B2=-.641234365= -00. 101 001 000 010 101 100 110 110

 LDA TEMP,_I2,R01,IN+
 ADD TEMP,_I2,R03
 ADD TEMP,_I2,R06
 ADD TEMP,_I2,R11
 ADD TEMP,_I2,R13
 SUB SUM1,TEMP,R00,CVT5

 AID DAR,KM2,R00,END6   ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R02
 ADD SUM1,_Z2,R03,CVT7

;B1=.625840501= 00.101 000 000 011 011 100 010 101

 LDA SUM2,_ZP,R01
 ADD SUM2,_ZP,R03,CVT6
 ADD SUM2,_ZP,R10
 SUB SUM2,_ZP,R13,CVT5
 NOP
 CVT4
 NOP

;1/A0=.799725756= 00.110 010 100 110 110 100 000 001

 LDA TEMP,_ZP,R01,CVT3
 LDS I/I,R00,CND4
 ADD TEMP,_ZP,R02,CVT2
 LDA I/I,R00,CND4
 ADD TEMP,_ZP,R05,CVT1
 LDA I/I,R00,CND4
 ADD TEMP,_ZP,R07,CVT0

;READY FOR INPUT

 LDA INP,DAR,R03
 ADD SUM1,INP,R03
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R05
 ADD TEMP,SUM2,R02
 ADD TEMP,SUM2,R07

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP2,R05,CND5
 LDA _Z2,_Z1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _Z2,_P1,R00,CND5
 SUB _OSC,KP1,R05@
 LDA _P1,SUM1,R00,CNU5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP+,TEMP,L01,CND5
```

D-5 F100.SRC Listing

```
;*************************************************************
; FILTER _                          Fc=80Hz       BF 80 Hz

 LDA DAR,OUTP$,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.872774990534= 01. 110 111 110 110 111 000 101 110

 LDA SUM1,_P1,R12
 SUB SUM1,SUM1,R03
 ADD SUM1,_P1,R01
 SUB SUM1,_P1,R03
 SUB SUM1,_P1,R09,OUT4
 SUB SUM1,_P1,R11,OUT$

;B2=-.8864856685= -00. 111 000 101 111 000 010 111 001

 LDA TEMP,_I2,R07,OUT$

 SUB DAR,DAR,R00,IN+

 ADD TEMP,_I2,R06,IN+
 ADD TEMP,TEMP,R10
 SUB TEMP,_I2,R12
 ADD TEMP,_I2,R00
 SUB TEMP,_I2,R03
 SUB SUM1,TEMP,R00,CVT5

 ADD DAR,KM2,R00,CND6  ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R07
 ADD SUM1,_Z2,R08,CVT7

;C1=.886095240726= 00.111 000 101 101 011 100 100 011

 LDA SUM2,_ZP,R00
 SUB SUM2,_ZP,R03,CVT6
 ADD SUM2,SUM2,R13
 ADD SUM2,_ZP,R06,CVT5
 SUB SUM2,_ZP,R08
 SUB SUM2,_ZP,R11,CVT4
 SUB SUM2,_ZP,R12

;K0=.639e770352= 00.101 000 111 100 000 111 011 111

 LDA TEMP,_ZP,R01,CVT3
 LDA 1,1,R00,CND4
 ADD TEMP,_ZP,R03,CVT2
 LDA 1,1,R00,CND4
 ADD TEMP,_ZP,R06,CVT1
 LDA 1,1,R00,CND4
 CVT0

;READY FOR INPUT

 LDA INP,DAR,R02
 ADD SUM1,INP,R08
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R03
 ADD TEMP,SUM2,R06

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP0,R05,CND5
 LDA _Z2,_Z1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _I2,_P1,R09,CND5
 SUB _OSC,KP1,R05C
 LDA _Z1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP+,TEMP,R00,CND5
```

D-6 F80.SRC Listing

```
;*****************************************************************
; FILTER _                          Fc=75Hz     BF 300

 LDA DAR,OUTP#,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.47e179776= 01. 011 110 011 110 011 011 101 011

 LDA SUM1,_P1,R00
 ADD SUM1,_P1,R01
 ADD SUM1,_P1,R05
 ADD SUM1,_P1,R07,OUT+
 SUB SUM1,_P1,R11,OUT#
 OUT+

 SUB DAR,DAR,R00,(N+

;B2=-.e412B4365= -00. 101 001 000 010 101 100 110 110

 LDA TEMP,_T2,R01,IN+
 ADD TEMP,_T2,R03
 ADD TEMP,_T2,R06
 ADD TEMP,_T2,R11
 ADD TEMP,_T2,R13
 SUB SUM1,TEMP,R00,OUT5

 OD DAR,AM2,R00,CND6  ;ANALOG

;A1 AND A2 COEFFICIENTS

 LDA SUM1,_Z1,R02
 ADD SUM1,_Z2,R03,OUT7

;C1=.625840501= 00.101 000 000 011 011 100 010 101

 LDA SUM2,_ZP,R01
 ADD SUM2,_ZP,R03,OUT6
 ADD SUM2,_ZP,R10
 SUB SUM2,_ZP,R13,OUT5
 NOP
 OUT*
 NOP

;A1/A0=.79072576= 00.110 010 100 110 110 100 000 001

 LDA TEMP,_ZP,R01,OUT3
 LDA 1,1,R00,CND4
 ADD TEMP,_ZP,R02,OUT2
 LDA 1,1,R00,CND4
 ADD TEMP,_ZP,R05,OUT1
 LDA 1,1,R00,CND4
 ADD TEMP,_ZP,R07,OUT0

;READY FOR INPUT

 LDA INP,DAR,R03
 ADD SUM1,INP,R03
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R02
 ADD TEMP,SUM2,R05
 ADD TEMP,SUM2,R07

;TIME DELAY

 LDA DAR,_DSC,R00

 LDA _DSC,KP3,R05,CND5
 LDA _Z2,_Z1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _T2,_P1,R00,CND5
 SUB _DSC,KP1,R05#
 LDA _P1,SUM1,R00,CND5
 LDA _Z2,SUM2,R00,CND5
 LDA OUTP+,TEMP,L01,CND5
```

D-7 F75.SRC Listing

```
;*****************************************************************
; FILTER _                          Fc=70Hz      BF 70

 LDA DAR,OUTP+,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.869336203= 01. 111 000 111 010 101 110 001 010

 LDA SUM1,_P1,R06
 SUB SUM1,_P1,R09
 ADD SUM1,_P1,R12
 ADD SUM1,SUM1,R05
 ADD SUM1,_P1,R01,OUT4
 SUB SUM1,_P1,R03,OUT#

;B2=-.899911152= -00. 111 001 100 110 000 010 010 100

 LDA TEMP,_T2,R05,OUT4

 SUB DAR,DAR,R00,IN+

 SUB TEMP,_T2,R07,IN+
 ADD TEMP,TEMP,R04
 ADD TEMP,_T2,R00
 SUB TEMP,_T2,R03
 NOP
 SUB SUM1,TEMP,R00,CVT5

 ADD DAR,KM2,R00,CND6  ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R07
 ADD SUM1,_Z2,R08,CVT7

;C1=.399648807= 00.111 001 100 100 111 100 110 001

 LDA SUM2,_ZP,R00
 SUB SUM2,_ZP,R04,CVT6
 ADD SUM2,SUM2,R12
 SUB SUM2,_ZP,R04,CVT5
 ADD SUM2,_ZP,R08
 ADD SUM2,_ZP,R07,CVT4
 ADD SUM2,_ZP,R10

;1/K0=.269369648= 00.110 111 101 000 111 100 000 010

 LDA TEMP,_ZP,R00,CVT3
 LDA T/T,R00,CND4
 SUB TEMP,_ZP,R03,CVT2
 LDA T/T,R00,CND4
 SUB TEMP,_ZP,R06,CVT1
 LDA T/T,R00,CND4
 CVT0

;READY FOR INPUT

 LDA INP,DAR,R03
 ADD SUM1,INP,R06
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R02
 SUB TEMP,SUM2,R05
 ADD TEMP,SUM2,R08

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP0,R05,CND5
 LDA _Z2,_Z1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _T2,_P1,R00,CND5
 SUB _OSC,KP1,R05@
 LDA _P1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP+,TEMP,R00,CND5
```

D-8 F70.SRC Listing

ORIGINAL PAGE IS
OF POOR QUALITY

```
;*****************************************************************
; FILTER _                              Fc=60Hz     BF 300

 LDA DAR,OUTF8,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.47617976= 01. 011 110 011 110 011 011 101 011

 LDA SUM1,_P1,R00
 ADD SUM1,_P1,R01
 SUB SUM1,_P1,R05
 ADD SUM1,_P1,R07,OUT+
 SUB SUM1,_P1,R11,OUT+
 OUT+

 SUB DAR,DAR,R00,IN+

;B2=-.61284365= -00. 101 001 000 010 101 100 110 110

 LDA TEMP,_T2,R01,IN+
 ADD TEMP,_T2,R03
 ADD TEMP,_T2,R06
 ADD TEMP,_T2,R11
 ADD TEMP,_T2,R13
 SUB SUM1,TEMP,R00,CVT5

 ADD DAR,KM2,R00,CND6   ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R02
 ADD SUM1,_Z2,R03,CVT7

;A1=.82584501= 00.101 000 000 011 011 100 010 101

 LDA SUM2,_ZP,R01
 ADD SUM2,_Z2,R03,CVT6
 ADD SUM2,_ZP,R10
 SUB SUM2,_ZP,R13,CVT5
 NOP
 CVT+
 NOP

;1/K0=.79072576= 00.110 010 100 110 110 100 000 001

 LDA TEMP,_ZP,R01,CVT3
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R02,CVT2
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R05,CVT1
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R07,CVT0

;READY FOR INPUT

 LDA INP,DAR,R03
 ADD SUM1,INP,R03
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R02
 ADD TEMP,SUM2,R05
 ADD TEMP,SUM2,R07

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP4,R05,CND5
 LDA -42,_Z1,R00,CND5
 LDA -41,INP,R00,CND5
 LDA -12,_P1,R00,CND5
 SUB _OSC,KP1,R05@
 LDA _P1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP+,TEMP,L01,CND5
```

D-9 F60.SRC Listing

```
;*************************************************************
; FILTER _                          Fc=50Hz       SF 300

 LDA DAR,OUTP$,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.47617977$= 01. 011 110 011 110 011 011 101 011

 LDA SUM1,_P1,R00
 ADD SUM1,_P1,R01
 SUB SUM1,_P1,R05
 ADD SUM1,_P1,R07,OUT=
 SUB SUM1,_P1,R11,OUT$
 OUT$

 SUB DAR,DAR,R00,IN+

;B2=-.64128436$= -00.' 101 001 000 010 101 100 110 110

 LDA TEMP,_I2,R01,IN+
 ADD TEMP,_I2,R02
 ADD TEMP,_I2,R06
 ADD TEMP,_I2,R11
 ADD TEMP,_I2,R13
 SUB SUM1,TEMP,R00,CVT5

 ADD DAR,KM2,R00,CN0o  ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R02
 ADD SUM1,_Z2,R03,CVT7

;C1=.625840501= 00.101 000 000 011 011 100 010 101

 LDA SUM2,_ZP,R01
 ADD SUM2,_ZP,R03,CVT6
 ADD SUM2,_ZP,R10
 SUB SUM2,_ZP,R13,CVT5
 NOP
 CVT+
 NOP

;1/A0=.790725756= 00.110 010 100 110 110 100 000 001

 LDA TEMP,_ZP,R01,CVT3
 LDA 1,T,R00,CN04
 ADD TEMP,_ZP,R02,CVT2
 LDA 1,T,R00,CN04
 ADD TEMP,_ZP,R05,CVT1
 LDA 1,T,R00,CN04
 ADD TEMP,_ZP,R07,CVT0

;READY FOR INPUT

 LDA INP,DAR,R03
 ADD SUM1,INP,R03
 ADD SUM2,SUM1,R03
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R03
 ADD TEMP,SUM2,R05
 ADD TEMP,SUM2,R07

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP5,R05,CN0S
 LDA _Z2,_Z1,R00,CN0S
 LDA _Z1,INP,R00,CN0S
 LDA _I2,_P1,R00,CN0S
 SUB _OSC,KP1,R05E
 LDA _Z1,SUM1,R00,CN0S
 LDA _ZP,SUM2,R00,CN0S
 LDA OUTP+,TEMP,L01,CN0S
```

D-10 F50.SRC Listing

```
;*****************************************************************
; FILTER _                          Fc=40Hz      BP 60 Hz

 LDA DAR,OUTP$,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.87277490534= 01. 110 111 110 110 111 000 101 110

 LDA SUM1,_P1,R12
 SUB SUM1,SUM1,R03
 ADD SUM1,_P1,C01
 SUB SUM1,_P1,R05
 SUB SUM1,_P1,R07,OUT4
 SUB SUM1,_P1,R11,OUT$

;B2=-.88685a665+= -00. 111 000 101 111 000 010 111 001

 LDA TEMP,_T2,R07,OUT$

 SUB DAR,DAR,R00,IN+

 ADD TEMP,_T2,R08,IN+
 ADD TEMP,TEMP,R10
 SUB TEMP,_T2,R12
 ADD TEMP,_T2,R00
 SUB TEMP,_T2,R03
 SUB SUM1,TEMP,R00,CVT3

 ADD DAR,KM2,R00,CND6   ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R07
 ADD SUM1,_Z2,R08,CVT7

;C1=.88609524072e= 00.111 000 101 101 011 100 100 011

 LDA SUM2,_ZP,R00
 SUB SUM2,_ZP,R03,CVT6
 ADD SUM2,SUM2,R13
 ADD SUM2,_ZP,R06,CVT5
 SUB SUM2,_ZP,R08
 SUB SUM2,_ZP,R11,CVT4
 SUB SUM2,_ZP,R12

;1/K0=.6396770352= 00.101 000 111 100 000 111 011 111

 LDA TEMP,_ZP,R01,CVT3
 LDA I,T,R00,CND4
 ADD TEMP,_ZP,R03,CVT2
 LDA I,T,R00,CND4
 ADD TEMP,_ZP,R06,CVT1
 LDA I,T,R00,CND4
 CVT0

;READY FOR INPUT

 LDA INP,DAR,R02
 ADD SUM1,INP,R08
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R03
 ADD TEMP,SUM2,R06

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP1,R05,CND5
 LDA _Z1,_Z1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _T2,_P1,R00,CND5
 SUB _OSC,KP1,R05$
 LDA _P1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP4,TEMP,R00,CND5
```

D-11 F40.SRC Listing

```
;*****************************************************************
; FILTER -                    Fc=35Hz         BF 75

 LDA DAR,OUTP0,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.889336203= 01. 111 000 111 010 101 110 001 010

 LDA SUM1,_P1,R06
 SUB SUM1,_P1,R07
 ADD SUM1,_P1,R12
 ADD SUM1,SUM1,R05
 ADD SUM1,_P1,C01,OUT4
 SUB SUM1,_P1,R03,OUT4

;B2=-.899911152= -00. 111 001 100 110 000 010 010 100

 LDA TEMP,_I2,R05,OUT4

 SUB DAR,DAR,R00,IN+

 SUB TEMP,_I2,R07,IN+
 ADD TEMP,TEMP,R04
 ADD TEMP,_I2,R00
 SUB TEMP,_I2,R03
 NOP
 SUB SUM1,TEMP,R00,OUT3

 ADD DAR,KM2,R00,CND6  ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R07
 ADD SUM1,_Z2,R06,OUT7

;C1=.899645867= 00.111 001 100 100 111 100 110 001

 LDA SUM2,_ZP,R00
 SUB SUM2,_ZP,R04,OUT5
 ADD SUM2,SUM2,R12
 SUB SUM2,_ZP,R04,OUT5
 ADD SUM2,_ZP,R00
 ADD SUM2,_ZP,R07,OUT4
 ADD SUM2,_ZP,R10

;1/K0=.869369645= 00.110 111 101 000 111 100 000 010

 LDA TEMP,_ZP,R00,OUT3
 LDA I,T,R00,CND4
 SUB TEMP,_ZP,R03,OUT2
 LDA T,T,R00,CND4
 SUB TEMP,_ZP,R08,OUT1
 LDA T,T,R00,CND4
 OUT0

;READY FOR INPUT

 LDA INP,DAR,R03
 ADD SUM1,INP,R08
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R00
 SUB TEMP,SUM2,R03
 SUB TEMP,SUM2,R00

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP1,R05,CND5
 LDA _I2,_I1,R00,CND5
 LDA _I1,INP,R00,CND5
 LDA _I0,_P1,R00,CND5
 SUB _OSC,KP1,R05
 LDA _P1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP+,TEMP,R00,CND5
```

D-12 F35.SRC Listing

```
;*****************************************************************
; FILTER _                          Fc=20Hz        BF 80 dB

 LDA DAR,OUTP$,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.872774990534= 01. 110 111 110 110 111 000 101 110

 LDA SUM1,_P1,R12
 SUB SUM1,SUM1,R03
 ADD SUM1,_P1,R01
 SUB SUM1,_P1,R03
 SUB SUM1,_P1,R09,OUT+
 SUB SUM1,_P1,R11,OUT$

;B2=-.8834856854= -00. 111 000 101 111 000 010 111 001

 LDA TEMP,_T2,R07,OUT$

 SUB DAR,DAR,R00,IN+

 ADD TEMP,_T2,R08,IN+
 ADD TEMP,TEMP,R10
 SUB TEMP,_T2,R12
 ADD TEMP,_T2,R00
 SUB TEMP,_T2,R03
 SUB SUM1,TEMP,R00,CVT5

 ADD DAR,KM2,R00,CND6  ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R07
 ADD SUM1,_Z2,R08,CVT7

;C1=.886095240726= 00.111 000 101 101 011 100 100 011

 LDA SUM2,_ZP,R00
 SUB SUM2,_ZP,R03,CVT6
 ADD SUM2,SUM2,R13
 ADD SUM2,_ZP,R06,CVT5
 SUB SUM2,_ZP,R08
 SUB SUM2,_ZP,R11,CVT4
 SUB SUM2,_ZP,R12

;1/K0=.639a770352= 00.101 000 111 100 000 111 011 111

 LDA TEMP,_ZP,R01,CVT3
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R03,CVT2
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R06,CVT1
 LDA T,T,R00,CND4
 SUB

;READY FOR INPUT

 LDA INP,DAR,R02
 ADD SUM1,INP,R08
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R03
 ADD TEMP,SUM2,R06

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP3,R05,CND8
 LDA _Z2,_Z1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _T2,_P1,R00,CND5
 SUB _OSC,KP1,R05
 LDA _P1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP+,TEMP,R00,CND5
```

D-13 F20.SRC Listing

ORIGINAL PAGE IS
OF POOR QUALITY

```
;*****************************************************************
; FILTER 1                              Fc=15Hz     @ 30 Hz

 LDA DAR,OUTP*,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.872774990534= 01. 110 111 110 110 111 000 101 110

 LDA SUM1,_P1,R12
 SUB SUM1,SUM1,R03
 ADD SUM1,_P1,L01
 SUB SUM1,_P1,R05
 SUB SUM1,_P1,R09,OUT+
 SUB SUM1,_P1,R11,OUT*

;B2=-.8864856854= -00. 111 000 101 111 000 010 111 001

 LDA TEMP,_T2,R07,OUT*

 SUB DAR,DAR,R00,IN+

 ADD TEMP,_T2,R08,IN+
 ADD TEMP,TEMP,R10
 SUB TEMP,_T2,R12
 ADD TEMP,_T2,R00
 SUB TEMP,_T2,R03
 SUB SUM1,TEMP,R00,CVT5

 ADD DAR,KM2,R00,CND0   ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R07
 ADD SUM1,_Z2,R08,CVT7

;C1=.88e09524072E= 00.111 000 101 101 011 100 100 011

 LDA SUM2,_ZP,R00
 SUB SUM2,_ZP,R03,CVT6
 ADD SUM2,SUM2,R13
 ADD SUM2,_ZP,R06,CVT5
 SUB SUM2,_ZP,R08
 SUB SUM2,_ZP,R11,CVT4
 SUB SUM2,_ZP,R12

;1/K0=.6396770352= 00.101 000 111 100 000 111 011 111

 LDA TEMP,_ZP,R01,CVT3
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R05,CVT2
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R06,CVT1
 LDA T,T,R00,CND4
 CVT0

;READY FOR INPUT

 LDA INP,DAR,R02
 ADD SUM1,INP,R08
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R03
 ADD TEMP,SUM2,R06

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP4,R05,CND5
 LDA _OSC,_Z1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _T2,_P1,R00,CND5
 SUB _OSC,KP1,R05@
 LDA _P1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP+,TEMP,R00,CND5
```

D-14 F16.SRC Listing

```
;*************************************************************
; FILTER _                          Fc=14Hz      BF 7.
 LDA DAR,OUTP*,R00
 ADD DAR,KP1,R01
;COEFFICIENT CALCULATIONS
;B1=1.83933e203= 01. 111 000 111 010 101 110 001 010
 LDA SUM1,_P1,R06
 SUB SUM1,_P1,R09
 ADD SUM1,_P1,R12
 ADD SUM1,SUM1,R05
 ADD SUM1,_P1,L01,OUT4
 SUB SUM1,_P1,R03,OUT*
;B2=-.899911152= -00. 111 001 100 110 000 010 010 100
 LDA TEMP,_T2,R05,OUT*
 SUB DAR,DAR,R00,IN+
 SUB TEMP,_T2,R07,IN+
 ADD TEMP,TEMP,R04
 ADD TEMP,_T2,R00
 SUB TEMP,_T2,R03
 NOP
 SUB SUM1,TEMP,R00,CVT5
 ADD DAR,KM2,R00,CND0  ;ANALOG
;A1 AND A2 COEFFICIENTS
 ADD SUM1,_Z1,R07
 ADD SUM1,_Z2,R08,CVT7
;C1=.899645867= 00.111 001 100 100 111 100 110 001
 LDA SUM2,_ZP,R00
 SUB SUM2,_ZP,R04,CVT6
 ADD SUM2,SUM2,R12
 SUB SUM2,_ZP,R04,CVT5
 ADD SUM2,_ZP,R09
 ADD SUM2,_ZP,R07,CVT4
 ADD SUM2,_ZP,R10
;1/K0=.869369648= 00.110 111 101 000 111 100 000 010
 LDA TEMP,_ZP,R00,CVT3
 LDA T,T,R00,CND4
 SUB TEMP,_ZP,R03,CVT2
 LDA T,T,R00,CND4
 SUB TEMP,_ZP,R08,CVT1
 LDA T,T,R00,CND4
 CVT0
;READY FOR INPUT
 LDA INP,DAR,R03
 ADD SUM1,INP,R08
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R00
 SUB TEMP,SUM2,R03
 SUB TEMP,SUM2,R06
;TIME DELAY
 LDA DAR,_OSC,R00
 LDA _OSC,KP4,R05,CND5
 LDA _Z2,_Z1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _T2,_P1,R00,CND5
 SUB _OSC,KP1,R05@
 LDA _P1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA OUTP+,TEMP,R00,CND5
```

D-15 F14.SRC Listing

ORIGINAL PAGE IS
OF POOR QUALITY

```
;*************************************************************
; FILTER -                          Fc=10Hz       BF 80 Hz

 LDA DAR,OUTP$,R00
 ADD DAR,KP1,R01

;COEFFICIENT CALCULATIONS

;B1=1.87277499534= 01. 110 111 110 110 111 000 101 110

 LDA SUM1,_P1,R12
 SUB SUM1,SUM1,R03
 ADD SUM1,_P1,R01
 SUB SUM1,_P1,R05
 SUB SUM1,_P1,R09,OUT4
 SUB SUM1,_P1,R11,OUT$

;B2=-.8864856854= -00. 111 000 101 111 000 010 111 001

 LDA TEMP,_T2,R07,OUT$

 SUB DAR,DAR,R00,IN+

 ADD TEMP,_T2,R08,IN+
 ADD TEMP,TEMP,R10
 SUB TEMP,_T2,R12
 ADD TEMP,_T2,R00
 SUB TEMP,_T2,R03
 SUB SUM1,TEMP,R00,CVT5

 ADD DAR,KM2,R00,CND6  ;ANALOG

;A1 AND A2 COEFFICIENTS

 ADD SUM1,_Z1,R07
 ADD SUM1,_Z2,R08,CVT7

;A1=.8860952407?6= 00.111 000 101 101 011 100 100 011

 LDA SUM2,_ZP,R00
 SUB SUM2,_ZP,R03,CVT6
 ADD SUM2,SUM2,R13
 ADD SUM2,_ZP,R09,CVT5
 SUB SUM2,_ZP,R08
 SUB SUM2,_ZP,R11,CVT4
 SUB SUM2,_ZP,R12

;1/K0=.039677035?= 00.101 000 111 100 000 111 011 111

 LDA TEMP,_ZP,R01,CVT3
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R03,CVT2
 LDA T,T,R00,CND4
 ADD TEMP,_ZP,R06,CVT1
 LDA T,T,R00,CND4
 CVT0

;READY FOR INPUT

 LDA INP,DAR,R02
 ADD SUM1,INP,R05
 ADD SUM2,SUM1,R02
 ADD TEMP,SUM2,R01
 ADD TEMP,SUM2,R03
 ADD TEMP,SUM2,R06

;TIME DELAY

 LDA DAR,_OSC,R00

 LDA _OSC,KP7,R05,CND5
 LDA _Z1,_Z1,R00,CND5
 LDA _Z1,INP,R00,CND5
 LDA _Z2,_P1,R00,CND5
 SUB _OSC,KP1,R05?
 LDA _P1,SUM1,R00,CND5
 LDA _ZP,SUM2,R00,CND5
 LDA SU_P#,TEMP,R00,CND5
```

D-16 FIO.SRC Listing

```
LINE        SOURCE STATEMENT

  1 $       TITLE('ALARM PROGRAM')
  2         NAME    BEEP
  3         EXTRN   CO,CSTS,ISIS
  4 ;
  5         CSEG
  6 BELL    EQU     007H
  7 EXIT    EQU     9
  8 ;
  9 START:
 10         CALL    CSTS
 11         RRC
 12         JC      DONE
 13         MVI     C,BELL
 14         LXI     H,0C000H
 15 ;
 16 BEEP:
 17         INR     L
 18         JNZ     BEEP
 19         CALL    CO
 20         INR     H
 21         JNZ     BEEP
 22 ;
 23 NBEEP:
 24         LXI     H,00H
 25
 26 LOOP:
 27         INR     L
 28         JNZ     LOOP
 29         INR     H
 30         JNZ     LOOP
 31         JMP     START
 32 ;
 33 DONE:
 34         MVI     C,EXIT
 35         LXI     D,EBLK
 36         CALL    ISIS
 37         JMP     START
 38 ;
 39 ;
 40         DSEG
 41 ;
 42 EBLK:
 43         DW      ESTAT
 44 ESTAT:
 45         DS      2
 46 ;
 47         END     START
```

Listing D-17 BEEP Program Source Code

```
LINE        SOURCE STATEMENT

    1           NAME HELP
    2           EXTRN CO,ISIS
    3
    4           CSEG
    5
    6 EXIT      EQU     9
    7 READ      EQU     3
    8 OPEN      EQU     0
    9 CLOSE     EQU     1
   10
   11 READC:
   12           MVI     C,READ
   13           LXI     D,CBLK
   14           CALL    ISIS
   15           LDA     RSTAT
   16           ORA     A
   17           JNZ     EXCEPT
   18           LXI     H,CBUF
   19 SPCX:
   20           MOV     A,M
   21           CPI     0DH
   22           JZ      OPENF
   23           CPI     20H
   24           JNZ     MVE
   25           INX     H
   26           JMP     SPCX
   27 MVE:
   28           INX     H
   29           MOV     B,M
   30           INX     H
   31           MOV     C,M
   32           LXI     H,EXT
   33           MOV     M,A
   34           INX     H
   35           MOV     M,B
   36           INX     H
   37           MOV     M,C
   38
   39 OPENF:
   40           MVI     C,OPEN
   41           LXI     D,OBLK
   42           CALL    ISIS
   43           LDA     OSTAT
   44           ORA     A
   45           JNZ     EXCEPT
   46           LHLD    OAFT
   47           SHLD    RAFT
   48           SHLD    CAFT
   49 READF:
   50           MVI     C,READ
   51           LXI     D,RBLK
   52           CALL    ISIS
   53           LDA     RSTAT
   54           ORA     A
   55           JNZ     EXCEPT
   56 WRITF:
   57           LXI     H,IBUF
   58 LOOP:
   59           MOV     C,M
   60           CALL    CO
   61           MVI     A,0FFH
   62           CMP     M
   63           JZ      CLOSF
   64           INX     H
   65           JMP     LOOP
   66 CLOSF:
   67           MVI     C,CLOSE
   68           LXI     D,CLBLK
   69           CALL    ISIS
   70           LDA     CSTAT
   71           ORA     A
   72           JNZ     EXCEPT
   73 DONE:
   74           MVI     C,EXIT
   75           LXI     D,EBLK
   76           CALL    ISIS
   77           JMP     OPENF
   78
   79 EXCEPT:
   80           LXI     H,BUF
   81 LUP:
   92           MOV     C,M
   83           CALL    CO
   84           INX     H
   85           MVI     A,0FEH
   86           CMP     M
   87           JZ      DONE
   88           JMP     LUP
   89
   90
   91           DSEG
   92 BUF:
   93           DB      0DH,0AH,'***SYNTAX ERROR***',0DH,0AH

   94           DB      'TYPE HELP FOR COMMAND LISTING',0DH,0AH,0AH,0FEH

   95 OBLK:
   96           DW      OAFT
   97           DW      OFILE
   98 ACCESS:   DW      1
   99 ECHO:     DW      0
  100           DW      OSTAT
  101 OAFT:     DS      2
  102 OSTAT:    DS      2
  103 OFILE:
  104           DB      ':F0:HELP.'
  105 EXT:      DB      'HEL'
  106 CBUF:     DS      16
  107           DB      0FEH
  108
  109 RBLK:
  110 RAFT:     DS      2
  111           DW      IBUF
  112 RCNT:     DW      2048
  113           DW      ACTUAL
  114           DW      RSTAT
  115 ACTUAL:   DS      2
  116 RSTAT:    DS      2
  117 IBUF:     DS      2048
  118 CBLK:
  119           DW      1
  120           DW      CBUF
  121           DW      16
  122           DW      ACTUAL
  123           DW      RSTAT
  124
  125 EBLK:     DW      ESTAT
  126 ESTAT:    DS      2
  127
  128 CLBLK:
  129 CAFT:     DS      2
  130           DW      CSTAT
  131 CSTAT:    DS      2
  132
  133 END READC
```

Listing D-18 HELP Program Source Code

```
2920 COMMAND HELP FILE

DOCUMENTATION AVAILABLE UNDER HELP COMMAND

        HELP COMmand
             HELP COM GIVES A SHORT LIST OF THE COMMON SYSTEM COMMANDS.

        HELP FREquencies
             HELP FRE CONTAINS A LISTING OF THE AVAILABLE FILTER
          CORNER FREQUENCIES AND THEIR RESPECTIVE SAMPLING RATES.

        HELP HELp
             HELP HEL OUTPUTS THE CONTENTS OF THIS FILE.

        HELP PROcedure
             HELP PRO OUTLINES THE PROPER SEQUENCE FOR USING
           THE DO FILTER COMMAND.

        HELP COPy
             HELP COP LISTS COMMANDS USED TO COPY FILES TO THE
          LINE PRINTER.

        HELP EXAmple
             HELP EXA LISTS A SHORT 2920 PROGRAMMING EXAMPLE.
```


Listing D-19 Listing of HELP.HEL File

2920 DIGITAL FILTER HELP FILE:

COMMANDS

DO FILTER(F1 ,F2 ,F3 ,F4 [,$LIST])

THE DO FILTER COMMAND ASSEMBLES FOUR THIRD ORDER BUTTERWORTH FILTERS WHICH HAVE CORNER FREQUENCIES SPECIFIED BY F1,F2,F3 AND F4. IF THE LIST OPTION IS SPECIFIED THE ASSEMBLED CODE LISTING IS INCLUDED IN THE LIST FILE. F1,F2,F3 AND F3 ARE SELECTED FROM FILTERS LISTED BY HELP.FIL COMMAND.

DO SIM(FILTER)

DO SIM(FILTER) SETS UP THE 2920 SIMULATOR PROGRAM FOR SIMULATION OF AN ASSEMBLED FILTER. CONSULT THE 2920 SIMULATOR USER'S GUIDE FOR ADDITIONAL INFORMATION.

COPY FILTER.LST TO :NN:

COPY FILTER.LST COPIES THE ASSEMBLED LIST FILE TO THE LINE PRINTER (:NN:=:LP:),OR CONSOLE (:NN:=:CO:). FILTER.LST CONTAINS FILTER PINOUT AND CODE LISTING IF SPECIFIED BY $LIST OPTION.

HELP FREQUENCIES

HELP FRE LISTS AVAILIBLE FILTER CORNER FREQUENCIES AND THEIR CORRESPONDING SAMPLE RATES.

Listing D-20 Listing of HELP.COM File

2920 AVAILABLE FILTER MODULES:

FC (HZ)	SAMPLE FREQUENCY (HZ)	INPUT SAMPLING DIVISOR	BASE FILTER (HZ)
300	4166	1	300
150	2083	2	300
100	1388	3	300
80	4166	1	80
75	1042	4	300
70	4166	1	70
60	833	5	300
50	694	6	300
40	2083	2	80
35	2083	2	70
20	1042	4	80
16	833	5	80
14	833	5	70
10	521	8	80

FC = CORNER FREQUENCY OF THE THIRD ORDER BUTTERWORTH FILTER
FS = SAMPLING FREQUENCY
SAMPLING DIVISOR = THE FILTER ACCEPTS AN INPUT SAMPLE EVERY NTH PASS.
BASE FILTER = INDICATES WHICH FILTER MODULE WAS USED FOR DESIGN.

Listing D-21 Listing of HELP.FRE File

2920 FILTER PREPARATION PROCEDURE:

1. ERASE EPROM MEMORY OF 2920 WITH 20 MINUTE EXPOSURE UNDER UV LIGHT.

2. POWER UP AND RESET UNIVERSAL PROM PROGRAMMER, THEN PLACE ERASED 2920 IN PROGRAMMING SOCKET #1.

3. SELECT DESIRED FILTER CORNER FREQUENCIES FROM THE FILTERS LISTED UNDER HELP FRE. FOUR FILTER MODULES ARE REQUIRED TO COMPLETE A PROPERLY ASSEMBLED FILTER.

4. ASSEMBLE FILTER MODULES WITH THE COMMAND DO FILTER(F1,F2,F3,F4). F1, F1, F2, AND F4 ARE THE CORNER FREQUENCIES OF THE FOUR FILTER MODULES SELECTED IN STEP 3.

5. AFTER ASSEMBLING THE MODULES THE PROGRAM OUTPUTS A STATUS TABLE, THEN SOUNDS AN ALARM. IF THE ASSEMBLY PROCESS WAS SUCCESSFUL THE TABLE WILL LIST 0 ERRORS, 0 WARNINGS AND ROMSIZE = 192.

6. IF A PROBLEM OCCURS IN ASSEMBLY, MANUALLY RESET THE SYSTEM WITH THE RESET KEY AND TRY AGAIN. IF NO ERRORS OCCUR, PRESS SPACE BAR AND THE PROGRAM WILL RESUME WITH THE EPROM BURNING.

7. #F0:DO RESTORE .. INDICATES COMPLETION OF PROGRAMMING PROCESS.

Listing D-22 Listing of HELP.PRO File

```
2920 HELP COPY FILE

COPY COMMANDS

        COPY HELP.??? TO :LP:
             THIS COMMANDS COPIES ALL THE HELP FILES TO THE
          LINE PRINTER.

        COPY FILTER.LST TO :LP:
             COPIES ASSEMBLED FILTER LISTING TO THE LINE PRINTER.
          IF THE $LIST OPTION WAS SPECIFIED IN THE DO FILTER
          COMMAND, THE LISTING WILL INCLUDE THE CODE LISTING.

        COPY FILTER.LST TO :CO:
             COPIES ASSEMBLED FILTER LISTING TO CONSOLE
```


Listing D-23 Listing of HELP.COP File

2920 PROGRAMMING EXAMPLE

```
-
-DO FILTER(10,50,300,80,$LIST)
          .          .
          .          .
          .          .
          .          .
ASSEMBLY COMPLETE
ERRORS    =     0
WARNINGS  =     0
RAMSIZE   =    30
ROMSIZE   =   192
          .
          .
          .
-BEEP1

-UPM
  .
  .

-:F0:DO RESTORE xxxxxxxxxxx
```

THIS COMMAND SPECIFIES A FILTER WITH CORNERS FREQUENCIES AT 10,50,300 AND 80 HZ. THE $LIST OPTION IS SPECIFIED SO THE CODE LISTING WILL BE INCLUDED IN FILTER.LST.

IF THE ASSEMBLY PROCESS IS SUCCESSFUL THE ERROR TABLE WILL APPEAR AS SHOWN. ASSEMBER ERRORS COULD BE CAUSED BY AN INPROPER CORNER FREQUENCY.

IF AN ERROR OCCURS, RESET TO THE ISIS SYSTEM BY PRESSING RESET ON THE CONSOLE.

IF THERE ARE NO ERRORS, PRESS THE SPACE BAR TO CONTINUE PROGRAMMING.

IN THIS SECTION THE EPROM OF THE 2920 IS BURNED. IT IS ASSUMED THAT AN ERASED 2920 IS IN SOCKET 1 OF THE PROGRAMMER

INDICATES COMPLETION OF 2920 PROGRAMMING!

Listing D-24 Listing of HELP.EXA File

APPENDIX E

SUPPORT PROGRAMS

Appendix E contains listings and sample runs of three support programs used in this investigation. TRCLC calculates the transfer function of the analog circuit of Figure 1-2 as a function of a normalized value of R1. The program illustrates the transfer function drift for various values of R1.

MAGCK calculates the frequency response of the third order Butterworth transfer function H(s) and the frequency response of the Bilinear Transform resultant H(z). The output illustrates the warping properties of the Bilinear Transform and the theoretical frequency response characteristics of the digital filter.

MAGND calculates the magnitude response of the numerator and denominator of H(z) separately. The overshoot characteristics of the recursive and non-recursive direct form algorithms are illustrated by the results.

LISTING E-1: TRCLC.BAS SOURCE CODE

```
100 ! TRCLC IS A PROGRAM WHICH ALLOWS THE USER TO
101 ! ENTER A NORMALIZED VALUE OF R1, THE PROGRAM
102 ! THEN CALCULATES THE TRANSFER FUNCTION OF THE
103 ! RESULTING FUNCTION.  THE BASIC TRANSFER FUNCTION
104 ! IS DERIVED FROM THE CIRCUIT DFRF USES AS A PRE-
105 ! SAMPLING FILTER.
106 !
110   R2=R3=C1=C2=C3=1
111   PRINT " Enter normalized value of R1";
112   INPUT R1
113   PRINT " Enter delta frequency";
114   INPUT D
115   A=R1/(2*PI)^3%\B=2*R1/(2*PI)^2%\C=2*R1/(2*PI)
116   PRINT "FREQ-(HZ)","S-TRAN(DB)","S-PH(DEG)"
120   FOR N%=0% TO 25%
130   F1=N%*D
140   W1=2.*PI*F1
150   T1=1.-B*W1^2%
151   T2=C*W1-A*W1^3%
160   M1=1./SQR(T1*T1+T2*T2)
170   P1=-ATN(T2/T1)
180   IF T1<0 THEN P1=P1-PI*SGN(T2)
190   M1=20.*LOG10(M1)
200   P1=P1*180./PI
210   PRINT F1,M1,P1
220   NEXT N%
221   PRINT\PRINT\PRINT\PRINT
230   GOTO 100
240   END
```

```
Enter normalized value of R1? 1.1

Enter delta frequency? .1
```

FREQ-(HZ)	S-TRAN(DB)	S-PH(DEG)
0	0	0
.1	-.190723E-1	-12.6162
.2	-.761015E-1	-25.3051
.3	-.172346	-38.1642
.4	-.315523	-51.3264
.5	-.526579	-64.9424
.6	-.84524	-79.1223
.7	-1.32885	-93.838
.8	-2.03793	-108.825
.9	-3.00909	-123.575
1	-4.23246	-137.49
1.1	-5.65374	-150.095
1.2	-7.19881	-161.173
1.3	-8.79901	-170.737
1.4	-10.4033	-178.934
1.5	-11.979	174.038
1.6	-13.5077	167.988
1.7	-14.9801	162.744
1.8	-16.3929	158.166
1.9	-17.7462	154.14
2	-19.0417	150.573
2.1	-20.2824	147.391
2.2	-21.4712	144.536
2.3	-22.6115	141.959
2.4	-23.7064	139.621
2.5	-24.759	137.49

SAMPLE RUN E-1: TRCLC.BAS for +10% DRIFT

```
Enter normalized value of R1? 1.0

Enter delta frequency? .1
```

FREQ-(HZ)	S-TRAN(DB)	S-PH(DEG)
0	0	0
.1	-.434294E-5	-11.4785
.2	-.27794E-3	-23.0782
.3	-.316485E-2	-34.9451
.4	-.177524E-1	-47.2648
.5	-.673338E-1	-60.2551
.6	-.19804	-74.1161
.7	-.483054	-88.916
.8	-1.01109	-104.432
.9	-1.851	-120.066
1	-3.0103	-135
1.1	-4.42724	-148.535
1.2	-6.00536	-160.331
1.3	-7.65431	-170.389
1.4	-9.30925	-178.901
1.5	-10.9309	173.884
1.6	-12.4986	167.731
1.7	-14.0032	162.436
1.8	-15.4422	157.839
1.9	-16.8166	153.812
2	-18.1291	150.255
2.1	-19.3835	147.09
2.2	-20.5835	144.253
2.3	-21.7329	141.696
2.4	-22.8353	139.377
2.5	-23.8942	137.265

SAMPLE RUN E-2: TRCLC.BAS for ZERO DRIFT

```
Enter normalized value of R1? .9

Enter delta frequency? .1
```

FREQ-(HZ)	S-TRAN(DB)	S-PH(DE)
0	0	0
.1	.156593E-1	-10.3362
.2	.627646E-1	-20.8154
.3	.140392	-31.6079
.4	.242385	-42.9332
.5	.349601	-55.0691
.6	.418226	-68.3236
.7	.367338	-82.9291
.8	.791653E-1	-98.8235
.9	-.564731	-115.415
1	-1.61368	-131.634
1.1	-3.00892	-146.419
1.2	-4.62435	-159.198
1.3	-6.33601	-169.928
1.4	-8.05727	-178.858
1.5	-9.73885	173.686
1.6	-11.357	167.401
1.7	-12.9028	162.045
1.8	-14.3749	157.426
1.9	-15.7758	153.401
2	-17.1096	149.859
2.1	-18.3811	146.715
2.2	-19.595	143.902
2.3	-20.7556	141.37
2.4	-21.8673	139.076
2.5	-22.9337	136.987

SAMPLE RUN E-3: TRCLC.BAS for -10% DRIFT

LISTING E-2: MAGCK.BAS SOURCE CODE

```
100   ! MAG(NITUDE) C(HEC)K CALCULATES THE EXPECTED
101   ! VALUE OF THE S DOMAIN FUNCTION AND THE ACTUAL
102   ! Z DOMAIN RESULTS USING THE ROOT POLE PLOTS
103   ! DESCRIBED IN ANTONIOU.
104   !
109     PRINT
110     INPUT "Enter desired corner frequency (Hz) ";F0
111     W0=2.*PI*F0
112     N1=F0/10
113     F=3.2E6/768
114     GOSUB 300
115     PRINT  "FREQ-(HZ)","S-TRAN(DB)","Z-TRAN(DB)";
116     PRINT  "    S-PH(DEG)","Z-PH(DEG)"
140     G=2**-4
150     FOR N%=0% TO 25%
160     F1=N%*N1
161     W1=2.*PI*F1
162     A =COS(W1/F)
163     B =SIN(W1/F)
170     X=FNR(A0,A1,A2,A3)
171     Y=FNI(A0,A1,A2,A3)
172     W=FNR(K0,K1,K2,K3)
173     Z=FNI(K0,K1,K2,K3)
180     M=SQR((X^2%+Y^2%)/(W^2%+Z^2%))
181     P1=ATN(Y/X)
182     IF X<0. THEN P1=P1+PI*SGN(Y)
190     P2=ATN(Z/W)
191     IF W<0. THEN P2=P2+PI*SGN(Z)
195     P=P1-P2
196     IF P>PI THEN P=P-2*PI
197     IF P<-PI THEN P=P+2*PI
200     T1=W0^3%-2.*W0*W1^2%
201     T2=2.*W0^2%*W1-W1^3%
210     M1=W0^3%/SQR(T1*T1+T2*T2)
211     P1=-ATN(T2/T1)
212     IF T1<0 THEN P1=P1-PI*SGN(T2)
213     M1=.8*M1
214     M =G*M
215     M1=20*LOG10(M1)
216     M =20*LOG10(M)
217     P1=P1*180./PI
218     P =P *180./PI
220     PRINT F1,M1,M,P1,P
230     NEXT N%
240     PRINT
241     STOP
250     DEF* FNR(C0,C1,C2,C3)
251     FNR=C0*A^3%+C1*A^2%+C2*A-3.*C0*A*B^2%-C1*B^2%+C3
```

```
252     FNEND
265     DEF* FNI(C0,C1,C2,C3)
266     FNI=-C0*B^3%+3.*C0*A^2%*B+2.*C1*A*B+C2*B
267     FNEND
300     L=W0/(2*F*TAN(W0/(2*F)))
310     R=L*F/W0
320     K0=8*R**3+8*R**2+4*R+1
330     K1=-24.*R**3-8.*R**2+4.*R+3.
340     K2=24.*R**3-8.*R**2-4.*R+3.
350     K3=-8.*R**3+8.*R**2-4.*R+1.
360     K3=K3/K0
370     K2=K2/K0
380     K1=K1/K0
390     A0=.8/K0*2**4
400     A1=.8*3/K0*2**4
410     A2=A1
420     A3=A0
421     K0=1.0
422     PRINT
423     PRINT "Z-TRANSFORM COEFFICIENTS:"
424     PRINT
425     PRINT "A(0)= ";A0
426     PRINT "A(1)= ";A1,"K(1)= ";K1
427     PRINT "A(2)= ";A2,"K(2)= ";K2
428     PRINT "A(3)= ";A3,"K(3)= ";K3
429     PRINT
430     PRINT "Rsc= ";R
431     PRINT
440     RETURN
500     END
```

```
Enter desired corner frequency (Hz) ? 300

Z-TRANSFORM COEFFICIENTS:

A(0)=   .988407E-1
A(1)=   .296522                 K(1)= -2.10202
A(2)=   .296522                 K(2)=  1.56514
A(3)=   .988407E-1              K(3)= -.401342

Rsc=  2.17266
```

FREQ-(HZ)	S-TRAN(DB)	Z-TRAN(DB)	S-PH(DEG)	Z-PH(DEG)
0	-1.9382	-1.9382	0	0
30	-1.9382	-1.9382	-11.4785	-11.2833
60	-1.93848	-1.93845	-23.0782	-22.6934
90	-1.94137	-1.94108	-34.9451	-34.3812
120	-1.95595	-1.95447	-47.2648	-46.538
150	-2.00553	-2.00052	-60.2551	-59.3927
180	-2.13624	-2.12375	-74.1161	-73.1691
210	-2.42125	-2.39753	-88.916	-87.9778
240	-2.94929	-2.9159	-104.432	-103.648
270	-3.7892	-3.75945	-120.066	-119.607
300	-4.9485	-4.9485	-135	-135
330	-6.36544	-6.42695	-148.535	-149.047
360	-7.94356	-8.09529	-160.331	-161.339
390	-9.59251	-9.85655	-170.389	-171.84
420	-11.2475	-11.6401	-178.901	179.261
450	-12.8691	-13.4031	173.884	171.707
480	-14.4368	-15.1232	167.731	165.248
510	-15.9414	-16.7904	162.436	159.673
540	-17.3804	-18.4022	157.839	154.814
570	-18.7548	-19.9599	153.812	150.536
600	-20.0673	-21.4666	150.255	146.737
630	-21.3217	-22.9263	147.09	143.336
660	-22.5217	-24.3434	144.253	140.267
690	-23.6711	-25.722	141.696	137.48
720	-24.7735	-27.0663	139.377	134.934
750	-25.8324	-28.3801	137.265	132.595

SAMPLE RUN E-4: MAGCK.BAS for a 300 Hz FILTER

```
Enter desired corner frequency (Hz) ? 80

Z-TRANSFORM COEFFICIENTS:

A(0)=   .249874E-2
A(1)=   .749622E-2          K(1)= -2.75887
A(2)=   .749622E-2          K(2)=  2.54594
A(3)=   .249874E-2          K(3)= -.785511

Rsc=  8.27926

FREQ-(HZ)   S-TRAN(DB)    Z-TRAN(DB)    S-PH(DEG)     Z-PH(DEG)
 0          -1.9382       -1.9382        0             0
 8          -1.9382       -1.9382       -11.4785      -11.4646
 16         -1.93848      -1.93848      -23.0782      -23.051
 24         -1.94137      -1.94134      -34.9451      -34.9052
 32         -1.95595      -1.95584      -47.2648      -47.2133
 40         -2.00553      -2.00517      -60.2551      -60.1942
 48         -2.13624      -2.13534      -74.1161      -74.0493
 56         -2.42125      -2.41956      -88.916       -88.85
 64         -2.94929      -2.94693      -104.432      -104.377
 72         -3.7892       -3.78712      -120.066      -120.034
 80         -4.9485       -4.9485       -135          -135
 88         -6.36544      -6.3697       -148.535      -148.57
 96         -7.94356      -7.954        -160.331      -160.401
 104        -9.59251      -9.61062      -170.389      -170.49
 112        -11.2475      -11.2743      -178.901      -179.029
 120        -12.8691      -12.9056       173.884       173.733
 128        -14.4368      -14.4835       167.731       167.558
 136        -15.9414      -15.999        162.436       162.244
 144        -17.3804      -17.4494       157.839       157.629
 152        -18.7548      -18.8359       153.812       153.584
 160        -20.0673      -20.1611       150.255       150.011
 168        -21.3217      -21.4287       147.09        146.829
 176        -22.5217      -22.6426       144.253       143.977
 184        -23.6711      -23.8065       141.696       141.404
 192        -24.7735      -24.9241       139.377       139.07
 200        -25.8324      -25.9986       137.265       136.943
```

SAMPLE RUN E-5: MAGCK.BAS for a 80 Hz FILTER

```
Enter desired corner frequency (Hz) ? 70

Z-TRANSFORM COEFFICIENTS:

A(0)=  .169798E-2
A(1)=  .509395E-2          K(1)= -2.78898
A(2)=  .509395E-2          K(2)=  2.59964
A(3)=  .169798E-2          K(3)= -.809601

Rsc=  9.46471
```

FREQ-(HZ)	S-TRAN(DB)	Z-TRAN(DB)	S-PH(DEG)	Z-PH(DEG)
0	-1.9382	-1.9382	0	0
7	-1.9382	-1.9382	-11.4785	-11.4679
14	-1.93848	-1.93848	-23.0782	-23.0573
21	-1.94137	-1.94135	-34.9451	-34.9145
28	-1.95595	-1.95587	-47.2648	-47.2254
35	-2.00553	-2.00526	-60.2551	-60.2084
42	-2.13624	-2.13555	-74.1161	-74.065
49	-2.42125	-2.41996	-88.916	-88.8655
56	-2.94929	-2.94748	-104.432	-104.39
63	-3.7892	-3.78761	-120.066	-120.042
70	-4.9485	-4.9485	-135	-135
77	-6.36544	-6.3687	-148.535	-148.562
84	-7.94356	-7.95155	-160.331	-160.384
91	-9.59251	-9.60637	-170.389	-170.467
98	-11.2475	-11.268	-178.901	-178.999
105	-12.8691	-12.897	173.884	173.769
112	-14.4368	-14.4726	167.731	167.598
119	-15.9414	-15.9855	162.436	162.289
126	-17.3804	-17.4332	157.839	157.678
133	-18.7548	-18.8168	153.812	153.638
140	-20.0673	-20.139	150.255	150.068
147	-21.3217	-21.4036	147.09	146.89
154	-22.5217	-22.6142	144.253	144.042
161	-23.6711	-23.7746	141.696	141.473
168	-24.7735	-24.8886	139.377	139.142
175	-25.8324	-25.9595	137.265	137.018

SAMPLE RUN E-6: MAGCK.BAS for a 70 Hz FILTER

LISTING E-3: MAGND.BAS SOURCE CODE

```
100   ! MAG(NITUDE) N(UMERATOR) & D(ENOMINATOR) CALCU-
101   ! LATES THE PHASE AND MAGNITUDE RESPONSE OF THE
102   ! RECURSIVE RESULTS USING THE ROOT POLE PLOTS
103   ! DESCRIBED IN ANTONIOU.
104   !
109     PRINT
110     INPUT "Enter desired corner frequency (Hz) ";F0
111     W0=2.*PI*F0
112     N1=F0/10
113     F=3.2E6/768
114     GOSUB 300
115     PRINT "FREQ(Hz)","NUM (dB)","DEN (dB)","MAG (dB)"
140     G=2**-4
150     FOR N%=0% TO 25%
160     F1=N%*N1
161     W1=2.*PI*F1
162     A =COS(W1/F)
163     B =SIN(W1/F)
170     X=FNR(A0,A1,A2,A3)
171     Y=FNI(A0,A1,A2,A3)
172     W=FNR(K0,K1,K2,K3)
173     Z=FNI(K0,K1,K2,K3)
180     M1=SQR(X^2%+Y^2%)
181     P1=ATN(Y/X)
182     IF X<0. THEN P1=P1+PI*SGN(Y)
185     M2=1/SQR(Z^2%+W^2%)
190     P2=ATN(Z/W)
191     IF W<0. THEN P2=P2+PI*SGN(Z)
213     M1=20*LOG10(G*M1)
214     M2=20*LOG10(M2)
217     P1=P1*180./PI
218     P2=P2*180./PI
220     PRINT F1,M1,M2,M1+M2
230     NEXT N%
240     PRINT
241     STOP
250     DEF* FNR(C0,C1,C2,C3)
251     FNR=C0*A^3%+C1*A^2%+C2*A-3.*C0*A*B^2%-C1*B^2%+C3
252     FNEND
265     DEF* FNI(C0,C1,C2,C3)
266     FNI=-C0*B^3%+3.*C0*A^2%*B+2.*C1*A*B+C2*B
267     FNEND
300     L=W0/(2*F*TAN(W0/(2*F)))
310     R=L*F/W0
320     K0=8*R**3+8*R**2+4*R+1
330     K1=-24.*R**3-8.*R**2+4.*R+3.
340     K2=24.*R**3-8.*R**2-4.*R+3.
350     K3=-8.*R**3+8.*R**2-4.*R+1.
```

```
360     K3=K3/K0
370     K2=K2/K0
380     K1=K1/K0
390     A0=.8/K0*2**4
400     A1=.8*3/K0*2**4
410     A2=A1
420     A3=A0
421     K0=1.0
422     PRINT
423     PRINT "Z-TRANSFORM COEFFICIENTS:"
424     PRINT
425     PRINT "A(0)= ";A0
426     PRINT "A(1)= ";A1,"K(1)= ";K1
427     PRINT "A(2)= ";A2,"K(2)= ";K2
428     PRINT "A(3)= ";A3,"K(3)= ";K3
429     PRINT
430     PRINT "Rsc= ";R
431     PRINT
440     RETURN
500     END
```

```
Enter desired corner frequency (Hz) ? 300

Z-TRANSFORM COEFFICIENTS:

A(0)=   .988407E-1
A(1)=   .296522                 K(1)= -2.10202
A(2)=   .296522                 K(2)=  1.56514
A(3)=   .988407E-1              K(3)= -.401342

Rsc=  2.17266

FREQ(Hz)        NUM (dB)        DEN (dB)        MAG (dB)
 0              -26.1219         24.1837        -1.9382
 30             -26.1285         24.1903        -1.9382
 60             -26.1486         24.2101        -1.93845
 90             -26.1819         24.2408        -1.94108
 120            -26.2287         24.2742        -1.95447
 150            -26.2889         24.2884        -2.00052
 180            -26.3626         24.2388        -2.12375
 210            -26.4499         24.0524        -2.39753
 240            -26.5509         23.635         -2.9159
 270            -26.6656         22.9062        -3.75945
 300            -26.7943         21.8458        -4.9485
 330            -26.9369         20.51          -6.42695
 360            -27.0938         18.9985        -8.09529
 390            -27.2651         17.4085        -9.85655
 420            -27.4509         15.8108        -11.6401
 450            -27.6514         14.2483        -13.4031
 480            -27.867          12.7439        -15.1232
 510            -28.0978         11.3074        -16.7904
 540            -28.3441         9.94192        -18.4022
 570            -28.6063         8.64637        -19.9599
 600            -28.8846         7.41794        -21.4666
 630            -29.1793         6.25297        -22.9263
 660            -29.4909         5.14756        -24.3434
 690            -29.8198         4.09783        -25.722
 720            -30.1664         3.10014        -27.0663
 750            -30.5312         2.15108        -28.3801
```

SAMPLE RUN E-7: MAGND.BAS for a 300 Hz FILTER

```
Enter desired corner frequency (Hz) ? 80

Z-TRANSFORM COEFFICIENTS:

A(0)=   .249874E-2
A(1)=   .749622E-2          K(1)= -2.75887
A(2)=   .749622E-2          K(2)=  2.54594
A(3)=   .249874E-2          K(3)= -.785511

Rsc=  8.27926

FREQ(Hz)       NUM (dB)       DEN (dB)       MAG (dB)
 0             -58.0662        56.128        -1.9382
 8             -58.0667        56.1285       -1.9382
 16            -58.0681        56.1296       -1.93848
 24            -58.0705        56.1291       -1.94134
 32            -58.0738        56.1179       -1.95584
 40            -58.078         56.0729       -2.00517
 48            -58.0833        55.9479       -2.13534
 56            -58.0894        55.6699       -2.41956
 64            -58.0965        55.1496       -2.94693
 72            -58.1046        54.3175       -3.78712
 80            -58.1136        53.1651       -4.9485
 88            -58.1236        51.7539       -6.3697
 96            -58.1345        50.1805       -7.954
 104           -58.1464        48.5358       -9.61062
 112           -58.1592        46.8849       -11.2743
 120           -58.173         45.2674       -12.9056
 128           -58.1877        43.7042       -14.4835
 136           -58.2034        42.2044       -15.999
 144           -58.2201        40.7706       -17.4494
 152           -58.2377        39.4018       -18.8359
 160           -58.2563        38.0952       -20.1611
 168           -58.2758        36.8471       -21.4287
 176           -58.2963        35.6537       -22.6426
 184           -58.3178        34.5112       -23.8065
 192           -58.3402        33.4161       -24.9241
 200           -58.3636        32.365        -25.9986
```

SAMPLE RUN E-8: MAGND.BAS for a 80 Hz FILTER

```
Enter desired corner frequency (Hz) ? 70

Z-TRANSFORM COEFFICIENTS:

A(0)=  .169798E-2
A(1)=  .509395E-2          K(1)= -2.78898
A(2)=  .509395E-2          K(2)=  2.59964
A(3)=  .169798E-2          K(3)= -.809601

Rsc=  9.46471

FREQ(Hz)       NUM (dB)       DEN (dB)       MAG (dB)
 0             -61.4219        59.4837       -1.9382
 7             -61.4223        59.4841       -1.9382
 14            -61.4234        59.4849       -1.93848
 21            -61.4252        59.4838       -1.94135
 28            -61.4277        59.4719       -1.95587
 35            -61.431         59.4257       -2.00526
 42            -61.435         59.2994       -2.13555
 49            -61.4397        59.0198       -2.41996
 56            -61.4452        58.4977       -2.94748
 63            -61.4513        57.6637       -3.78761
 70            -61.4582        56.5097       -4.9485
 77            -61.4659        55.0972       -6.3687
 84            -61.4742        53.5227       -7.95155
 91            -61.4833        51.8769       -9.60637
 98            -61.4931        50.2251       -11.268
 105           -61.5037        48.6067       -12.897
 112           -61.5149        47.0424       -14.4726
 119           -61.527         45.5415       -15.9855
 126           -61.5397        44.1065       -17.4332
 133           -61.5532        42.7364       -18.8168
 140           -61.5674        41.4283       -20.139
 147           -61.5823        40.1788       -21.4036
 154           -61.598         38.9838       -22.6142
 161           -61.6144        37.8397       -23.7746
 168           -61.6315        36.7429       -24.8886
 175           -61.6494        35.69         -25.9595
```

SAMPLE RUN E-9: MAGND.BAS for a 70 Hz FILTER

GLOSSARY

A_i	gain of section i
A/D	Analog to Digital
ADC	Analog to Digital Convertors
ALU	Arithmetic Logic Unit
ASP	Analog Signal Processor
C_i	gain scale factor
D/A	Digital to Analog
DAC	Digital to Analog Converters
DAR	Digital Analog Register
DFRF	Dryden Flight Research Facility
E_{rm}	finite multiplication accuracy error
E_{rms}	finite signal accuracy error
EPROM	Electrically Programmable Read Only Memory
F_c	filter corner frequency (Hz)
F_{cry}	processor crystal frequency (Hz)
FIR	Finite Impulse Response
F_{pcr}	program cycle rate (Hz)
F_s	filter sample frequency (Hz)
F_{smax}	maximum 2920 sample rate (Hz)
F_{smin}	minimum acceptable 2920 sample frequency (Hz)
H(s)	analog transfer function
H(z)	discrete transfer function

I/O	Input-Output
IIR	Infinite Impulse Response filter structure
L_w	pre-warping factor
N_m	Number of multiplications
N_{ps}	Number of program steps in filter code
PCM	Pulse Coded Modulation
PSF	Pre-Sampling Filter
PSFSS	Pre-Sampling Filter Support Software
R_i	binary Resolution of input section i
R_s	binary Resolution of signal
RAM	Random Access Memory
R_{sc}	sample rate to corner frequency ratio (Hz/Rad)
s	Laplacean operator
SF_i	binary Scale Factor for section i
SCB	Signal Conditioning Buffers
W_a	analog frequency variable (radians/sec)
W_d	digital frequency variable (radians/sec)
W_c	filter corner frequency (radians/sec)
X(nT)	discrete section input at time nT
Y(nT)	discrete section output at time nT
z	discrete time operator
2920	Intel 2920 Analog Signal Processor