

NAVAL POSTGRADUATE SCHOOL

Monterey, California

S DTIC
ELECTE
JUN 06 1989
D ^{CS} **D**



AD-A208 485

THESIS

ALGORITHMS FOR COMPUTER AIDED
DESIGN OF DIGITAL FILTERS

by

Thomas H. Rich

December 1988

Thesis Advisor:

Robert Strum

Approved for public release; distribution is unlimited

89 6 05 118

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE				Form Approved OMB No 0704-0188	
1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED			1b RESTRICTIVE MARKINGS		
2a SECURITY CLASSIFICATION AUTHORITY			3 DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.		
2b DECLASSIFICATION/DOWNGRADING SCHEDULE					
4. PERFORMING ORGANIZATION REPORT NUMBER(S)			5 MONITORING ORGANIZATION REPORT NUMBER(S)		
6a. NAME OF PERFORMING ORGANIZATION Naval Postgraduate School		6b OFFICE SYMBOL (If applicable) 62	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School		
6c. ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000			7b ADDRESS (City, State, and ZIP Code) Monterey, California 93943-5000		
8a. NAME OF FUNDING/SPONSORING ORGANIZATION		8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER		
8c. ADDRESS (City, State, and ZIP Code)			10. SOURCE OF FUNDING NUMBERS		
			PROGRAM ELEMENT NO	PROJECT NO	TASK NO
					WORK UNIT ACCESSION NO
11 TITLE (Include Security Classification) ALGORITHMS FOR COMPUTER AIDED DESIGN OF DIGITAL FILTERS					
12 PERSONAL AUTHOR(S) RICH, Thomas H.					
13a TYPE OF REPORT Master's Thesis		13b TIME COVERED FROM _____ TO _____		14 DATE OF REPORT (Year, Month, Day) 1988 December	
15 PAGE COUNT 57					
16 SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.					
17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)		
FIELD	GROUP	SUB-GROUP	polynomial substitution; analog-digital design; digital-digital design		
19 ABSTRACT (Continue on reverse if necessary and identify by block number)					
The algorithms necessary for the computer aided design of digital filters from lowpass prototypes has been developed and compared, then implemented in a computer program to aid in the instruction of digital filter design.					
20 DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ NOT USED			21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED		
22a NAME OF RESPONSIBLE INDIVIDUAL STRUM, Robert D.			22b TELEPHONE (Include Area Code) 08-646-3451		22c OFFICE SYMBOL 62St

DD Form 1473, JUN 86

Previous editions are obsolete

SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-014-6603

UNCLASSIFIED

Approved for public release; distribution is unlimited

ALGORITHMS FOR COMPUTER AIDED
DESIGN OF DIGITAL FILTERS

by

Thomas H. Rich
Captain, United States Marine Corps
BSME, Rensselaer Polytechnic Institute, 1981

Submitted in partial fulfillment of the
requirements for the degree of

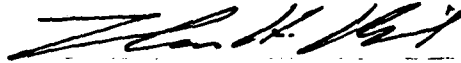
MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

December 1988

Author:



Thomas H. Rich

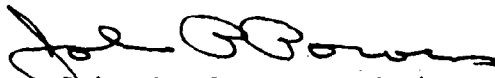
Approved by:



Robert D. Strum, Thesis Advisor



Janine V. England, Second Reader



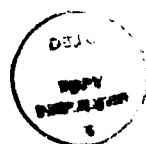
John P. Powers, Chairman,
Department of Electrical and
Computer Engineering



Gordon E. Schacher,
Dean of Science and Engineering

ABSTRACT

The algorithms necessary for the computer aided design of digital filters from lowpass prototypes have been developed and compared, then implemented in a computer program to aid in the instruction of digital filter design.



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

TABLE OF CONTENTS

I.	DEVELOPMENT OF GENERAL ALGORITHMS.....	1
A.	INTRODUCTION.....	1
B.	GENERAL DISCUSSION.....	4
II.	DEVELOPMENT OF SPECIFIC TRANSFORMATIONS.....	13
A.	INTRODUCTION.....	13
B.	BILINEAL TRANSFORMATION.....	13
C.	LOWPASS TO LOWPASS DIGITAL TRANSFORMATION.....	16
D.	LOWPASS TO HIGHPASS DIGITAL TRANSFORMATION.....	19
E.	LOWPASS TO BANDSTOP DIGITAL TRANSFORMATION.....	20
F.	LOWPASS TO BANDPASS DIGITAL TRANSFORMATION.....	23
G.	LOWPASS TO LOWPASS ANALOG TRANSFORMATION.....	24
H.	LOWPASS TO HIGHPASS ANALOG TRANSFORMATION.....	25
I.	LOWPASS TO BANDPASS ANALOG TRANSFORMATION.....	26
J.	LOWPASS TO BANDSTOP ANALOG TRANSFORMATION.....	28
K.	GENERAL OBSERVATIONS.....	29
L.	IMPLEMENTATION.....	32
	LIST OF REFERENCES.....	34
	APPENDIX: DFCADD FORTRAN PROGRAM.....	35
	INITIAL DISTRIBUTION LIST.....	52

I. DEVELOPMENT OF GENERAL ALGORITHMS

A. INTRODUCTION

The procedure for the design of lowpass, highpass, bandpass and bandstop digital filters from lowpass prototypes is well established, and is based on transforming the prototype frequency response to meet desired characteristics. This transformation can be accomplished in either the analog (s) domain or the digital (z) domain (Figure 1).

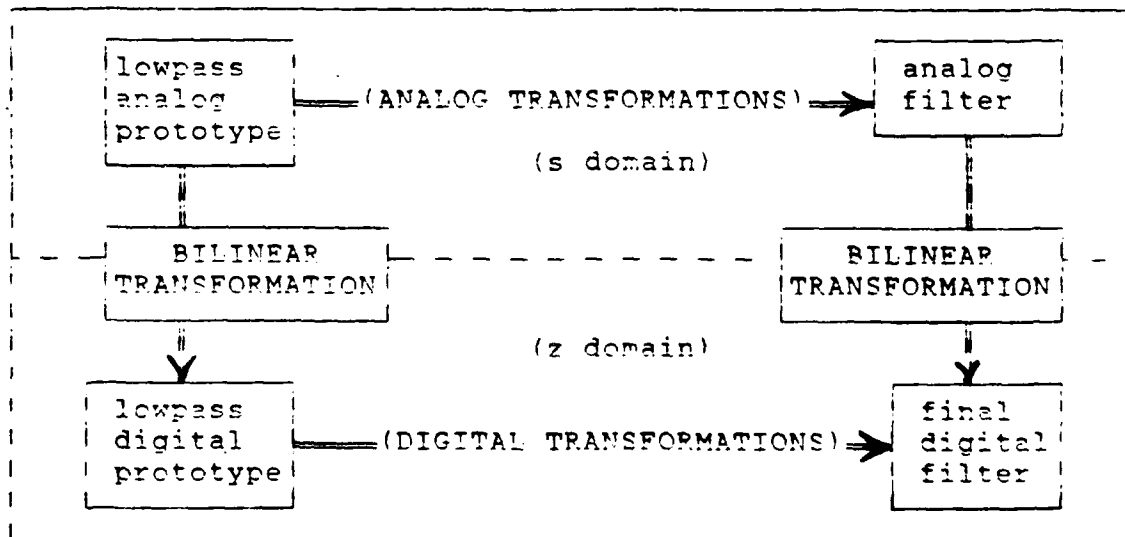


Figure 1. Two methods for digital filter design. Note that each of the horizontal arrows actually represent four different transformations:

- 1) lowpass prototype to lowpass filter
- 2) lowpass prototype to highpass filter
- 3) lowpass prototype to bandpass filter
- 4) lowpass prototype to bandstop filter

Therefore, the two classical design techniques, analog-digital and digital-digital design, are based on doing the frequency response transformations in their respective analog or digital domains. Since the theory for the standard filter types (Butterworth, Chebyshev, and elliptic) is developed in the analog or 's' domain, the two methods for designing digital filters differ basically in the sequence of the design steps. In the analog-digital method the analog prototype is transformed and then digitized, while in the digital-digital method the analog prototype is digitized and then transformed. *Key word: Transformations*

Each of the transformations (arrows in Figure 1) involves substituting a function of 's' or 'z' for the appropriate variable in the filter transfer function. The purpose of this thesis is to develop efficient algorithms to carry out these transformations and then to implement these algorithms in a computer program to aid in the design of digital filters.

Computer-aided design of digital filters offers a significant advantage since it allows the digital filter coefficients to be calculated quickly and accurately. The accuracy of the filter coefficients is an especially significant point since the frequency response of the final filter is very sensitive to small inaccuracies in the calculated coefficients. As an example, compare the frequency response for the filters in Figure 2 and 3. Figure

2 shows the frequency response magnitude for a 1/2-dB ripple Chebyshev bandpass filter, and Figure 3 shows the frequency response when one of the coefficients has been changed by 1/1000th.

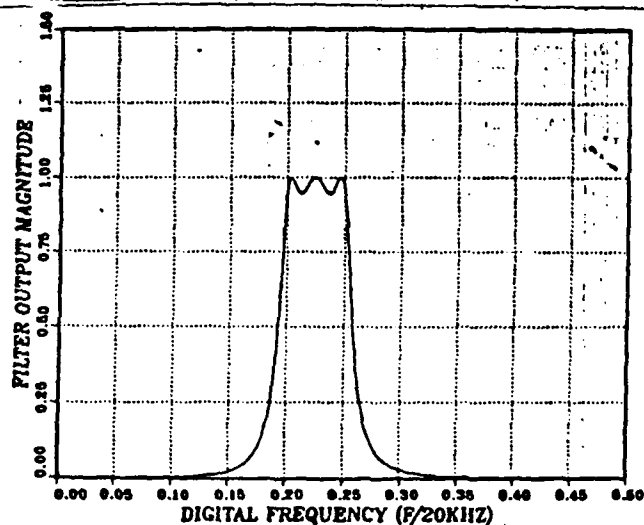


Figure 2. Chebyshev Bandpass Filter

$$H(z) = \frac{.00229(z - 1)^2}{z^4 - .87775z^3 + 2.80076z^2 - 1.5389z + 2.46277z - .67494z + .67529}$$

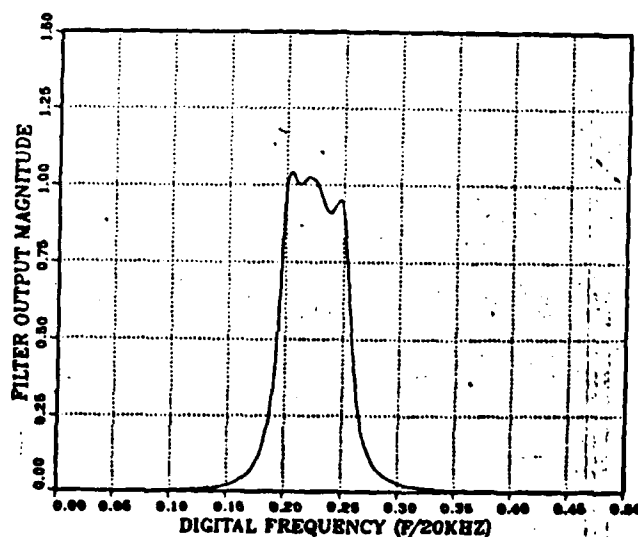


Figure 3. Same Filter With Denominator Coefficient of z changed to $-.67594$

B. GENERAL DISCUSSION

Each of the transformations involves substituting a ratio of two polynomials of s or z into the filter transfer function which is also a ratio of two polynomials in s or z , namely

$$H(y) = \frac{Y(y)}{U(y)} \quad \left| \quad y = \frac{p(x)}{q(x)} \right. \quad (1.1)$$

The variables ' x ' and ' y ' can be either ' s ' or ' z ' depending on the transformation to be performed. Therefore, we have either

$$H(y) = \frac{a_0 + a_1 y + a_2 y^2 + \dots + a_m y^m}{b_0 + b_1 y + b_2 y^2 + \dots + b_n y^n} \quad \left| \quad y = \frac{p(x)}{q(x)} \right. \quad (1.2)$$

or

$$H(x) = \frac{a_0 + a_1 \frac{p(x)}{q(x)} + a_2 \frac{(p(x))^2}{(q(x))^2} + \dots + a_m \frac{(p(x))^m}{(q(x))^m}}{b_0 + b_1 \frac{p(x)}{q(x)} + b_1 \frac{(p(x))^2}{(q(x))^2} + \dots + b_n \frac{(p(x))^n}{(q(x))^n}} \quad (1.3)$$

In general, the order of the numerator ' m ', is less than the order of the denominator ' n ' (except for the transformations in the digital domain where the result of the bilinear transformation is to make $n=m$). We can generalize the

results, however, by assuming that the order of the numerator polynomial $Y(x)$ is equal to the order of the denominator polynomial $U(x)$ and making the additional coefficients $a_{n+1}, a_{n+2}, \dots, a_n$ equal to zero. Then multiplying both the numerator and denominator of the above equation by $(g(x))^n$ gives

$$H(x) = \frac{a_0 (\sigma(x))^n + a_1 p(x) (g(x))^{n-1} + \dots + a_n (p(x))^n}{b_0 (g(x))^n + b_1 p(x) (g(x))^{n-1} + \dots + b_n (p(x))^n} \quad (1.4)$$

or, in a different form,

$$H(x) = \frac{\sum_{i=0}^{i=n} a_i p(x)^i g(x)^{n-i}}{\sum_{i=0}^{i=n} b_i p(x)^i g(x)^{n-i}} \quad (1.5)$$

Now with the numerator and denominator of the same form, we can use the same transformation for both the numerator and denominator polynomials of the filter transfer function, namely,

$$a'(x) = a_0' + a_1'x + a_2'x^2 + \dots + a_n'x^n = \sum_{i=0}^{i=n} a_i p(x)^i g(x)^{n-i} \quad (1.6)$$

In Equation 1.6, the a_i are the coefficients of the old numerator or denominator polynomials and a_i' are the new, transformed numerator or denominator polynomial coefficients. Since the new coefficients a_i' are a linear combination of

the old coefficients a_i and the coefficients of $p(x)$ and $q(x)$, this suggests the use of a matrix transformation;

$$a' = aA(n) \quad (1.7)$$

where

a is the coefficient row vector for the old numerator or denominator polynomial of length $n+1$.

$$a = [a_n \ a_{n-1} \ a_{n-2} \ \dots \ a_0]$$

a' is the coefficient row vector for the new numerator or denominator polynomial of length $n'+1$.

$$a' = [a'_n \ a'_{n-1} \ a'_{n-2} \ \dots \ a'_0]$$

$A(n)$ is the transformation matrix with dimensions $(n+1) \times (n'+1)$. The elements of $A(n)$ depend upon the coefficients of $p(x)$ and $q(x)$. Note that if the rows and columns of $A(n)$ are numbered 0 to n' , then

$$a'_j = \sum_{i=0}^{n'} a_i A(n)_{ij} \quad (1.8)$$

(where $A(n)_{ij}$ denotes the element in row i , column j of matrix $A(n)$.)

n is the order of the transfer function before the transformation

n' is the order of the transfer function after the transformation. Note that n' equals n times the highest order of $p(x)$ and $q(x)$.

For example, when $n=1$ and $n'=2$, $p(x)$ and $q(x)$ are second order and Equation 1.7 becomes,

$$\begin{bmatrix} a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} A(2)_{00} & A(2)_{01} & A(2)_{02} \\ A(2)_{10} & A(2)_{11} & A(2)_{12} \\ A(2)_{20} & A(2)_{21} & A(2)_{22} \end{bmatrix} = \begin{bmatrix} a'_2 & a'_1 & a'_0 \end{bmatrix} \quad (1.9)$$

In the case of the lowpass digital, highpass digital, and bilinear transformations, the two functions $p(x)$ and $q(x)$ are first order. Professor P.H. Moose of the Naval Postgraduate School of Monterey has developed the matrix $A(n)$ for these two cases by expanding Equation 1.6 using the binomial theorem (Reference 1). This method cannot be used for transformations involving higher order polynomials such as the bandpass and bandstop transformations and it is difficult to develop because intricate manipulation of the binomial coefficients are involved. The binomial expansion approach, therefore, was not pursued.

A general method has been found that develops the required transformation matrices in an iterative manner. In other words, the matrix used to transform a numerator or denominator filter polynomial of order n is developed from the matrix for the polynomial of order $n-1$. This approach is easy to implement in a computer program and can be used for any of the polynomial substitutions needed. The next chapter is concerned with developing the transformation matrices for each specific transformation. However, as an introduction to the method used, consider the generic case where both $p(x)$ and $q(x)$ are second-order polynomials. Since all of the analog and digital transformations involve substitutions with polynomials of second order or less, this second-order generic case will be all that is needed to develop the specific transformations later.

With $p(x)=ax^2 + bx + c$, $q(x) = dx^2 + ex + f$, and substituting into Equation 1.6 for the first order case of $n=1$ (remembering that with $p(x)$ and $q(x)$ of second order, the resulting polynomial will be second order), we get

$$a'(x) = a_0' + a_1'x + a_2'x^2 = a_0(dx^2 + ex + f) + a_1(ax^2 + bx + c) \quad (1.10)$$

Using Equation 1.8 or 1.9 it is easy to pick out the matrix coefficients $A(1)_{ij}$ ($i=0,1,2$; $j=0,1,2$) for the first order generic case. The coefficients are,

$$A(1) = \begin{bmatrix} f & e & d \\ c & b & a \end{bmatrix} \quad (1.11)$$

Now with $n=2$ ($n'=4$) substituting into Equation 1.6 gives,

$$\begin{aligned} a'(x) &= a_0' + a_1'x + a_2'x^2 + a_3'x^3 + a_4'x^4 \\ &= a_0(dx^2 + ex + f)^2 + a_1(ax^2 + bx + c)(dx^2 + ex + f) + a_2(ax^2 + bx + c)^2 \end{aligned} \quad (1.12)$$

and the matrix $A(2)$ is;

$$A(2) = \begin{bmatrix} f^2 & 2ef & 2df+e^2 & 2de & d^2 \\ fc & ec+bf & cd+fa+eb & db+ea & da \\ c^2 & 2bc & 2ac+b^2 & 2ab & a \end{bmatrix} \quad (1.13)$$

To see how this matrix is derived from the previous one, consider the first row of the matrices in Equations 1.11 and 1.13. For both matrices, this row is multiplied by the a_0 coefficient in the old polynomial. For $n=1$ we have $a_0 p(x)$

and we pick out the coefficients of the powers of x (x') to get the matrix elements $A(n)_{ij}$. For $n=2$, we have $a_0(p(x))^2$ and we do the same thing only now we have the additional factor of $p(x)$. The matrix elements can be calculated by hand in this fashion but it requires a lot of algebra. A simpler method is to use the matrix elements of the previous matrix and get the appropriate elements by convolving coefficients. For example, consider the sequence of elements in row 0 (the top row) of $A(1)$, namely

$$f \quad e \quad d \quad (1.14)$$

To get the elements for row 0 of $A(2)$, we need to multiply (convolve coefficients) by another $p(x)$. Therefore, with rows numbered 0 to n , and columns number 0 to n' , we have the following.

For row 0 column 0, the appropriate matrix element is

$$A(2)_{00} = \begin{matrix} & & f \\ d & e & \\ & f & e & d \end{matrix} = f^2 \quad (1.15)$$

For row 0 column 1, the appropriate matrix element is

$$A(2)_{01} = \begin{matrix} & d & e & f \\ & & f & e & d \end{matrix} = 2fe \quad (1.16)$$

For row 0 column 2, the appropriate matrix element is

$$A(2)_{02} = \begin{matrix} & & d & e & f \\ & d & e & f \\ & f & e & d \end{matrix} = 2fd + e^2 \quad (1.17)$$

For row 0 column 3, the appropriate matrix element is

$$A(2)_{03} = \begin{matrix} & d & e & f \\ f & e & d & \end{matrix} = 2de \quad (1.18)$$

For row 0 column 4, the appropriate matrix element is

$$A(2)_{04} = \begin{matrix} & d & e & f \\ f & e & d & \end{matrix} = d^2 \quad (1.19)$$

This convolution of the coefficients of the old matrix with the coefficients of $p(x)$ can be carried out for every row of the old matrix to obtain all the rows of the new matrix except the last row. In the case of the last row, we are multiplying by an additional $g(x)$ rather than an additional $p(x)$ and so must obtain the last row of the new matrix by convolving the coefficients of the last row of the old matrix with the coefficients of $g(x)$.

In summary we have,

(i) for row 0 to row $n-1$:

$$\text{row } i \text{ of } A(n) = \text{row } i \text{ of } A(n-1) * \text{coefficients of } p(x) \quad (1.20)$$

(ii) for row n :

$$\text{row } n \text{ of } A(n) = \text{row } n-1 \text{ of } A(n-1) * \text{coefficients of } g(x) \quad (1.21)$$

where $*$ indicates linear convolution of the coefficients.

To see how this method of convolving coefficients works, consider multiplying two polynomials:

$$f_1(x) = f_2(x)f_3(x) \quad (1.22)$$

Since 'x' is the independent variable. no matter what letter we use. the polynomials will remain the same. Substituting 'z⁻¹' for 'x' in the above equation gives us.

$$f_1(z^{-1}) = f_2(z^{-1})f_3(z^{-1}) \quad (1.23)$$

and we now have the familiar form where we can either multiply the two functions of 'z' or convolve the sequences found by taking the inverse z-transform. For f₂(z⁻¹) and f₃(z⁻¹) we find

$$\begin{aligned} f_2(z^{-1}) &= a_0 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_n z^{-n} \\ f_3(z^{-1}) &= b_0 + b_1 z^{-1} + b_2 z^{-2} + \dots + b_m z^{-m} \end{aligned} \quad (1.24)$$

It is now clear that we can find the coefficients of the product f₁(x) by convolving the sequence {a_n} with the sequence {b_m} to produce the sequence {c_q} where.

$$\begin{aligned} \{a_n\} &= \{a_0, a_1, a_2, a_3, \dots, a_n\} \\ \{b_m\} &= \{b_0, b_1, b_2, b_3, \dots, b_m\} \\ \text{and with } q &= m + n \end{aligned} \quad (1.25)$$

$$\{c_q\} = \{c_0, c_1, c_2, c_3, \dots, c_q\}$$

Then by taking the z-transform of the sequence {c_q} and substituting 'x' for 'z⁻¹' we get.

$$f_1(x) = c_0 + c_1 x + \dots + c_q x^q \quad (1.26)$$

We now have a simple straight-forward method of developing the necessary transformation matrices for digital filter design by picking out the first-order transformation matrix using Equations 1.6 and 1.11, and then using the convolution of coefficients method iteratively on the rows of each successive transformation matrix with Equations 1.20 and 1.21 until the appropriate matrix $A(n)$ has been generated. The next chapter will develop the transformation matrices for each of the following transformations:

- bilinear transformation
- digital lowpass prototype to digital lowpass filter
- digital lowpass prototype to digital highpass filter
- digital lowpass prototype to digital bandpass filter
- digital lowpass prototype to digital bandstop filter
- analog lowpass prototype to analog lowpass filter
- analog lowpass prototype to analog highpass filter
- analog lowpass prototype to analog bandpass filter
- analog lowpass prototype to analog bandstop filter.

We will see that some of these transformations are similar and consequently, we will not have to develop separate transformation matrices for each case.

II. DEVELOPMENT OF SPECIFIC TRANSFORMATIONS

A. INTRODUCTION

The individual transformations are discussed in the following sections. The specific form of the transformations used in each case are from Reference 2. Since the final result will be to develop a computer program to assist in the calculation of digital filter coefficients, confining the specific transformations to the form in Reference 2, will allow the computer program to be used in conjunction with Reference 2 as an aid in designing digital filters.

B. BILINEAR TRANSFORMATION

The bilinear transformation may be used to transform an analog transfer function, $H(s)$, to a digital transfer function $H(z)$. The transformation from $H(s)$ to $H(z)$ is described by

$$H(z) = H(s) \Big|_{s = \frac{z-1}{z+1}} \quad (2.1)$$

Referring to Equation 1.1, $p(x)=z-1$ and $q(x)=z+1$. Since both of the transformation polynomials $p(x)$ and $q(x)$ are first order, the bilinear transformation will not change the order of the transfer function. Therefore, for an analog transfer function of order 'n', the transformation matrix $A(n)$ will be

(n+1) by (n+1). From Equation 1.6, the numerator or denominator polynomial of the transfer function $H(z)$ is given by

$$a'(z) = a_0' + a_1'z + \dots + a_n'z^n = \sum_{i=0}^{i=n} a_i (z-1)^i (z+1)^{n-i} \quad (2.2)$$

where the unprimed coefficients, a_i , come from the analog transfer function's numerator or denominator polynomial. In matrix form, with $n=1$ and referring to Equation 1.11, the first order matrix $A(1)$ for the bilinear transformation is

$$A(1) = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix} \quad (2.3)$$

Using the convolution of coefficients method developed in Chapter 1, the transformation matrix $A(n+1)$ can be developed from the matrix $A(n)$. The results for the bilinear transformation matrices $A(1)$ through $A(5)$ are given in Table 1.

TABLE 1
FIRST-ORDER TO FIFTH-ORDER BILINEAR TRANSFORMATION MATRICES

$$A(1) = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}$$

$$A(2) = \begin{bmatrix} 1 & 2 & 1 \\ -1 & 0 & 1 \\ 1 & -2 & 1 \end{bmatrix}$$

$$A(3) = \begin{bmatrix} 1 & 3 & 3 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ -1 & 3 & -3 & 1 \end{bmatrix}$$

$$A(4) = \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ -1 & -2 & 0 & 2 & 1 \\ 1 & 0 & -2 & 0 & 1 \\ -1 & 2 & 0 & -2 & 1 \\ 1 & -4 & 6 & -4 & 1 \end{bmatrix}$$

$$A(5) = \begin{bmatrix} 1 & 5 & 10 & 10 & 5 & 1 \\ -1 & -3 & -2 & 2 & 3 & 1 \\ 1 & 1 & -2 & -2 & 1 & 1 \\ -1 & 1 & 2 & -2 & -1 & 1 \\ 1 & -3 & 2 & 2 & -3 & 1 \\ -1 & 5 & -10 & 10 & -5 & 1 \end{bmatrix}$$

Referring to Table 1 and keeping in mind the convolution of coefficients method that produced the matrices, the following relations are apparent and will be useful in computer implementation.

$$A(n)_{0,j} \text{ (the first row)} = \binom{n}{j} \quad \text{for } j=0,1,\dots,n \quad (2.4)$$

where $\binom{n}{j}$ indicates the binomial coefficient

$$A(n)_{i,0} \text{ (the first column)} = (-1)^i \quad \text{for } i=0,1,\dots,n \quad (2.5)$$

$$A(n)_{i,n} \text{ (the last column)} = 1 \quad \text{for } i=0,1,\dots,n \quad (2.6)$$

$$A(n)_{i,n-1-j} = (-1)^{n-j} A(n)_{i,j} \quad \text{for } i=0,1,\dots, \text{integer}(n/2)+1 \quad (2.7)$$

$$A(n)_{i,j} = A(n-1)_{i,j} + A(n-1)_{i,j-1} \quad \text{for } i=0,1,\dots,n-1 \quad (2.8)$$

$$j=1,2,\dots,n-1$$

Notice that the values of the matrix elements for the bilinear transformation are whole numbers. This is important from a numerical accuracy standpoint since whole numbers can be expressed precisely in floating point form. This means that the bilinear transformation will cause a loss of accuracy of the original coefficients only because of the $n+1$ multiplications and additions used to calculate each new digital coefficient.

C. LOWPASS TO LOWPASS DIGITAL TRANSFORMATION

The lowpass to lowpass digital transformation is used to transform a lowpass prototype digital filter $H(z)$ into a lowpass digital filter $H(z)$. The polynomial substitution

used is

$$H(z) = H_p(z) \left| z = \frac{z - \alpha}{1 - \alpha z} \right. \quad (2.9)$$

where the subscript p indicates the prototype digital filter transfer function. The design constant α is determined [References 2 and 3] by

$$\alpha = \frac{\sin(\theta_c/2 - \theta_c'/2)}{\sin(\theta_c/2 - \theta_c''/2)} \quad (2.10)$$

with θ_c = the prototype critical frequency (usually $\pi/2$) and θ_c' = the desired critical frequency of the lowpass filter. Again, referring to Equation 1.1 with $p(x) = z - \alpha$ and $g(z) = 1 - \alpha z$, the lowpass digital transformation matrix will not change the order of the transfer function and the transformation matrix $A(n)$ will be $(n+1)$ by $(n+1)$. Using Equations 1.6 and 1.11 as before, the first order transformation matrix $A(1)$ for the lowpass digital transformation is:

$$A(1) = \begin{bmatrix} 1 & -\alpha \\ -\alpha & 1 \end{bmatrix} \quad (2.11)$$

Once again using the convolution of coefficients method, the higher-order transformation matrixes $A(n)$ are developed from the lower-order matrix $A(n-1)$. The results are shown in Table 2 for the first-order to fifth-order transformation matrices.

TABLE 2
FIRST-ORDER TO FIFTH-ORDER LOWPASS DIGITAL
TRANSFORMATION MATRICES

$$A(1) = \begin{bmatrix} 1 & -\alpha \\ -\alpha & 1 \end{bmatrix}$$

$$A(2) = \begin{bmatrix} 1 & -2\alpha & \alpha^2 \\ -\alpha & (1+\alpha^2) & -\alpha \\ \alpha^2 & -2\alpha & 1 \end{bmatrix}$$

$$A(3) = \begin{bmatrix} 1 & -3\alpha & 3\alpha^2 & -\alpha^3 \\ -\alpha & (1+2\alpha^2) & (-2\alpha-\alpha^3) & \alpha^2 \\ \alpha^2 & (-2\alpha-\alpha^3) & (1+2\alpha^2) & -\alpha \\ -\alpha^3 & 3\alpha^2 & -3\alpha & 1 \end{bmatrix}$$

$$A(4) = \begin{bmatrix} 1 & -4\alpha & 6\alpha^2 & -4\alpha^3 & \alpha^4 \\ -\alpha & (1+3\alpha^2) & (-3\alpha-3\alpha^3) & (3\alpha^2+\alpha^4) & -\alpha^3 \\ \alpha^2 & (-2\alpha-2\alpha^3) & (1+4\alpha^2+\alpha^4) & (-2\alpha-2\alpha^3) & \alpha^2 \\ -\alpha^3 & (3\alpha^2+\alpha^4) & (-3\alpha-3\alpha^3) & (1+3\alpha^2) & -\alpha \\ \alpha^4 & -4\alpha^3 & 6\alpha^2 & -4\alpha & 1 \end{bmatrix}$$

$$A(5) = \begin{bmatrix} 1 & -5\alpha & 10\alpha^2 & -10\alpha^3 & 5\alpha^4 & -\alpha^5 \\ -\alpha & (1+3\alpha^2) & (-4\alpha-6\alpha^3) & (6\alpha^2+\alpha^4) & (-4\alpha^3-\alpha^5) & \alpha^4 \\ \alpha^2 & (-2\alpha-3\alpha^3) & (1+6\alpha^2+3\alpha^4) & (-3\alpha-6\alpha^3-\alpha^5) & (3\alpha^2+2\alpha^4) & -\alpha^3 \\ -\alpha^3 & (3\alpha^2+2\alpha^4) & (-3\alpha-6\alpha^2-\alpha^5) & (1+6\alpha^2+3\alpha^4) & (-2\alpha-3\alpha^3) & \alpha^2 \\ \alpha^4 & (-4\alpha^3-\alpha^5) & (6\alpha^2+4\alpha^4) & (-4\alpha-6\alpha^3) & (1+4\alpha^2) & -\alpha \\ -\alpha^5 & 5\alpha^4 & -10\alpha^3 & 10\alpha^2 & -5\alpha & 1 \end{bmatrix}$$

Referring to Table 2, the following relations are observed and will be used in the computer implementation.

$$A(n)_{0,j} \text{ (the first row)} = \binom{n}{j} (-\alpha)^j \text{ for } j=0,1,\dots,n \quad (2.12)$$

$$A(n)_{i,0} \text{ (the first column)} = (-\alpha)^i \text{ for } i=0,1,\dots,n \quad (2.13)$$

$$A(n)_{i,j} = A(n)_{(n-i),(n-j)} \text{ for } i,j=0,1,\dots,n \quad (2.14)$$

$$A(n)_{i,j} = A(n-1)_{(i-1),(j-1)} + (-\alpha)A(n-1)_{(i-1),j} \quad (2.15)$$

for $i=1,2,\dots,n-1$
 $j=1,2,\dots,n$

Note that the elements of the lowpass digital transformation matrices are not whole numbers as in the bilinear transformation case, but are polynomials of the design constant α . Numerical error is introduced in this case by both the $n+1$ additions or multiplications, and the error in calculating the matrix elements that multiply the original coefficients.

D. LOWPASS TO HIGHPASS DIGITAL TRANSFORMATION

The lowpass to highpass digital transformation is used to transform a lowpass prototype to a highpass filter. The polynomial substitution for this transformation is:

$$H(z) = H_b(z) \Big|_{z = -\frac{z-\alpha}{1-\alpha z}} \quad (2.16)$$

Since this transformation is the same as that for the lowpass to lowpass digital transformation except for the minus sign.

we can make the substitution of '-z' for 'z' in the original prototype transfer function and then use the same transformation matrices as the lowpass to lowpass case. For the lowpass to highpass transformation, the design constant α is (References 2 and 3)

$$\alpha = \frac{\cos(\theta_c/2 - \theta_c'/2)}{\cos(\theta_c/2 + \theta_c'/2)} \quad (2.17)$$

with θ_c - the prototype critical frequency (usually $\pi/2$) and θ_c' - the desired critical frequency of the highpass filter.

E. LOWPASS TO BANDSTOP DIGITAL TRANSFORMATION

The polynomial substitution which transforms a digital lowpass prototype to a bandstop digital filter is (References 2 and 3):

$$H(z) = H_p(z) \left| \begin{array}{l} z^2 - \frac{2\alpha}{1+k} z + \frac{1-k}{1+k} \\ z = \frac{2\alpha}{1-k} z + \frac{1+k}{1-k} z^2 \end{array} \right| \quad (2.18)$$

with

$$\alpha = \frac{\cos(\theta_u'/2 + \theta_l'/2)}{\cos(\theta_u'/2 - \theta_l'/2)} \quad (2.19)$$

and

$$K = \tan(\theta_c/2) \tan(\theta_u'/2 - \theta_l'/2) \quad (2.20)$$

with once again, θ_c = the prototype critical frequency (usually $\pi/2$), θ_l' = the desired upper critical digital

frequency, and θ_1' = the desired low critical digital frequency.

By letting $b = 2\alpha/(k+1)$ and $c = (1-k)/(1+k)$, Equation 2.18 can be simplified to

$$H(z) = H_p(z) \Big|_{z = \frac{z^2 - bz + c}{1 - bz + cz^2}} \quad (2.21)$$

Again, using Equations 1.6 and 1.11 the first-order transformation matrix is

$$A(1) = \begin{bmatrix} 1 & -b & c \\ c & -b & 1 \end{bmatrix} \quad (2.22)$$

Note that since the substitution polynomials are second order, this transformation will double the order of the original prototype filter. Therefore, the transformation matrix $A(n)$ will be $(n+1)$ by $(2n+1)$. The higher order transformation matrices are again developed recursively and the results for the first-order to third-order transformations are given in Table 3.

TABLE 3
FIRST-ORDER TO THIRD-ORDER BANDSTOP
DIGITAL TRANSFORMATION MATRICES

$$A(1) = \begin{bmatrix} 1 & -b & c \\ c & -b & 1 \end{bmatrix}$$

$$A(2) = \begin{bmatrix} 1 & -2b & (b^2+2c) & -2bc & c^2 \\ c & (-b-bc) & (1+b^2+c^2) & (-b-bc) & c \\ c^2 & -2bc & (b^2+2c) & -2c & 1 \end{bmatrix}$$

$$A(3) = \begin{bmatrix} 1 & -3c & (3b^2+3c) & (-b^3-6bc) & \dots \\ c & (-2bc-b) & (1+2b^2+b_c+2c) & (-b^3-2b-2bc-2bc^2) & \dots \\ c^2 & (-2bc-bc^2) & (c^3+2b^2c+2c+b^2) & \dots & \\ c^3 & -3c^2b & (3b^2c+3c^2) & \dots & \end{bmatrix}$$

(Redundant entries not shown for A(3) due to space limitations. Entries not shown can be found using Equation 2.25)

Notice the increased amount of calculation needed to determine the matrix elements for the bandstop case. Each matrix element is a function of 'b' and 'c' which are themselves functions of the design constants α and k . Table 3 only gives the transformation matrices for first- to third-order. However, to illustrate the amount of calculation required, the matrix element $A(5)_{2,5}$ is

$$A(5)_{2,5} = (-12bc^2 - 6bc^3 - 8b^3c - 6bc - 6b^3 - 3b - 3bc^4 - 6b^3c^2 - b^5) \quad (2.23)$$

Since each coefficient in the final filter is found by multiplying $n+1$ matrix elements by the $n+1$ original

coefficients and then adding the $n+1$ products, the accuracy of the design constants and the original coefficients can be critical in the calculation of the final filter coefficients.

The symmetry relations for the bandstop transformation are:

$$A(n)_{i0} \text{ (the first column)} = c^i \text{ for } i=0,1,\dots,n \quad (2.24)$$

$$A(n)_{ij} = A(n)_{(n-i)(2n-j)} \text{ for } i=0,1,\dots,n \quad (2.25)$$

$$j=0,1,\dots,2n$$

$$A(n)_{ij} = A(n-1)_{ij} - bA(n-1)_{i(i-1)} + cA(n-1)_{i(i-2)} \quad (2.26)$$

$$\text{for } i=0,1,\dots,n-1$$

$$j=0,1,\dots,2n$$

F. LOWPASS TO BANDPASS DIGITAL TRANSFORMATION

Again, the polynomial substitution which transforms the lowpass prototype to a bandpass filter is

$$H(z) = H_p(z) \quad \left| \quad \begin{aligned} z^2 - \frac{2\alpha k}{k+1} z + \frac{k-1}{k+1} \\ z = - \frac{1 - \frac{2\alpha k}{k+1} z + \frac{k-1}{k+1} z^2}{\quad} \end{aligned} \right. \quad (2.27)$$

where in this case $k = \tan(\theta/2) \cot(\theta_0'/2 - \theta_1'/2)$ and α is found from the same expression as the bandstop case. With the simplification of $b=2\alpha k/(k+1)$ and $c=(k-1)/(k+1)$, Equation 2.27 becomes

$$H(z) = H_p(z) \Big|_{z = -\frac{z^2 - bz + c}{1 - bz + cz^2}} \quad (2.28)$$

Since this is the same form as the bandstop transformation, we can use the matrices previously developed if '-z' is first substituted for 'z' as was done in the lowpass and highpass situation.

G. LOWPASS TO LOWPASS ANALOG TRANSFORMATION

The lowpass to lowpass analog transformation is used to transform an analog lowpass prototype to a lowpass analog filter. The polynomial substitution is a simple one, namely

$$H(s) = H_p(s) \Big|_{s = s/w_c} \quad (2.29)$$

where w_c is the prewarped critical frequency ($w_c = \tan(\theta_c/2)$) and θ_c is the desired digital critical frequency ($\theta_c = 2(\pi)f_c/f_s$). This transformation matrix is found to be

$$A(n) = \begin{bmatrix} 1 & 0 & 0 & . & . & . & 0 \\ 0 & 1/w_c & 0 & . & . & . & 0 \\ 0 & 0 & 1/w_c^2 & . & . & . & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & 0 & 0 & . & . & . & 1/w_c^n \end{bmatrix}$$

or $A(n) = \text{diag}(1, 1/w_c, 1/w_c^2, ., ., 1/w_c^n)$ throughout. (2.30)

Note that in this case, each new filter coefficient depends only on multiplying the old coefficient of s^1 by $(w_c)^{-1}$. Therefore, any loss of accuracy due to the transformation is due to this one multiplication and the error introduced in calculation of the matrix element $(w_c)^{-1}$.

H. LOWPASS TO HIGHPASS ANALOG TRANSFORMATION

The polynomial substitution for the lowpass to highpass transformation is

$$H(s) = H_p(s) \Big|_{s = w_c / s} \quad (2.31)$$

Again using Equations 1.6 and 1.11, the first-order transformation matrix is

$$A(1) = \begin{bmatrix} 0 & 1 \\ w_c & 0 \end{bmatrix} \quad (2.32)$$

The method of convolution of coefficients can be used to get the higher-order transformation matrices. However, notice that Equation 2.32 is the same matrix as the lowpass to lowpass transformation with the reciprocal taken of each non zero element, and then each row reversed. This can be shown to be the case in general, so that the lowpass to highpass transformation matrix is given by

$$A(n) = \begin{bmatrix} 0 & 0 & . & . & . & 0 & 1 \\ 0 & 0 & . & . & . & w_c & 0 \\ . & . & . & . & . & . & . \\ . & . & . & . & . & . & . \\ 0 & w_c^{n-1} & . & . & . & 0 & 0 \\ w_c^n & 0 & . & . & . & 0 & 0 \end{bmatrix} \quad (2.33)$$

Again, each new filter coefficient depends only on one multiplication although the calculation of the matrix element itself will require calculations involving raising the design constant, w_c , to the $(n-1)^{th}$ power where i is the power of s in the final filter transfer function.

I. LOWPASS TO BANDPASS ANALOG TRANSFORMATION

The polynomial substitution for transforming a lowpass analog prototype to a bandpass analog filter is

$$H(s) = H_p(s) \left| \begin{array}{l} s^2 + w_o^2 \\ s = \frac{Bs}{Bs} \end{array} \right. \quad (2.34)$$

where w_o is defined as the geometric mean of the passband;

$w_o = (w_u w_l)^{1/2}$, and B is the width of the passband, $B = w_u - w_l$.

(w_u and w_l are the upper and lower critical frequencies respectively). Again using Equations 1.6 and 1.11 and picking out the appropriate matrix elements, the first-order transformation matrix is given by

$$A(1) = \begin{bmatrix} 0 & B & 0 \\ w_0^2 & 0 & 1 \end{bmatrix} \quad (2.35)$$

The convolution of coefficients technique is again used to generate the higher order transformation matrices with the results for the first-order to third-order matrices listed in Table 4. As with the bandpass digital transformation, the analog bandpass transformation will double the order of the original transfer function since the substitution polynomial is of second order.

TABLE 4
FIRST-ORDER TO THIRD-ORDER BANDPASS
ANALOG TRANSFORMATION MATRICES

$$A(1) = \begin{bmatrix} 0 & B & 0 \\ w_0^2 & 0 & 1 \end{bmatrix}$$

$$A(2) = \begin{bmatrix} 0 & 0 & B^2 & 0 & 0 \\ 0 & Bw_0^2 & 0 & B & 0 \\ w_0^4 & 0 & 2w_0^2 & 0 & 1 \end{bmatrix}$$

$$A(3) = \begin{bmatrix} 0 & 0 & 0 & B^3 & 0 & 0 & 0 \\ 0 & 0 & B^2w_0^2 & 0 & B^2 & 0 & 0 \\ 0 & Bw_0^4 & 0 & 2Bw_0^2 & 0 & B & 0 \\ w_0^6 & 0 & 3w_0^4 & 0 & 3w_0^2 & 0 & 1 \end{bmatrix}$$

Note that each matrix is slightly over half empty and that the nonzero elements are simple functions of the design constants w_0 and B .

J. LOWPASS TO BANDSTOP ANALOG TRANSFORMATION

For the analog bandstop transformation, the polynomial substitution is the reciprocal of the bandpass case:

$$H(s) = H_p(s) \quad \left| \quad s = \frac{Bs}{s^2 + \omega_o^2} \right. \quad (2.36)$$

Using the same method as before, the first-order transformation matrix is found to be:

$$A(1) = \begin{bmatrix} \omega_o^2 & 0 & 1 \\ 0 & B & 0 \end{bmatrix} \quad (2.37)$$

Note that this is the same matrix as the bandpass case but with the rows in reverse order. Using the convolution of coefficients technique the higher-order matrices are developed and the results for the first-order to third-order transformation matrices are shown in Table 5. Notice that the reversal of the order of the rows compared with the bandpass case is true in general. Therefore, once again the matrices are over half empty and nonzero elements are simple functions of B and ω_o .

TABLE 5
FIRST-ORDER TO THIRD-ORDER BANDSTOP
ANALOG TRANSFORMATION MATRICES

$$A(1) = \begin{bmatrix} w_c^2 & 0 & 1 \\ 0 & B & 0 \end{bmatrix}$$

$$A(2) = \begin{bmatrix} w_c^4 & 0 & 2w_c^2 & 0 & 1 \\ 0 & Bw_c^4 & 0 & B & 0 \\ 0 & 0 & B^2 & 0 & 0 \end{bmatrix}$$

$$A(3) = \begin{bmatrix} w_c^6 & 0 & 2w_c^4 & 0 & 3w_c^2 & 0 & 1 \\ 0 & Bw_c^4 & 0 & 2Bw_c^2 & 0 & B & 0 \\ 0 & 0 & B^2 w_c^2 & 0 & B^2 & 0 & 0 \\ 0 & 0 & 0 & B^3 & 0 & 0 & 0 \end{bmatrix}$$

K. GENERAL OBSERVATIONS

Now that the various transformation matrices for digital filter design have been developed, it is possible to compare the two methods of digital filter design mentioned earlier and depicted in Figure 1. For this comparison, consider an analog lowpass prototype filter transfer function with numerator of order m , and denominator of order n (m less than or equal to n).

First consider the digital to digital direct design method. Transforming the analog prototype to a digital prototype using the bilinear transformation will require $m+1$

multiplications and additions per digital prototype coefficient for the numerator polynomial, and $n+1$ multiplications and additions per digital prototype coefficient for the denominator polynomial. Then, each of the digital to digital transformations will require $2(n+1)$ multiplications and additions since a result of the bilinear transformation is to make $n = m$. Therefore, going from an analog prototype to a final digital filter will require $(3n+m+4)$ multiplications and additions. If lowpass digital prototypes are developed prior to the design process, this can be reduced to $2(n+1)$ since then only the digital to digital transformations will be used in designing the final filter.

Now consider the analog design method for the same case of an m^{th} -order numerator and n^{th} -order denominator lowpass prototype transfer function. For the lowpass and highpass case each new coefficient will require only 1 multiplication and addition. For the bandpass and bandstop transformations a simple expression for the number of calculations per coefficient is difficult to find, however referring to Tables 4 and 5, the number of multiplications and additions per coefficient is always less than half the order of the polynomial to be transformed, on the average. This gives approximately $(n+m)/2$ for the bandpass to bandstop analog transformations. Finally the analog filter must be converted to a digital filter using the bilinear transformation which

will require $(n+m+2)$ calculations per coefficient for the highpass and lowpass case and $2(n+m+1)$ for the bandpass and bandstop case. The total number of calculations in using the analog design method then is $(n+m+4)$ for the high/low pass and $(2.5(n+m)+2)$ for the bandpass/stop. These results are summarized in Table 6.

TABLE 6
NUMBER OF MULTIPLICATIONS AND ADDITIONS REQUIRED
PER COEFFICIENT TO FIND THE DIGITAL FILTER
COEFFICIENTS FROM A LOWPASS ANALOG PROTOTYPE
(WITH AN M^{th} ORDER NUMERATOR AND N^{th} ORDER DENOMINATOR)

<u>Analog Design</u>		
	high/low pass	bandpass/stop
analog transformations	2	$.5(n+m)$
bilinear transformation	$n+m+2$	$2(n+m)+2$
Total	$n+m+4$	$2.5(n+m)+2$

<u>Digital Design</u>		
	high/low pass	bandpass/stop
bilinear transformation	$n+m+2$	$n+m+2m$
digital transformations	$2(n+1)$	$2(n+1)$
Total	$3(n+1)+m+1$	$3(n+1)+m+1$

Referring to Table 6, the advantage of one method over the other depends upon what type of filter is being designed and the order of the prototype transfer functions numerator and denominator. Consider designing a fifth-order low/highpass Chebyshev or Butterworth filter. In this case, the order of the numerator polynomial is zero and the analog method will have a slight advantage of 9 multiplications and additions versus 12 for the digital design with precalculated prototypes (and 19 for digital design without precalculated prototypes). However, if a bandpass/stop elliptic filter was being designed with $n=5$, then $m=4$ and the direct digital design method (with precalculated prototypes) has an advantage of 12 multiplications and additions compared to approximately 24 for the analog method.

A final consideration in favor of analog design is the complexity of the elements of the transformation matrices. Since the elements of the digital transformation matrices require much more calculation than the elements of the analog transformation matrices or the bilinear transformation (compare Tables 1-5), the problem of roundoff error is much more serious in the case of the digital to digital design method.

D. IMPLEMENTATION

The final step is to implement the algorithms developed previously in a computer program to assist in the teaching of digital filter design. Since the program is intended to be

used for instructional purposes, the numerical considerations discussed earlier are not considered as important as conceptual simplicity. The direct design method was chosen for implementation as the design constants can be found easily without reference to "prewarping." The completed program, called 'DFCADD' (Digital Filter Computer Aided Direct Design), is contained in Appendix A with its own documentation. The program follows the procedures set forth in Chapter 10 of Reference 2. Prototype coefficients are stored within the program for first-order to fifth-order Butterworth and Chebyshev (1/2, 1, & 2 dB ripples) filters. While second-order to sixth-order elliptic lowpass prototype coefficients are calculated within the program for any combination of 1/2, 1, 2, and 3 dB passband ripple and -20, -30, -40, -50, -60, and -70 dB stopband attenuation, with the help of a program developed by Professor D.E. Kirk of the Naval Postgraduate School and References 4 and 5. As a final option, the user may enter his/her own prototype coefficients for up to a tenth-order analog prototype. After calculating the digital filter coefficients an option exists for the program to create a data file, titled "filter", with the digital filter magnitude and phase as a function of the digital frequency from zero to π . This data file can then be used with any plotting routine to generate a plot of the frequency response of the filter.

LIST OF REFERENCES

1. Moose, P.H. of the United States Naval Postgraduate School, Monterey, California, Letter to Strum, R.D. and Kirk, D.E., Subject: Algorithm for Digital-Digital Transformations, 5 June 1987.
2. Strum, R.D. and Kirk, D.E., First Principles of Discrete Systems and Digital Signal Processing, pp. 612-701, Addison-Wesley, 1988.
3. Constantinides, A.G., "Spectral Transformations for Digital Filters, "Proceedings of IEEE, p. 117,(August): 1585-1590.
4. Ludeman, L.C., Fundamentals of Digital Signal Processing, pp. 149-160, Harper and Row, 1986.
5. Antoniou, A., Digital Filters: Analysis and Design, pp. 114-125, McGraw-Hill, 1979.

APPENDIX DFCADD FORTRAN PROGRAM

FILE: DFCADD FORTRAN A

```

C *****
C * THIS PROGRAM ASSISTS IN THE CALCULATION OF DIGITAL FILTER *
C * COEFFICIENTS ACCORDING TO THE PROCEDURE SET FORTH IN CHAPTER *
C * 10 (SUMMARIZED ON PG 695) OF - FIRST PRINCIPLES OF DISCRETE *
C * SYSTEMS AND DIGITAL SIGNAL PROCESSING - BY R.D. STRUM AND *
C * D.E. KIRK (REFERENCE(2)) *
C *****
C INTEGER ORDER,TYPE,PASS,ORDER2,END,AFORDF
C REAL KPR,LAM,OMEGA(20),V(20),AO(20),BO(20),B1(20)
C COMPLEX Z,NEWZ,ZN,ZD,HOFZN,HOFZD,HOFZ
C DOUBLE PRECISION TEMP,SUMR
C DOUBLE PRECISION R,S0,A01,A02,B01,B02,B11,B12,H0
C DOUBLE PRECISION A03,A04,B03,B04,B13,B14
C DOUBLE PRECISION SFREQ,AFC,DFC,ALPHA,PI,PID4,DFCD2
C DOUBLE PRECISION ALC,AUC,DLG,DUC,AK,B,C,DLCD2,DUCD2
C DOUBLE PRECISION APROT,BILINR,DPROT,LPHPTR,BPBSTR,DFLTR
C *****
C FORMAT OF ARRAYS:
C -APROT CONTAINS THE ANALOG PROTOTYPE
C APROT(NORD,PWR)
C -BILINR IS THE ANALOG-DIGITAL BILINEAR TRANSFORMATION MATRIX
C BILINR(ORDER,ROW,COL)
C -DPROT IS THE DIGITAL PROTOTYPE FROM APROT AND BILINR
C DPROT(NORD,PWR)
C -LPHPTR IS THE LP&HP DIGITAL-DIGITAL TRANSFORMATION MATRIX
C LPHPTR(ORDER,ROW,COL)
C -DFLTR IS THE FINAL DIGITAL FILTER
C DFLTR(NORD,PWR)
C -BPBSTR IS THE BP&BS DIGITAL-DIGITAL TRANSFORMATION MATRIX
C BPBSTR(ORDER,ROW,COL)
C *****
C DIMENSION APROT(2,0:10)
C DIMENSION BILINR(10,0:10,0:10)
C DIMENSION DPROT(2,0:10)
C DIMENSION LPHPTR(10,0:10,0:10)
C DIMENSION BPBSTR(10,0:10,0:20)
C DIMENSION DFLTR(2,0:20)
C *****
C OPEN(UNIT=8,FILE='FILTER DATA',STATUS='UNKNOWN')
C *****
C PI=3.1415926535898D+00
C SPI=3.14159
C PID4=PI/4.0D+00
C *****
C CLEAR APROT PRIOR TO LOADING
C DO 1 K=1,2
C DO 2 L=0,10
C APROT(K,L)=0.0D+00
C CONTINUE
C CONTINUE
C *****
C BEGIN INTERACTION
C CALL EXCMS ('CLRSCRN')
C WRITE(6,*) 'THIS IS A PROGRAM TO CALCULATE DIGITAL FILTERS USING'
C WRITE(6,*) 'THE DIRECT DESIGN METHOD.'
C WRITE(6,*)
C WRITE(6,*) '*****'
C WRITE(6,*) '**WARNING-THIS PROGRAM IS NOT USER FRIENDLY!!! **'
C WRITE(6,*) '** YOU MUST ENTER VALUES OF THE APPROPRIATE TYPE**'
C WRITE(6,*) '** (REAL OR INTEGER), AND PROPER RANGE **'
C WRITE(6,*) '*****'
C WRITE(6,*)
C WRITE(6,*) ' ** ENTER 1 TO CONTINUE ** '
C READ(5,*) LOOK
C CALL EXCMS ('CLRSCRN')
C WRITE(6,*) 'WHAT TYPE OF FILTER DO YOU WANT?'
C WRITE(6,*) '(1-4 ALLOW ONLY UP TO 5TH ORDER PROTOTYPES)'
C WRITE(6,*) ' 1-BUTTERNORTH '
C WRITE(6,*) ' 2-CHEBYSHEV WITH 1/2 DB RIPPLE '
C WRITE(6,*) ' 3-CHEBYSHEV WITH 1 DB RIPPLE '
C WRITE(6,*) ' 4-CHEBYSHEV WITH 2 DB RIPPLE'

```

FILE: DFCADD FORTRAN A

```

WRITE(6,*) ' 5-ELLIPTIC (ALLOWS UP TO A 6TH ORDER PROTOTYPE)' NEW00730
WRITE(6,*) ' -PASSBAND RIPPLE 0.5,1.0,2.0,OR 3.0 DB' NEW00740
WRITE(6,*) ' -STOPBAND ATTENUATION -20,-30,-40,-50,-60, ' NEW00750
WRITE(6,*) ' OR -70 DB ' NEW00760
WRITE(6,*) ' 6-OTHER - ALLOWS YOU TO USE UP TO A 10TH ORDER' NEW00770
WRITE(6,*) ' PROTOTYPE, HOWEVER YOU MUST ENTER THE ANALOG' NEW00780
WRITE(6,*) ' PROTOTYPE COEFFICIENTS ' NEW00790
WRITE(6,*) ' ** INTEGER 1-6 **' NEW00800
READ(5,*) TYPE NEW00810
***** NEW00820
C CALL EXCMS ('CLRSCRN') NEW00830
14 WRITE(6,*) NEW00840
15 WRITE(6,*) 'WHAT IS THE ORDER OF THE PROTOTYPE YOU WISH TO USE?' NEW00850
WRITE(6,*) ' - MUST BE AN INTEGER IN THE RANGE 1-5 UNLESS YOU' NEW00860
WRITE(6,*) ' CHOSE TO ENTER YOUR OWN PROTOTYPE (THEN 1-10)' NEW00870
WRITE(6,*) ' OR YOU CHOSE AN ELLIPTIC FILTER (THEN 1-6)' NEW00880
WRITE(6,*) ' - OR ENTER "0" IF YOU WANT HELP SELECTING THE ORDER' NEW00890
WRITE(6,*) ' BY CHANGING SPECIFICATION FREQUENCIES TO THE' NEW00900
WRITE(6,*) ' CORRESPONDING DIGITAL PROTOTYPE FREQUENCIES ' NEW00910
WRITE(6,*) ' NOTE: YOU WILL NEED PROTOTYPE FREQUENCY ' NEW00920
WRITE(6,*) ' RESPONSE PLOTS LIKE THOSE GIVEN IN REF(2) FOR ' NEW00930
WRITE(6,*) ' BUTTERWORTH FILTERS (FIG 10.33, PGS 691&692) ' NEW00940
WRITE(6,*) ' AND CHEBYSHEV FILTERS (FIG 10.34, PGS 696-698) ' NEW00950
READ(5,*) ORDER NEW00960
IF((TYPE.GT.4).AND.(TYPE.LT.8))THEN NEW00970
IF(ORDER.EQ.1)THEN NEW00980
WRITE(6,*) 'ELLIPTIC FILTERS GIVEN FOR 2ND-9TH ORDER ONLY' NEW00990
WRITE(6,*) NEW01000
GO TO 10 NEW01010
ENDIF NEW01020
ENDIF NEW01030
C CALL EXCMS ('CLRSCRN') NEW01040
WRITE(6,*) 'DO YOU WANT TO ENTER SPECIFICATION FREQUENCIES' NEW01050
WRITE(6,*) 'AS ANALOG OR DIGITAL FREQUENCIES?' NEW01060
WRITE(6,*) ' 1-ANALOG ' NEW01070
WRITE(6,*) ' 2-DIGITAL ' NEW01080
WRITE(6,*) ' ** INTEGER 1 OR 2 ** ' NEW01090
READ(5,*) AFORDF NEW01100
CALL EXCMS ('CLRSCRN') NEW01110
IF(AFORDF.EQ.1)THEN NEW01120
WRITE(6,*) 'WHAT IS THE SAMPLING FREQUENCY IN KHZ?' NEW01130
WRITE(6,*) ' ** DECIMAL **' NEW01140
READ(5,*) SSFREQ NEW01150
SSFREQ=DBLE(SSFREQ) NEW01160
ENDIF NEW01170
CALL EXCMS ('CLRSCRN') NEW01180
WRITE(6,*) 'WHAT TYPE OF PASSBAND DUE YOU WANT?' NEW01190
WRITE(6,*) ' 1-LOWPASS' NEW01200
WRITE(6,*) ' 2-HIGHPASS' NEW01210
WRITE(6,*) ' 3-BANDPASS' NEW01220
WRITE(6,*) ' 4-BANDSTOP' NEW01230
WRITE(6,*) ' ** INTEGER 1-4 ** ' NEW01240
READ(5,*) PASS NEW01250
C FORMULAS FOR HIGHPASS AND LOWPASS NEW01260
IF((PASS.EQ.1).OR.(PASS.EQ.2))THEN NEW01270
CALL EXCMS ('CLRSCRN') NEW01280
IF(AFORDF.EQ.1)THEN NEW01290
WRITE(6,*) 'WHAT IS THE CRITICAL FREQUENCY IN KHZ? ' NEW01300
WRITE(6,*) ' - NEG 3DB PT FOR BUTTERWORTH ' NEW01310
WRITE(6,*) ' - RIPPLE EDGE FOR ELLIPTIC AND CHEBYSHEV' NEW01320
WRITE(6,*) ' ** DECIMAL ** ' NEW01330
READ(5,*) SAFC NEW01340
AFC=DBLE(SAFC) NEW01350
DFC=(AFC/SSFREQ)*PI*(2.0D+00) NEW01360
ENDIF NEW01370
IF(AFORDF.EQ.2)THEN NEW01380
WRITE(6,*) 'WHAT IS THE CRITICAL DIGITAL FREQUENCY?' NEW01390
NEW01400
NEW01410
NEW01420
NEW01430
NEW01440

```

FILE: DFCADD FORTRAN A

```

      WRITE(6,*) ' - NEG 3DB FOR BUTTERWORTH ' NEW01450
      WRITE(6,*) ' - RIPPLE EDGE FOR ELLIPTIC AND CHEBYSHEV' NEW01460
      WRITE(6,*) ' ** DECIMAL ** ' NEW01470
      READ(5,*) SDFC NEW01480
      DFC=DBLE(SDFC) NEW01490
      ENDIF NEW01500
      DFC2=DFC/2.0D+00 NEW01510
C     CALCULATE THE APPROPRIATE VALUES OF THE DESIGN CONSTANT ALPHA NEW01520
C     FROM TABLE 10.7 REFERENCE (2) AND REFERENCE (3) NEW01530
      IF(PASS.EQ.1)THEN NEW01540
        ALPHA=DSIN((PID4)-(DFCD2))/DSIN((PID4)+(DFCD2)) NEW01550
      ENDIF NEW01560
      IF(PASS.EQ.2)THEN NEW01570
        ALPHA=DCOS((PID4)-(DFCD2))/DCOS((PID4)+(DFCD2)) NEW01580
      ENDIF NEW01590
C     ***** NEW01600
C     OPTION TO SHOW CALCULATED VALUE OF ALPHA NEW01610
      CALL EXCMS ('CLRSCRN') NEW01620
      WRITE(6,*) 'DO YOU WISH TO SEE THE CALCULATED VALUE OF ALPHA?' NEW01630
      WRITE(6,*) ' ** 1-YES/2-NO **' NEW01640
      READ(5,*) LOOK NEW01650
      IF(LOOK.EQ.1)THEN NEW01660
        WRITE(6,*) NEW01670
        WRITE(6,903) ALPHA NEW01680
        WRITE(6,*) NEW01690
        WRITE(6,*) '** ENTER 1 TO CONTINUE **' NEW01700
        READ(5,*) LOOK NEW01710
      ENDIF NEW01720
C     ***** NEW01730
C     BANDPASS AND BANDSTOP FORMULAS NEW01740
C     IF((PASS.EQ.3).OR.(PASS.EQ.4))THEN NEW01750
      CALL EXCMS ('CLRSCRN') NEW01760
      IF(AFORDF.EQ.1)THEN NEW01770
        WRITE(6,*) 'WHAT IS THE LOWER CRITICAL FREQUENCY IN KHZ?' NEW01780
        WRITE(6,*) ' - NEG 3DB FOR BUTTERWORTH' NEW01790
        WRITE(6,*) ' - RIPPLE EDGE FOR ELLIPTIC AND CHEBYSHEV ' NEW01800
        WRITE(6,*) ' ** DECIMAL **' NEW01810
        READ(5,*) SALC NEW01820
        ALC=DBLE(SALC) NEW01830
        CALL EXCMS ('CLRSCRN') NEW01840
        WRITE(6,*) 'WHAT IS THE UPPER CRITICAL FREQUENCY IN KHZ?' NEW01850
        WRITE(6,*) ' - NEG 3DB FOR BUTTERWORTH' NEW01860
        WRITE(6,*) ' - RIPPLE EDGE FOR ELLIPTIC AND CHEBYSHEV ' NEW01870
        WRITE(6,*) ' ** DECIMAL **' NEW01880
        READ(5,*) SAUC NEW01890
        AUC=DBLE(SAUC) NEW01900
        DLC=(ALC/SFREQ)*(2.0D+00)*PI NEW01910
        DUC=(AUC/SFREQ)*(2.0D+00)*PI NEW01920
      ENDIF NEW01930
      IF(AFORDF.EQ.2)THEN NEW01940
        WRITE(6,*) 'WHAT IS THE LOWER CRITICAL DIGITAL FREQUENCY?' NEW01950
        WRITE(6,*) ' - NEG 3DB FOR BUTTERWORTH ' NEW01960
        WRITE(6,*) ' - RIPPLE EDGE FOR CHEBYSHEV AND ELLIPTIC' NEW01970
        WRITE(6,*) ' ** DECIMAL ** ' NEW01980
        READ(5,*) SDLC NEW01990
        DLC=DBLE(SDLC) NEW02000
        CALL EXCMS ('CLRSCRN') NEW02010
        WRITE(6,*) 'WHAT IS THE UPPER CRITICAL DIGITAL FREQUENCY?' NEW02020
        WRITE(6,*) ' - NEG 3DB FOR BUTTERWORTH ' NEW02030
        WRITE(6,*) ' - RIPPLE EDGE FOR CHEBYSHEV AND ELLIPTIC' NEW02040
        WRITE(6,*) ' ** DECIMAL ** ' NEW02050
        READ(5,*) SDUC NEW02060
        DUC=DBLE(SDUC) NEW02070
      ENDIF NEW02080
      DLC2=DLC/2.0D+00 NEW02090
      DUC2=DUC/2.0D+00 NEW02100
C     CALCULATE APPROPRIATE VALUES OF DESIGN CONSTANTS ALPHA AND K NEW02110
C     FROM TABLE 10.7 REFERENCE (2) AND REFERENCE (3) NEW02120
      ALPHA=DCOS((DUC2)+(DLC2))/DCOS((DUC2)-(DLC2)) NEW02130
      IF(PASS.EQ.4)THEN NEW02140
        RK=DTAN(PID4)*DTAN((DUC2)-(DLC2)) NEW02150
      ENDIF NEW02160

```

11

FILE: DFCADD FORTRAN A

```

      B=((2.0D+00)*ALPHA)/(RK+1.0D+00)
      C=((1.0D+00)-RK)/((1.0D+00)+RK)
    ENDIF
    IF(PASS.EQ.3)THEN
      RK=DTAN(PID4)/DTAN((DUCD2)-(DLCD2))
      B=((2.0D+00)*ALPHA*RK)/(RK+(1.0D+00))
      C=(RK-(1.0D+00))/(RK+(1.0D+00))
    ENDIF
  C  OPTION TO SHOW CALCULATED VALUE OF ALPHA AND K
    CALL EXCMS ('CLRSCRN')
    WRITE(6,*) 'DO YOU WISH TO SEE THE CALCULATED VALUE OF ALPHA '
    WRITE(6,*) ' AND K? '
    WRITE(6,*) ' ** 1=YES/2=NO **'
    READ(5,*) LOOK
    IF(LOOK.EQ.1)THEN
      WRITE(6,*)
      WRITE(6,903) ALPHA
      WRITE(6,904) RK
      WRITE(6,*)
      WRITE(6,*) '*** ENTER 1 TO CONTINUE ***'
      READ(5,*) LOOK
    ENDIF
  C  ENDIF
  C  *****
  C  OPTION TO ASSIST IN DETERMINING THE PROTOTYPE ORDER BY CHANGING
  C  SPECIFICATION FREQUENCIES TO APPROPRIATE PROTOTYPE FREQUENCIES
  C  FROM EQN 10.274 REFERENCE(1)
  16 IF(ORDER.EQ.0)THEN
    CALL EXCMS ('CLRSCRN')
    IF(AFORDF.EQ.1)THEN
      WRITE(6,*) 'WHAT IS THE ANALOG SPECIFICATION FREQUENCY (KHZ)'
      WRITE(6,*) 'THAT YOU WANT TO CONVERT TO A DIGITAL'
      WRITE(6,*) 'LOWPASS PROTOTYPE FREQUENCY?'
      WRITE(6,*) ' ** DECIMAL **'
      READ(5,*) F
      DF=(2.0)*SPI*(F/SSFREQ)
    ENDIF
    IF(AFORDF.EQ.2)THEN
      WRITE(6,*) 'WHAT IS THE DIGITAL SPECIFICATION FREQUENCY'
      WRITE(6,*) 'THAT YOU WANT TO CONVERT TO A DIGITAL'
      WRITE(6,*) 'LOWPASS PROTOTYPE FREQUENCY?'
      WRITE(6,*) ' ** DECIMAL **'
      READ(5,*) DF
    ENDIF
    X=COS(DF)
    Y=SIN(DF)
    Z=CMPLX(X,Y)
    IF(PASS.EQ.1)THEN
      AS=REAL(ALPHA)
      NEWZ=(Z-AS)/((1.0)-(AS*Z))
    ENDIF
    IF(PASS.EQ.2)THEN
      AS=REAL(ALPHA)
      NEWZ=(-1.0)*((Z-AS)/((1.0)-(AS*Z)))
    ENDIF
    IF(PASS.EQ.3)THEN
      BS=REAL(B)
      CS=REAL(C)
      NEWZ=(-1.0)*(((Z**2)-(BS*Z)+CS)/((1.0)-(BS*Z)+(CS*(Z**2))))
    ENDIF
    IF(PASS.EQ.4)THEN
      BS=REAL(B)
      CS=REAL(C)
      NEWZ=((Z**2)-(BS*Z)+CS)/((1.0)-(BS*Z)+(CS*(Z**2)))
    ENDIF
    PTHETA=ABS(ATAN(AIMAG(NEWZ)/REAL(NEWZ)))
    IF(PASS.EQ.1)THEN
      IF(F.GT.SAFC)THEN
        PTHETA=SPI-PTHETA
      ENDIF
    ENDIF
    IF(PASS.EQ.2)THEN

```

NEW02170
 NEW02180
 NEW02190
 NEW02200
 NEW02210
 NEW02220
 NEW02230
 NEW02240
 NEW02250
 NEW02260
 NEW02270
 NEW02280
 NEW02290
 NEW02300
 NEW02310
 NEW02320
 NEW02330
 NEW02340
 NEW02350
 NEW02360
 NEW02370
 NEW02380
 NEW02390
 NEW02400
 NEW02410
 NEW02420
 NEW02430
 NEW02440
 NEW02450
 NEW02460
 NEW02470
 NEW02480
 NEW02490
 NEW02500
 NEW02510
 NEW02520
 NEW02530
 NEW02540
 NEW02550
 NEW02560
 NEW02570
 NEW02580
 NEW02590
 NEW02600
 NEW02610
 NEW02620
 NEW02630
 NEW02640
 NEW02650
 NEW02660
 NEW02670
 NEW02680
 NEW02690
 NEW02700
 NEW02710
 NEW02720
 NEW02730
 NEW02740
 NEW02750
 NEW02760
 NEW02770
 NEW02780
 NEW02790
 NEW02800
 NEW02810
 NEW02820
 NEW02830
 NEW02840
 NEW02850
 NEW02860
 NEW02870
 NEW02880

FILE: DFCADD FORTRAN A

```

      IF(F.LT.SAFC)THEN
        PTHETA=SPI-PTHETA
      ENDIF
    ENDIF
    IF(PASS.EQ.3)THEN
      IF((F.LT.SALC).OR.(F.GT.SAUC))THEN
        PTHETA=SPI-PTHETA
      ENDIF
    ENDIF
    IF(PASS.EQ.4)THEN
      IF((F.GT.SALC).AND.(F.LT.SAUC))THEN
        PTHETA=SPI-PTHETA
      ENDIF
    ENDIF
    WRITE(6,905) PTHETA
    WRITE(6,*)
    WRITE(6,*) 'WANT TO CONVERT ANOTHER FREQUENCY?'
    WRITE(6,*) ' ** 1-YES/2-NO **'
    READ(5,*) LOOK
    IF(LOOK.EQ.1)GO TO 16
    IF(LOOK.EQ.2)THEN
      WRITE(6,*)
      WRITE(6,*) 'WHAT ORDER PROTOTYPE DO YOU WANT TO USE?'
      WRITE(6,*) ' ** INTEGER 1-5 ** '
      READ(5,*) ORDER
    ENDIF
  ENDIF
  C *****
  C LOAD APROT MATRIX WITH APPROPRIATE COEFFICIENTS
  C APROT(TYPE=1) ARE BUTTERWORTH FILTERS
    IF(TYPE.EQ.1)THEN
      IF(ORDER.EQ.1)THEN
        APROT(1,0)=1.0D+00
        APROT(2,0)=1.0D+00
        APROT(2,1)=1.0D+00
      ELSEIF(ORDER.EQ.2)THEN
        APROT(1,0)=1.0D+00
        APROT(2,0)=1.0D+00
        APROT(2,1)=DSQRT(2.0D+00)
        APROT(2,2)=1.0D+00
      ELSEIF(ORDER.EQ.3)THEN
        APROT(1,0)=1.0D+00
        APROT(2,0)=1.0D+00
        APROT(2,1)=2.0D+00
        APROT(2,2)=2.0D+00
        APROT(2,3)=1.0D+00
      ELSEIF(ORDER.EQ.4)THEN
        APROT(1,0)=1.0D+00
        APROT(2,0)=1.0D+00
        APROT(2,1)=2.613126D+00
        APROT(2,2)=3.414214D+00
        APROT(2,3)=2.613126D+00
        APROT(2,4)=1.0D+00
      ELSEIF(ORDER.EQ.5)THEN
        APROT(1,0)=1.0D+00
        APROT(2,0)=1.0D+00
        APROT(2,1)=3.236068D+00
        APROT(2,2)=5.236068D+00
        APROT(2,3)=5.236068D+00
        APROT(2,4)=3.236068D+00
        APROT(2,5)=1.0D+00
      ENDIF
    ENDIF
  C APROT(TYPE=2) ARE CHEBYSHEV FILTERS WITH .5DB RIPPLE
    IF(TYPE.EQ.2)THEN
      IF(ORDER.EQ.1)THEN
        APROT(1,0)=2.862775D+00
        APROT(2,0)=2.862775D+00
        APROT(2,1)=1.0D+00
      ELSEIF(ORDER.EQ.2)THEN
        APROT(1,0)=1.431388D+00
        APROT(2,0)=1.516203D+00

```

NEW02890
 NEW02900
 NEW02910
 NEW02920
 NEW02930
 NEW02940
 NEW02950
 NEW02960
 NEW02970
 NEW02980
 NEW02990
 NEW03000
 NEW03010
 NEW03020
 NEW03030
 NEW03040
 NEW03050
 NEW03060
 NEW03070
 NEW03080
 NEW03090
 NEW03100
 NEW03110
 NEW03120
 NEW03130
 NEW03140
 NEW03150
 NEW03160
 NEW03170
 NEW03180
 NEW03190
 NEW03200
 NEW03210
 NEW03220
 NEW03230
 NEW03240
 NEW03250
 NEW03260
 NEW03270
 NEW03280
 NEW03290
 NEW03300
 NEW03310
 NEW03320
 NEW03330
 NEW03340
 NEW03350
 NEW03360
 NEW03370
 NEW03380
 NEW03390
 NEW03400
 NEW03410
 NEW03420
 NEW03430
 NEW03440
 NEW03450
 NEW03460
 NEW03470
 NEW03480
 NEW03490
 NEW03500
 NEW03510
 NEW03520
 NEW03530
 NEW03540
 NEW03550
 NEW03560
 NEW03570
 NEW03580
 NEW03590
 NEW03600

FILE: DFCADD FORTRAN A

```

      APROT(2,1)=1.425625D+00
      APROT(2,2)=1.0D+00
      ELSEIF(ORDER.EQ.3)THEN
        APROT(1,0)=0.715694D+00
        APROT(2,0)=0.715694D+00
        APROT(2,1)=1.534895D+00
        APROT(2,2)=1.252913D+00
        APROT(2,3)=1.0D+00
      ELSEIF(ORDER.EQ.4)THEN
        APROT(1,0)=0.357847D+00
        APROT(2,0)=0.379051D+00
        APROT(2,1)=1.025455D+00
        APROT(2,2)=1.716866D+00
        APROT(2,3)=1.197386D+00
        APROT(2,4)=1.0D+00
      ELSEIF(ORDER.EQ.5)THEN
        APROT(1,0)=0.178923D+00
        APROT(2,0)=0.178923D+00
        APROT(2,1)=0.752518D+00
        APROT(2,2)=1.309575D+00
        APROT(2,3)=1.937368D+00
        APROT(2,4)=1.172491D+00
        APROT(2,5)=1.0D+00
      ENDIF
    ENDIF
  C  APROT(TYPE=3) ARE CHEBYSHEV FILTERS WITH 1DB RIPPLE
    IF(TYPE.EQ.3)THEN
      IF(ORDER.EQ.1)THEN
        APROT(1,0)=1.196523D+00
        APROT(2,0)=1.196523D+00
        APROT(2,1)=1.0D+00
      ELSEIF(ORDER.EQ.2)THEN
        APROT(1,0)=0.982613D+00
        APROT(2,0)=1.102510D+00
        APROT(2,1)=1.097734D+00
        APROT(2,2)=1.0D+00
      ELSEIF(ORDER.EQ.3)THEN
        APROT(1,0)=0.491307D+00
        APROT(2,0)=0.491307D+00
        APROT(2,1)=1.238409D+00
        APROT(2,2)=0.988341D+00
        APROT(2,3)=1.0D+00
      ELSEIF(ORDER.EQ.4)THEN
        APROT(1,0)=0.245653D+00
        APROT(2,0)=0.275628D+00
        APROT(2,1)=0.742619D+00
        APROT(2,2)=1.453925D+00
        APROT(2,3)=0.952811D+00
        APROT(2,4)=1.0D+00
      ELSEIF(ORDER.EQ.5)THEN
        APROT(1,0)=0.122827D+00
        APROT(2,0)=0.122827D+00
        APROT(2,1)=0.580534D+00
        APROT(2,2)=0.974396D+00
        APROT(2,3)=1.688816D+00
        APROT(2,4)=0.936820D+00
        APROT(2,5)=1.0D+00
      ENDIF
    ENDIF
  C  APROT(TYPE=4) ARE CHEBYSHEV FILTERS WITH 2DB RIPPLE
    IF(TYPE.EQ.4)THEN
      IF(ORDER.EQ.1)THEN
        APROT(1,0)=1.307560D+00
        APROT(2,0)=1.307560D+00
        APROT(2,1)=1.0D+00
      ELSEIF(ORDER.EQ.2)THEN
        APROT(1,0)=0.505803D+00
        APROT(2,0)=0.636768D+00
        APROT(2,1)=0.803816D+00
        APROT(2,2)=1.0D+00
      ELSEIF(ORDER.EQ.3)THEN
        APROT(1,0)=0.326890D+00

```

NEW03610
 NEW03620
 NEW03630
 NEW03640
 NEW03650
 NEW03660
 NEW03670
 NEW03680
 NEW03690
 NEW03700
 NEW03710
 NEW03720
 NEW03730
 NEW03740
 NEW03750
 NEW03760
 NEW03770
 NEW03780
 NEW03790
 NEW03800
 NEW03810
 NEW03820
 NEW03830
 NEW03840
 NEW03850
 NEW03860
 NEW03870
 NEW03880
 NEW03890
 NEW03900
 NEW03910
 NEW03920
 NEW03930
 NEW03940
 NEW03950
 NEW03960
 NEW03970
 NEW03980
 NEW03990
 NEW04000
 NEW04010
 NEW04020
 NEW04030
 NEW04040
 NEW04050
 NEW04060
 NEW04070
 NEW04080
 NEW04090
 NEW04100
 NEW04110
 NEW04120
 NEW04130
 NEW04140
 NEW04150
 NEW04160
 NEW04170
 NEW04180
 NEW04190
 NEW04200
 NEW04210
 NEW04220
 NEW04230
 NEW04240
 NEW04250
 NEW04260
 NEW04270
 NEW04280
 NEW04290
 NEW04300
 NEW04310
 NEW04320

FILE: DFCADD FORTRAN A

```

      APROT(2,0)=0.326890D+00
      APROT(2,1)=1.022190D+00
      APROT(2,2)=0.737822D+00
      APROT(2,3)=1.0D+00
    ELSEIF(ORDER.EQ.4)THEN
      APROT(1,0)=0.163445D+00
      APROT(2,0)=0.205765D+00
      APROT(2,1)=0.516798D+00
      APROT(2,2)=1.256482D+00
      APROT(2,3)=0.716215D+00
      APROT(2,4)=1.0D+00
    ELSEIF(ORDER.EQ.5)THEN
      APROT(1,0)=0.081723D+00
      APROT(2,0)=0.081723D+00
      APROT(2,1)=0.459349D+00
      APROT(2,2)=0.693477D+00
      APROT(2,3)=1.499543D+00
      APROT(2,4)=0.706461D+00
      APROT(2,5)=1.0D+00
    ENDIF
  ENDIF
C   APROT(TYPE=5) ARE ELLIPTIC FILTERS
C   ELLIPTIC COEFFICIENTS ARE CALCULATED USING A PROGRAM DEVELOPED
C   BY PROFESSOR D. E. KIRK, NAVAL POSTGRADUATE SCHOOL, MONTEREY,
C   CALIFORNIA 93943
      A02=0.0D+00
      B02=0.0D+00
      B12=0.0D+00
      S0=0.0D+00
    IF(TYPE.EQ.5)THEN
      CALL EXCMS ('CLRSCRN')
      WRITE(6,*) 'WHAT PASSBAND RIPPLE DO YOU WANT? (IN DB)'
      WRITE(6,*) '*** MUST BE EITHER 0.5, 1.0, 2.0, OR 3.0 ***'
      READ(5,*) ASUBP
      WRITE(6,*)
      WRITE(6,*) 'WHAT STOPBAND ATTENUATION DO YOU WANT? (IN DB)'
      WRITE(6,*) '*** MUST BE -20,-30,-40,-50,-60,-70 ***'
      WRITE(6,*) '*** DO NOT INCLUDE A DECIMAL ***'
      READ(5,*) NATTN
      NATTN=-1*NATTN
      ASUBA=REAL(NATTN)
      IF(ASUBP.EQ.0.5)THEN
        IF(NATTN.EQ.20)THEN
          IF(ORDER.EQ.2)SR=2.76261
          IF(ORDER.EQ.3)SR=1.42189
          IF(ORDER.EQ.4)SR=1.13188
          IF(ORDER.EQ.5)SR=1.04465
          IF(ORDER.EQ.6)SR=1.01553
        ENDIF
        IF(NATTN.EQ.30)THEN
          IF(ORDER.EQ.2)SR=4.80880
          IF(ORDER.EQ.3)SR=1.92322
          IF(ORDER.EQ.4)SR=1.32446
          IF(ORDER.EQ.5)SR=1.12912
          IF(ORDER.EQ.6)SR=1.05394
        ENDIF
        IF(NATTN.EQ.40)THEN
          IF(ORDER.EQ.2)SR=8.848925
          IF(ORDER.EQ.3)SR=2.71147
          IF(ORDER.EQ.4)SR=1.62842
          IF(ORDER.EQ.5)SR=1.27264
          IF(ORDER.EQ.6)SR=1.12697
        ENDIF
        IF(NATTN.EQ.50)THEN
          IF(ORDER.EQ.2)SR=15.06069
          IF(ORDER.EQ.3)SR=3.90430
          IF(ORDER.EQ.4)SR=2.06924
          IF(ORDER.EQ.5)SR=1.48469
          IF(ORDER.EQ.6)SR=1.24101
        ENDIF
        IF(NATTN.EQ.60)THEN

```

NEW04330
 NEW04340
 NEW04350
 NEW04360
 NEW04370
 NEW04380
 NEW04390
 NEW04400
 NEW04410
 NEW04420
 NEW04430
 NEW04440
 NEW04450
 NEW04460
 NEW04470
 NEW04480
 NEW04490
 NEW04500
 NEW04510
 NEW04520
 NEW04530
 NEW04540
 NEW04550
 NEW04560
 NEW04570
 NEW04580
 NEW04590
 NEW04600
 NEW04610
 NEW04620
 NEW04630
 NEW04640
 NEW04650
 NEW04660
 NEW04670
 NEW04680
 NEW04690
 NEW04700
 NEW04710
 NEW04720
 NEW04730
 NEW04740
 NEW04750
 NEW04760
 NEW04770
 NEW04780
 NEW04790
 NEW04800
 NEW04810
 NEW04820
 NEW04830
 NEW04840
 NEW04850
 NEW04860
 NEW04870
 NEW04880
 NEW04890
 NEW04900
 NEW04910
 NEW04920
 NEW04930
 NEW04940
 NEW04950
 NEW04960
 NEW04970
 NEW04980
 NEW04990
 NEW05000
 NEW05010
 NEW05020
 NEW05030
 NEW05040

11

FILE: DFCADD FORTRAN A

```

IF(ORDER.EQ.2)SR=26.76230
IF(ORDER.EQ.3)SR=5.67937
IF(ORDER.EQ.4)SR=2.68325
IF(ORDER.EQ.5)SR=1.77664
IF(ORDER.EQ.6)SR=1.40138
ENDIF
IF(NATTN.EQ.70)THEN
IF(ORDER.EQ.2)SR=47.58053
IF(ORDER.EQ.3)SR=8.30124
IF(ORDER.EQ.4)SR=3.52135
IF(ORDER.EQ.5)SR=2.16377
IF(ORDER.EQ.6)SR=1.61413
ENDIF
ENDIF
IF(ASUBP.EQ.1.0)THEN
IF(NATTN.EQ.20)THEN
IF(ORDER.EQ.2)SR=2.32474
IF(ORDER.EQ.3)SR=1.30797
IF(ORDER.EQ.4)SR=1.09029
IF(ORDER.EQ.5)SR=1.02826
IF(ORDER.EQ.6)SR=1.00902
ENDIF
IF(NATTN.EQ.30)THEN
IF(ORDER.EQ.2)SR=4.00423
IF(ORDER.EQ.3)SR=1.73254
IF(ORDER.EQ.4)SR=1.25040
IF(ORDER.EQ.5)SR=1.09554
IF(ORDER.EQ.6)SR=1.03799
ENDIF
IF(NATTN.EQ.40)THEN
IF(ORDER.EQ.2)SR=7.04488
IF(ORDER.EQ.3)SR=2.41619
IF(ORDER.EQ.4)SR=1.51549
IF(ORDER.EQ.5)SR=1.21868
IF(ORDER.EQ.6)SR=1.09887
ENDIF
IF(NATTN.EQ.50)THEN
IF(ORDER.EQ.2)SR=12.48471
IF(ORDER.EQ.3)SR=3.46061
IF(ORDER.EQ.4)SR=1.90819
IF(ORDER.EQ.5)SR=1.40723
IF(ORDER.EQ.6)SR=1.19891
ENDIF
IF(NATTN.EQ.60)THEN
IF(ORDER.EQ.2)SR=22.17688
IF(ORDER.EQ.3)SR=5.02121
IF(ORDER.EQ.4)SR=2.46079
IF(ORDER.EQ.5)SR=1.67161
IF(ORDER.EQ.6)SR=1.34354
ENDIF
IF(NATTN.EQ.70)THEN
IF(ORDER.EQ.2)SR=39.42285
IF(ORDER.EQ.3)SR=7.33052
IF(ORDER.EQ.4)SR=3.21902
IF(ORDER.EQ.5)SR=2.02567
IF(ORDER.EQ.6)SR=1.53842
ENDIF
ENDIF
IF(ASUBP.EQ.2.0)THEN
IF(NATTN.EQ.20)THEN
IF(ORDER.EQ.2)SR=1.94332
IF(ORDER.EQ.3)SR=1.20808
IF(ORDER.EQ.4)SR=1.05569
IF(ORDER.EQ.5)SR=1.01567
IF(ORDER.EQ.6)SR=1.00447
ENDIF
IF(NATTN.EQ.30)THEN
IF(ORDER.EQ.2)SR=3.29235
IF(ORDER.EQ.3)SR=1.55690
IF(ORDER.EQ.4)SR=1.18280
IF(ORDER.EQ.5)SR=1.06594
IF(ORDER.EQ.6)SR=1.02460

```

```

NEW05050
NEW05060
NEW05070
NEW05080
NEW05090
NEW05100
NEW05110
NEW05120
NEW05130
NEW05140
NEW05150
NEW05160
NEW05170
NEW05180
NEW05190
NEW05200
NEW05210
NEW05220
NEW05230
NEW05240
NEW05250
NEW05260
NEW05270
NEW05280
NEW05290
NEW05300
NEW05310
NEW05320
NEW05330
NEW05340
NEW05350
NEW05360
NEW05370
NEW05380
NEW05390
NEW05400
NEW05410
NEW05420
NEW05430
NEW05440
NEW05450
NEW05460
NEW05470
NEW05480
NEW05490
NEW05500
NEW05510
NEW05520
NEW05530
NEW05540
NEW05550
NEW05560
NEW05570
NEW05580
NEW05590
NEW05600
NEW05610
NEW05620
NEW05630
NEW05640
NEW05650
NEW05660
NEW05670
NEW05680
NEW05690
NEW05700
NEW05710
NEW05720
NEW05730
NEW05740
NEW05750
NEW05760

```

11

FILE DECADD FORTRAN A

```

ENDIF
IF(NATTN.EQ.40)THEN
  IF(ORDER.EQ.2)SR=5.76107
  IF(ORDER.EQ.3)SR=2.13923
  IF(ORDER.EQ.4)SR=1.40842
  IF(ORDER.EQ.5)SR=1.16811
  IF(ORDER.EQ.6)SR=1.07316
ENDIF
IF(NATTN.EQ.50)THEN
  IF(ORDER.EQ.2)SR=10.19181
  IF(ORDER.EQ.3)SR=3.04137
  IF(ORDER.EQ.4)SR=1.75285
  IF(ORDER.EQ.5)SR=1.33243
  IF(ORDER.EQ.6)SR=1.15868
ENDIF
IF(NATTN.EQ.60)THEN
  IF(ORDER.EQ.2)SR=18.09398
  IF(ORDER.EQ.3)SR=4.39729
  IF(ORDER.EQ.4)SR=2.24440
  IF(ORDER.EQ.5)SR=1.56860
  IF(ORDER.EQ.6)SR=1.28693
ENDIF
IF(NATTN.EQ.70)THEN
  IF(ORDER.EQ.2)SR=32.15951
  IF(ORDER.EQ.3)SR=6.40894
  IF(ORDER.EQ.4)SR=2.92363
  IF(ORDER.EQ.5)SR=1.88906
  IF(ORDER.EQ.6)SR=1.46330
ENDIF
ENDIF
IF(ASUBP.EQ.3.0)THEN
  IF(NATTN.EQ.20)THEN
    IF(ORDER.EQ.2)SR=1.73915
    IF(ORDER.EQ.3)SR=1.15516
    IF(ORDER.EQ.4)SR=1.03853
    IF(ORDER.EQ.5)SR=1.00996
    IF(ORDER.EQ.6)SR=1.00260
  ENDIF
  IF(NATTN.EQ.30)THEN
    IF(ORDER.EQ.2)SR=2.20352
    IF(ORDER.EQ.3)SR=1.45814
    IF(ORDER.EQ.4)SR=1.14542
    IF(ORDER.EQ.5)SR=1.05020
    IF(ORDER.EQ.6)SR=1.01783
  ENDIF
  IF(NATTN.EQ.40)THEN
    IF(ORDER.EQ.2)SR=5.05584
    IF(ORDER.EQ.3)SR=1.98022
    IF(ORDER.EQ.4)SR=1.34663
    IF(ORDER.EQ.5)SR=1.13934
    IF(ORDER.EQ.6)SR=1.05892
  ENDIF
  IF(NATTN.EQ.50)THEN
    IF(ORDER.EQ.2)SR=8.93009
    IF(ORDER.EQ.3)SR=2.79862
    IF(ORDER.EQ.4)SR=1.66148
    IF(ORDER.EQ.5)SR=1.28850
    IF(ORDER.EQ.6)SR=1.13534
  ENDIF
  IF(NATTN.EQ.60)THEN
    IF(ORDER.EQ.2)SR=15.84610
    IF(ORDER.EQ.3)SR=4.03471
    IF(ORDER.EQ.4)SR=2.11596
    IF(ORDER.EQ.5)SR=1.50711
    IF(ORDER.EQ.6)SR=1.25325
  ENDIF
  IF(NATTN.EQ.70)THEN
    IF(ORDER.EQ.2)SR=28.15950
    IF(ORDER.EQ.3)SR=5.87251
    IF(ORDER.EQ.4)SR=2.74749
    IF(ORDER.EQ.5)SR=1.80680
    IF(ORDER.EQ.6)SR=1.41799
  ENDIF
ENDIF
NEW05770
NEW05780
NEW05790
NEW05800
NEW05810
NEW05820
NEW05830
NEW05840
NEW05850
NEW05860
NEW05870
NEW05880
NEW05890
NEW05900
NEW05910
NEW05920
NEW05930
NEW05940
NEW05950
NEW05960
NEW05970
NEW05980
NEW05990
NEW06000
NEW06010
NEW06020
NEW06030
NEW06040
NEW06050
NEW06060
NEW06070
NEW06080
NEW06090
NEW06100
NEW06110
NEW06120
NEW06130
NEW06140
NEW06150
NEW06160
NEW06170
NEW06180
NEW06190
NEW06200
NEW06210
NEW06220
NEW06230
NEW06240
NEW06250
NEW06260
NEW06270
NEW06280
NEW06290
NEW06300
NEW06310
NEW06320
NEW06330
NEW06340
NEW06350
NEW06360
NEW06370
NEW06380
NEW06390
NEW06400
NEW06410
NEW06420
NEW06430
NEW06440
NEW06450
NEW06460
NEW06470
NEW06480

```

FILE: DFCADD FORTRAN A

```

      ENDIF
      ENDIF
      SPI=3.14159
      RK=1.0/SR
      N=ORDER
      EN=N
      KPR=SQRT(1.0-RK**2)
      Q0=0.5*(1.0-SQRT(KPR))/(1.0+SQRT(KPR))
      Q=Q0+2.0*Q0**5+15.0*Q0**9+150.0*Q0**13
      D=(10.0*(0.1*ASUBA)-1.0)/(10.0*(0.1*ASUBP)-1.0)
      LAM=(1.0/(2.0*EN))*ALOG((10.0*(0.05*ASUBP)+1.0)/(10.0*(0.05
C      *ASUBP)-1.0))
      ISTOP=0
      DENSUM=0.0
      ENSUM=SINH(LAM)
      M=1
17    EM=M
      ENUM=(((-1.0)**M)*(Q**M*(M+1)))*SINH((2.0*EM+1.0)*LAM)
      DEN=2.0*(((-1.0)**M)*(Q**M*(M+2)))*COSH(2.0*EM*LAM)
      ENSUM=ENSUM+ENUM
      DENSUM=DENSUM+DEN
      IF(M.GE.2)THEN
        RN=ABS(ENUM/ENSUM)
        RD=ABS(DEN/(1.0+DENSUM))
        IF((RN.LE.1.0E-8).AND.(RD.LE.1.0E-8))THEN
          ISTOP=1
        ENDIF
      ENDIF
      M=M+1
      IF(ISTOP.EQ.0)THEN
        GO TO 17
      ENDIF
      SIGMA0=ABS((2.0*(Q**0.25)*ENSUM)/(1.0+DENSUM))
      W=SQRT((1.0+RK*(SIGMA0**2))*(1.0+(SIGMA0**2/RK)))
      IN=MOD(N,2)
      IF(IN.EQ.0)THEN
        IR=N/2
      ELSE
        IR=(N-1)/2
      ENDIF
      DO 18 I=1,IR
        IF(IN.EQ.0)THEN
          EMU=I-0.5
        ELSE
          EMU=I
        ENDIF
        ISTOP=0
        DENSUM=0.0
        ENSUM=SIN(PI*EMU/EN)
        M=1
19    EM=M
        ENUM=(((-1.0)**M)*(Q**M*(M+1)))*SIN((2.0*EM+1.0)*PI*EMU/EN)
        DEN=2.0*(((-1.0)**M)*(Q**M*(M+2)))*COS(2.0*EM*PI*EMU/EN)
        ENSUM=ENSUM+ENUM
        DENSUM=DENSUM+DEN
        IF(M.GE.2)THEN
          RN=ABS(ENUM/ENSUM)
          RD=ABS(DEN/(1.0+DENSUM))
          IF((RN.LE.1.0E-8).AND.(RD.LE.1.0E-8))THEN
            ISTOP=1
          ENDIF
        ENDIF
        M=M+1
        IF(ISTOP.EQ.0)THEN
          GO TO 19
        ENDIF
        OMEGA(I)=(2.0*(Q**0.25)*ENSUM)/(1.0+DENSUM)
        V(I)=SQRT((1.0-RK*OMEGA(I)**2)*(1.0-(OMEGA(I)**2/RK)))
        AD(I)=1.0/(OMEGA(I)**2)
        DN=1.0+(SIGMA0*OMEGA(I))**2
        BO(I)=((SIGMA0*V(I))**2+(OMEGA(I)*W)**2)/(DN**2)
        B1(I)=(2.0*SIGMA0*V(I))/DN
      ENDIF

```

NEW06490
 NEW06500
 NEW06510
 NEW06520
 NEW06530
 NEW06540
 NEW06550
 NEW06560
 NEW06570
 NEW06580
 NEW06590
 NEW06600
 NEW06610
 NEW06620
 NEW06630
 NEW06640
 NEW06650
 NEW06660
 NEW06670
 NEW06680
 NEW06690
 NEW06700
 NEW06710
 NEW06720
 NEW06730
 NEW06740
 NEW06750
 NEW06760
 NEW06770
 NEW06780
 NEW06790
 NEW06800
 NEW06810
 NEW06820
 NEW06830
 NEW06840
 NEW06850
 NEW06860
 NEW06870
 NEW06880
 NEW06890
 NEW06900
 NEW06910
 NEW06920
 NEW06930
 NEW06940
 NEW06950
 NEW06960
 NEW06970
 NEW06980
 NEW06990
 NEW07000
 NEW07010
 NEW07020
 NEW07030
 NEW07040
 NEW07050
 NEW07060
 NEW07070
 NEW07080
 NEW07090
 NEW07100
 NEW07110
 NEW07120
 NEW07130
 NEW07140
 NEW07150
 NEW07160
 NEW07170
 NEW07180
 NEW07190
 NEW07200

FILE: DFCADD FORTRAN A

```

18      CONTINUE
        HO=1.0
        IF(IN.EQ.0)THEN
          DO 20 I=1,IR
            HO=HO*BO(I)/AO(I)
20      CONTINUE
        HO=HO*(10.0**(-0.05*ASUBP))
        ELSE
          DO 21 I=1,IR
            HO=HO*BO(I)/AO(I)
21      CONTINUE
        HO=HO*SIGMAO
      ENDIF
      R=DBLE(SR)
      HO=DBLE(HO)
      AO1=DBLE(AO(1))
      BO1=DBLE(BO(1))
      B11=DBLE(B1(1))
      IF((ORDER.EQ.2).OR.(ORDER.EQ.3))THEN
        APROT(1,0)=HO*AO1
        APROT(1,1)=0.0D+00
        APROT(1,2)=HO
        APROT(1,3)=0.0D+00
        IF(ORDER.EQ.2)THEN
          APROT(2,0)=BO1
          APROT(2,1)=B11
          APROT(2,2)=1.0D+00
        ENDIF
        IF(ORDER.EQ.3)THEN
          SO=DBLE(SIGMAO)
          APROT(2,0)=SO*BO1
          APROT(2,1)=BO1+(B11*SO)
          APROT(2,2)=SO+B11
          APROT(2,3)=1.0D+00
        ENDIF
      ENDIF
      IF((ORDER.EQ.4).OR.(ORDER.EQ.5))THEN
        AO2=DBLE(AO(2))
        BO2=DBLE(BO(2))
        B12=DBLE(B1(2))
        APROT(1,0)=HO*AO1*AO2
        APROT(1,1)=0.0D+00
        APROT(1,2)=HO*(AO1+AO2)
        APROT(1,3)=0.0D+00
        APROT(1,4)=HO
        APROT(1,5)=0.0D+00
        IF(ORDER.EQ.4)THEN
          APROT(2,0)=BO1*BO2
          APROT(2,1)=(B11*BO2)+(BO1*B12)
          APROT(2,2)=BO1+BO2+(B11*B12)
          APROT(2,3)=B11+B12
          APROT(2,4)=1.0D+00
        ENDIF
        IF(ORDER.EQ.5)THEN
          SO=DBLE(SIGMAO)
          APROT(2,0)=SO*BO1*BO2
          APROT(2,1)=(SO*((B11*BO2)+(BO1*B12)))+(BO1*BO2)
          APROT(2,2)=(SO*(BO1+BO2+(B11*B12)))+(B11*BO2)+(BO1*B12)
          APROT(2,3)=(SO*(B11+B12))+BO1+BO2+(B11*B12)
          APROT(2,4)=SO+B11+B12
          APROT(2,5)=1.0D+00
        ENDIF
      ENDIF
      IF((ORDER.EQ.6).OR.(ORDER.EQ.7))THEN
        AO2=DBLE(AO(2))
        AO3=DBLE(AO(3))
        BO2=DBLE(BO(2))
        BO3=DBLE(BO(3))
        B12=DBLE(B1(2))
        B13=DBLE(B1(3))
        APROT(1,0)=HO*AO1*AO2*AO3
        APROT(1,1)=0.0D+00

```

NEW07210
 NEW07220
 NEW07230
 NEW07240
 NEW07250
 NEW07260
 NEW07270
 NEW07280
 NEW07290
 NEW07300
 NEW07310
 NEW07320
 NEW07330
 NEW07340
 NEW07350
 NEW07360
 NEW07370
 NEW07380
 NEW07390
 NEW07400
 NEW07410
 NEW07420
 NEW07430
 NEW07440
 NEW07450
 NEW07460
 NEW07470
 NEW07480
 NEW07490
 NEW07500
 NEW07510
 NEW07520
 NEW07530
 NEW07540
 NEW07550
 NEW07560
 NEW07570
 NEW07580
 NEW07590
 NEW07600
 NEW07610
 NEW07620
 NEW07630
 NEW07640
 NEW07650
 NEW07660
 NEW07670
 NEW07680
 NEW07690
 NEW07700
 NEW07710
 NEW07720
 NEW07730
 NEW07740
 NEW07750
 NEW07760
 NEW07770
 NEW07780
 NEW07790
 NEW07800
 NEW07810
 NEW07820
 NEW07830
 NEW07840
 NEW07850
 NEW07860
 NEW07870
 NEW07880
 NEW07890
 NEW07900
 NEW07910
 NEW07920

11

FILE: DFCADD FORTRAN A

```

APROT(1,2)=H0*((A02*A03)+(A01*(A02+A03)))
APROT(1,3)=0.0D+00
APROT(1,4)=H0*(A01+A02+A03)
APROT(1,5)=0.0D+00
APROT(1,6)=H0
IF((ORDER.EQ.6).OR.(ORDER.EQ.7))THEN
  APROT(2,0)=B01*B02*B03
  APROT(2,1)=(B11*B02*B03)+(B01*((B02*B13)+(B12*B03)))
  APROT(2,2)=(B02*B03)+(B11*((B02*B13)+(B12*B03)))+(
C      (B01*((B02+B03)+(B12*B13)))
  APROT(2,3)=B01*(B12+B13)+(B11*((B02+B03)+(B12*B13)))+(
C      ((B02*B13)+(B12*B03)))
  APROT(2,4)=B01+(B11*(B12+B13))+((B02+B03)+(B12*B13))
  APROT(2,5)=B11+B12+B13
  APROT(2,6)=1.0D+00
ENDIF
IF(ORDER.EQ.7)THEN
  S0=DBLE(SIGMA0)
  APROT(2,7)=1.0D+00
  APROT(2,6)=(S0*APROT(2,6))+APROT(2,5)
  APROT(2,5)=(S0*APROT(2,5))+APROT(2,4)
  APROT(2,4)=(S0*APROT(2,4))+APROT(2,3)
  APROT(2,3)=(S0*APROT(2,3))+APROT(2,2)
  APROT(2,2)=(S0*APROT(2,2))+APROT(2,1)
  APROT(2,1)=(S0*APROT(2,1))+APROT(2,0)
  APROT(2,0)=S0*APROT(2,0)
ENDIF
ENDIF
E'NDIF
C NORMALIZE THE ELLIPTIC PROTOTYPES
IF(TYPE.EQ.5)THEN
  DO 25 I=1,2
    DO 26 J=0,ORDER
      APROT(I,J)=APROT(I,J)/((DSQRT(R))*J)
26    CONTINUE
25  CONTINUE
ENDIF
C APROT(TYPE=6) ALLOWS THE INPUT OF ANY TYPE OF PROTOTYPE FILTER
C BY INPUTTING THE COEFFICIENTS OF S
IF(TYPE.EQ.6)THEN
  WRITE(6,*)
  WRITE(6,*) 'THIS OPTION ALLOWS YOU TO USE ANY ANALOG '
  WRITE(6,*) ' PROTOTYPE OF ORDER 1-10 '
  WRITE(6,*) '** COEFFICIENTS MUST BE IN DOUBLE PRECISION '
  WRITE(6,*) ' FORMAT (0.123456789D+00) ** '
  WRITE(6,*)
  WRITE(6,*) 'INPUT THE NUMERATOR COEFFICIENTS FIRST'
  DO 50 I=0,ORDER
    WRITE(6,*) 'WHAT IS THE COEFFICIENT OF S**',I,'?'
    READ(5,*) APROT(1,I)
50  CONTINUE
    WRITE(6,*)
    CALL EXCMS ('CLRSCRN')
    WRITE(6,*) 'NOW INPUT THE DENOMINATOR COEFFICIENTS'
    WRITE(6,*)
    DO 51 J=0,ORDER
      WRITE(6,*) 'WHAT IS THE COEFFICIENT OF S**',J,'?'
      READ(5,*) APROT(2,J)
51  CONTINUE
    ENDIF
    C *****
    C *****
    C DEFINE ELEMENTS OF THE BILINEAR TRANSFORMATION MATRIX
    C *****
    C *****
    C FOR ALL THE BILINEAR TRANSFORMATION MATRICES, THE FIRST COLUMN
    C HAS ALTERNATING +1'S AND -1'S. THE LAST COLUMN HAS ALL +1'S
    DO 60 I=1,ORDER
      DO 61 J=0,I
        BILINR(I,J,0)=(-1.0)**J
        BILINR(I,J,I)=1.0
61    CONTINUE
60  CONTINUE

```

11

FILE: DFCADD FOR:RAN A

```

C FIRST ORDER MATRIX HAS ALREADY BEEN LOADED CONTINUE INTERATION NEW08650
C UNTIL YOU HAVE THE APPROPRIATE MATRIX NEW08660
C DO 75 K=2,ORDER NEW08670
  KM1=K-1 NEW08680
C LOAD ELEMENTS IN INTERIOR COLUMNS, ALL ROWS EXCEPT LAST ONE NEW08690
  DO 76 L=0,KM1 NEW08700
    DO 77 M=1,KM1 NEW08710
      MM1=M-1 NEW08720
      BILINR(K,L,M)=BILINR(KM1,L,M)+BILINR(KM1,L,MM1) NEW08730
77 CONTINUE NEW08740
76 CONTINUE NEW08750
C NOW LOAD THE LAST ROW NEW08760
  DO 78 N=1,KM1 NEW08770
    BILINR(K,K,N)=BILINR(K,0,N)*((-1.0D+00)**(K+N)) NEW08780
78 CONTINUE NEW08790
75 CONTINUE NEW08800
C ***** NEW08810
C THE FOLLOWING CLEARS DPROT PRIOR TO LOADING NEW08820
C DO 80 K=1,2 NEW08830
  DO 81 L=0,10 NEW08840
    DPROT(K,L)=0.0 NEW08850
81 CONTINUE NEW08860
80 CONTINUE NEW08870
C ***** NEW08880
C DIGITIZE THE ANALOG PROTOTYPE USING THE BILINEAR TRANSFORMATION NEW08890
C ***** NEW08900
C DO 100 N=1,2 NEW08910
  DO 101 J=0,ORDER NEW08920
    SUMR=0.0 NEW08930
    DO 102 I=0,ORDER NEW08940
      TEMP=APROT(N,I)*BILINR(ORDER,I,J) NEW08950
      SUMR=SUMR+TEMP NEW08960
102 CONTINUE NEW08970
      DPROT(N,J)=SUMR NEW08980
101 CONTINUE NEW08990
100 CONTINUE NEW09000
C ***** NEW09010
C OPTION TO SHOW DIGITAL PROTOTYPE COEFFICIENTS NEW09020
C CALL EXCMS ('CLRSCRN') NEW09030
  WRITE(6,*) 'THE DIGITAL PROTOTYPE HAS BEEN COMPUTED. DO YOU' NEW09040
  WRITE(6,*) 'WISH TO CHECK THE PROTOTYPE COEFFICIENTS?' NEW09050
  WRITE(6,*) ' ** 1=YES/2=NO **' NEW09060
  READ(5,*) LOOK NEW09070
  IF(LOOK.EQ.1)THEN NEW09080
    MAKE DENOMINATOR COEFFICIENT OF Z**ORDER = 1.0 NEW09090
    DO 105 K=1,2 NEW09100
      DO 106 L=0,ORDER NEW09110
        DPROT(K,L)=DPROT(K,L)/DPROT(2,ORDER) NEW09120
106 CONTINUE NEW09130
105 CONTINUE NEW09140
      CALL EXCMS ('CLRSCRN') NEW09150
      WRITE(6,*) 'THE DIGITAL PROTOTYPE NUMERATOR COEFFICIENTS ARE:' NEW09160
      DO 111 N=ORDER,0,-1 NEW09170
        WRITE(6,902) DPROT(1,N),N NEW09180
111 CONTINUE NEW09190
        WRITE(6,*) NEW09200
        WRITE(6,*) 'THE DIGITAL PROTOTYPE DENOMINATOR COEFFICIENTS ARE:' NEW09210
        DO 112 M=ORDER,0,-1 NEW09220
          WRITE(6,902) DPROT(2,M),M NEW09230
112 CONTINUE NEW09240
          WRITE(6,*) NEW09250
          WRITE(6,*) '** 1 TO CONTINUE **' NEW09260
          READ(5,*) LOOK NEW09270
        ENDDIF NEW09280
C ***** NEW09290
C NOW DO THE DIGITAL TO DIGITAL TRANSFORMATIONS NEW09300
C ***** NEW09310
C IF((PASS.EQ.1).OR.(PASS.EQ.2))THEN NEW09320
  DEFINE ELEMENTS OF THE LOWPASS AND HIGHPASS DIGITAL-DIGITAL NEW09330
  TRANSFORMATION MATRIX NEW09340
  DEFINE THE ELEMENTS FOR FIRST AND LAST COLUMNS NEW09350
  NEW09360

```

11

FILE: DFCADD FORTRAN A

```

DO 170 I=1,10
DO 171 J=0,I
LPHPTR(I,J,0)=((-1.0)*ALPHA)**J
IMJ=I-J
LPHPTR(I,IMJ,I)=LPHPTR(I,J,0)
171 CONTINUE
170 CONTINUE
C FIRST ORDER MATRIX HAS BEEN DEFINED ABOVE
C SECOND TO TENTH ORDER BELOW
DO 175 K=2,10
KM1=K-1
DO 176 L=0,KM1
DO 177 M=1,KM1
MM1=M-1
LPHPTR(K,L,M)=LPHPTR(KM1,L,M)+((LPHPTR(KM1,L,MM1))*(-1.0D+00)
C*ALPHA)
177 CONTINUE
176 CONTINUE
DO 178 N=1,KM1
KMN=K-N
LPHPTR(K,K,N)=LPHPTR(K,0,KMN)
178 CONTINUE
175 CONTINUE
C *****
C CALCULATE THE FINAL FILTER
C FOR HIGHPASS, THE PROTOTYPE MUST SUBSTITUTE (-Z) FOR (Z)
C IF(PASS.EQ.2)THEN
DO 180 K=1,2
DO 181 L=0,ORDER
DPROT(K,L)=DPROT(K,L)*((-1.0)**L)
181 CONTINUE
180 CONTINUE
ENDIF
DO 185 K=1,2
DO 186 L=0,ORDER
SUMR=0.0D+00
DO 187 M=0,ORDER
TEMP=DPROT(K,M)*LPHPTR(ORDER,M,L)
SUMR=SUMR+TEMP
187 CONTINUE
DFLTR(K,L)=SUMR
186 CONTINUE
185 CONTINUE
ENDIF
C *****
C BANDPASS AND BANDSTOP FORMULAS
C IF((PASS.EQ.3).OR.(PASS.EQ.4))THEN
C LOAD THE BANDPASS/BANDSTOP TRANSFORMATION MATRIX
C FIRST LOAD THE FIRST AND LAST COLUMNS
DO 225 I=1,10
DO 226 J=0,I
BPPBSTR(I,J,0)=C**J
I2=2*I
IMJ=I-J
BPPBSTR(I,IMJ,I2)=BPPBSTR(I,J,0)
226 CONTINUE
225 CONTINUE
C LOAD THE REST OF THE FIRST ORDER MATRIX
BPPBSTR(1,0,1)=(-1.0D+00)*B
BPPBSTR(1,1,1)=(-1.0D+00)*B
C LOAD THE 2ND-10TH ORDER TRANSFORMATION MATRICES
DO 230 K=2,10
K2=2*K
K2M1=K2-1
KM1=K-1
K2M2=K2-2
KM12=2*KM1
KM12M1=KM12-1
C LOAD THE SECOND AND SECOND TO LAST COLUMNS
DO 231 L=0,KM1
BPPBSTR(K,L,1)=BPPBSTR(KM1,L,1)+(BPPBSTR(KM1,L,0))*(-1.0D+00)*B
BPPBSTR(K,0,K2M1)=(BPPBSTR(KM1,0,KM12))*(-1.0D+00)*B+

```

NEW09370
NEW09380
NEW09390
NEW09400
NEW09410
NEW09420
NEW09430
NEW09440
NEW09450
NEW09460
NEW09470
NEW09480
NEW09490
NEW09500
NEW09510
NEW09520
NEW09530
NEW09540
NEW09550
NEW09560
NEW09570
NEW09580
NEW09590
NEW09600
NEW09610
NEW09620
NEW09630
NEW09640
NEW09650
NEW09660
NEW09670
NEW09680
NEW09690
NEW09700
NEW09710
NEW09720
NEW09730
NEW09740
NEW09750
NEW09760
NEW09770
NEW09780
NEW09790
NEW09800
NEW09810
NEW09820
NEW09830
NEW09840
NEW09850
NEW09860
NEW09870
NEW09880
NEW09890
NEW09900
NEW09910
NEW09920
NEW09930
NEW09940
NEW09950
NEW09960
NEW09970
NEW09980
NEW09990
NEW10000
NEW10010
NEW10020
NEW10030
NEW10040
NEW10050
NEW10060
NEW10070
NEW10080

11

FILE: DFCADD FORTRAN A

```

C      (BPBSTR(KM1,0,KM12M1))*C)
KML=K-L
BPBSTR(K,KML,K2M1)=BPBSTR(K,L,1)
231  CONTINUE
C      NOW LOAD INTERIOR ELEMENTS EXCEPT FOR LAST ROW
DO 236 L=0,KM1
  DO 237 M=2,K2M2
    MM1=M-1
    MM2=M-2
    BPBSTR(K,L,M)=BPBSTR(KM1,L,M)+(BPBSTR(KM1,L,MM1)*(-1.0D+00)*
237  C      B)+(BPBSTR(KM1,L,MM2)*C)
236  CONTINUE
C      NOW FILL THE LAST ROW
DO 238 N=1,K2M1
  K2MN=K2-N
  BPBSTR(K,K,N)=BPBSTR(K,0,N)
238  CONTINUE
230  CONTINUE
C      *****
C      CALCULATE THE FINAL FILTER
C      FOR THE BANDPASS CASE REPLACE (Z) WITH (-Z)
IF((PASS.EQ.3) THEN
  DO 240 M=1,2
    DO 241 N=0,ORDER
      DPROT(M,N)=DPROT(M,N)*((-1.0D+00)**N)
241  CONTINUE
240  CONTINUE
  ENDDIF
  ORDER2=2*ORDER
  DO 245 K=1,2
    DO 246 L=0,ORDER2
      SUMR=0.0D+00
      DO 247 M=0,ORDER
        TEMP=DPROT(K,M)*BPBSTR(ORDER,M,L)
        SUMR=SUMR+TEMP
247  CONTINUE
      DFLTR(K,L)=SUMR
246  CONTINUE
245  CONTINUE
  ENDDIF
C      *****
C      OUTPUT OPTIONS
C      *****
IF((PASS.EQ.3).OR.(PASS.EQ.4)) THEN
  END=2*ORDER
  ENDDIF
IF((PASS.EQ.1).OR.(PASS.EQ.2)) THEN
  END=ORDER
  ENDDIF
C      MAKE DENOMINATOR COEFFICIENT OF Z**END EQUAL TO 1
DO 250 M=1,2
  DO 251 N=0,END
    DFLTR(M,N)=DFLTR(M,N)/DFLTR(2,END)
251  CONTINUE
250  CONTINUE
  CALL EXCMS ('CLRSRN')
  WRITE(6,*)
  WRITE(6,*) 'THE FINAL DIGITAL FILTER IS:'
  WRITE(6,*)
  WRITE(6,*) '    NUMERATOR COEFFICIENTS:'
  DO 300 I=END,0,-1
    WRITE(6,902) DFLTR(1,I),I
300  CONTINUE
  WRITE(6,*) '    DENOMINATOR COEFFICIENTS:'
  DO 301 J=END,0,-1
    WRITE(6,902) DFLTR(2,J),J
301  CONTINUE
  WRITE(6,*)
  WRITE(6,*) ' ** ENTER 1 TO CONTINUE **'
  READ(5,*) LOOK
C      *****

```

NEW10090
 NEW10100
 NEW10110
 NEW10120
 NEW10130
 NEW10140
 NEW10150
 NEW10160
 NEW10170
 NEW10180
 NEW10190
 NEW10200
 NEW10210
 NEW10220
 NEW10230
 NEW10240
 NEW10250
 NEW10260
 NEW10270
 NEW10280
 NEW10290
 NEW10300
 NEW10310
 NEW10320
 NEW10330
 NEW10340
 NEW10350
 NEW10360
 NEW10370
 NEW10380
 NEW10390
 NEW10400
 NEW10410
 NEW10420
 NEW10430
 NEW10440
 NEW10450
 NEW10460
 NEW10470
 NEW10480
 NEW10490
 NEW10500
 NEW10510
 NEW10520
 NEW10530
 NEW10540
 NEW10550
 NEW10560
 NEW10570
 NEW10580
 NEW10590
 NEW10600
 NEW10610
 NEW10620
 NEW10630
 NEW10640
 NEW10650
 NEW10660
 NEW10670
 NEW10680
 NEW10690
 NEW10700
 NEW10710
 NEW10720
 NEW10730
 NEW10740
 NEW10750
 NEW10760
 NEW10770
 NEW10780
 NEW10790
 NEW10800

11

FILE: DFCADD FORTRAN A

```

C      OPTION TO SPOT CHECK FILTER OUTPUT MAGNITUDE AND FREQUENCY      NEW10810
CALL EXCMS ('CLRSCRN')                                              NEW10820
WRITE(6,*) 'THE TRANSFER FUNCTION FOR THIS FILTER HAS BEEN'        NEW10830
WRITE(6,*) 'COMPUTED. DO YOU WANT TO SPOT CHECK THE MAGNITUDE?'    NEW10840
WRITE(6,*) 'AND PHASE OF THE FILTER OUTPUT FOR A GIVEN '           NEW10850
WRITE(6,*) 'FREQUENCY?'                                             NEW10860
WRITE(6,*) '          ** 1-YES/2-NO ** '                             NEW10870
READ(5,*) LOOK                                                       NEW10880
IF(LOOK.EQ.1)THEN                                                    NEW10890
310  CALL EXCMS ('CLRSCRN')                                          NEW10900
      IF(AFORDF.EQ.1)THEN                                            NEW10910
        WRITE(6,*) 'WHAT ANALOG FREQUENCY(IN KHZ)?'                 NEW10920
        WRITE(6,*) 'DO YOU WANT TO CHECK?'                          NEW10930
        WRITE(6,*) '          ** DECIMAL **'                       NEW10940
        READ(5,*) F                                                  NEW10950
        DF=(F/SSFREQ)*2.0*SPI                                       NEW10960
      ENDIF                                                         NEW10970
      IF(AFORDF.EQ.2)THEN                                           NEW10980
        WRITE(6,*) 'WHAT DIGITAL FREQUENCY DO YOU WANT TO CHECK?' NEW10990
        WRITE(6,*) '          ** DECIMAL **'                       NEW11000
        READ(5,*) DF                                               NEW11010
      ENDIF                                                         NEW11020
      X=COS(DF)                                                     NEW11030
      Y=SIN(DF)                                                     NEW11040
      Z=CMPLX(X,Y)                                                  NEW11050
      XN=REAL(DFLTR(1,0))                                          NEW11060
      YN=0.0                                                         NEW11070
      XD=REAL(DFLTR(2,0))                                          NEW11080
      YD=0.0                                                         NEW11090
      HOFZN=CMPLX(XN,YN)                                           NEW11100
      HOFZD=CMPLX(XD,YD)                                           NEW11110
      DO 320 N=1,END                                                NEW11120
        ZN=REAL(DFLTR(1,N))*(Z**N)                                  NEW11130
        ZD=REAL(DFLTR(2,N))*(Z**N)                                  NEW11140
        HOFZN=HOFZN+ZN                                              NEW11150
        HOFZD=HOFZD+ZD                                              NEW11160
320  CONTINUE                                                       NEW11170
      HOFZ=HOFZN/HOFZD                                              NEW11180
      FMAG=SQRT((REAL(HOFZ)**2)+(AIMAG(HOFZ)**2))                  NEW11190
      FMAGDB=20.0*ALOG10(FMAG)                                       NEW11200
      FPHSE=ATAN(AIMAG(HOFZ)/REAL(HOFZ))                            NEW11210
      WRITE(6,*) 'THE FILTER OUTPUT WILL BE:'                      NEW11220
      WRITE(6,914) FMAGDB                                           NEW11230
      WRITE(6,915) FPHSE                                           NEW11240
      WRITE(6,*)                                                    NEW11250
      WRITE(6,*) 'DO YOU WANT TO CHECK ANOTHER FREQUENCY?'        NEW11260
      WRITE(6,*) '          ** 1-YES/2-NO ** '                     NEW11270
      READ(5,*) LOOK                                                NEW11280
      IF(LOOK.EQ.1)GO TO 310                                         NEW11290
    ENDIF                                                            NEW11300
C      *****                                                        NEW11310
C      OPTION TO LOAD FREQUENCY RESPONSE INTO A DATA FILE          NEW11320
CALL EXCMS ('CLRSCRN')                                              NEW11330
WRITE(6,*) 'DO YOU WANT THE FILTERS FREQUENCY RESPONSE '          NEW11340
WRITE(6,*) 'DATA TO BE ENTERED INTO A DATA FILE FOR USE'        NEW11350
WRITE(6,*) 'BY A PLOTTING PROGRAM?'                                NEW11360
WRITE(6,*) 'NOTE: THE FILE NAME WILL BE FILTER, FILE TYPE'        NEW11370
WRITE(6,*) '          WILL BE DATA, 100 POINTS WILL BE CALCULATED' NEW11380
WRITE(6,*) '          FOR A DIGITAL FREQUENCY FROM 0 TO PI, THE FIRST' NEW11390
WRITE(6,*) '          COLUMN WILL BE THE DIGITAL FREQUENCY (EXPRESSED' NEW11400
WRITE(6,*) '          AS A FRACTION OF THE SAMPLING FREQUENCY 0-1/2)' NEW11410
WRITE(6,*) '          THE SECOND COLUMN WILL BE THE MAGNITUDE, AND'  NEW11420
WRITE(6,*) '          THE THIRD COLUMN WILL BE THE PHASE(DEGREES)'  NEW11430
WRITE(6,*) '          ** 1-MAKE DATA FILE/2-DON'T BOTHER **'      NEW11440
READ(5,*) LOOK                                                       NEW11450
IF(LOOK.EQ.1)THEN                                                    NEW11460
  DO 350 I=0,100                                                     NEW11470
    DF=REAL(I)*(5.0/1000.0)                                          NEW11480
    DFR=REAL(I)*(SPI/100.0)                                          NEW11490
    X=COS(DFR)                                                        NEW11500
    Y=SIN(DFR)                                                        NEW11510
    Z=CMPLX(X,Y)                                                      NEW11520
  
```

FILE: DFCADD FORTRAN A

```

      XN=REAL(DFLTR(1,0))
      YN=0.0
      XD=REAL(DFLTR(2,0))
      YD=0.0
      HOFZN=CMPLX(XN,YN)
      HOFZD=CMPLX(XD,YD)
      DO 351 N=1,END
        ZN=REAL(DFLTR(1,N))*(Z**N)
        ZD=REAL(DFLTR(2,N))*(Z**N)
        HOFZN=HOFZN+ZN
        HOFZD=HOFZD+ZD
351   CONTINUE
      HOFZ=HOFZN/HOFZD
      FMAG=SQRT((REAL(HOFZ)**2)+(AIMAG(HOFZ)**2))
      IF (REAL(HOFZ).NE.0.0) THEN
        FPHSER=ATAN(AIMAG(HOFZ)/REAL(HOFZ))
      ELSEIF (REAL(HOFZ).EQ.0.0) THEN
        IF (AIMAG(HOFZ).GT.0.0) FPHSER=1.570796
        IF (AIMAG(HOFZ).LT.0.0) FPHSER=-1.570796
        IF (AIMAG(HOFZ).EQ.0.0) FPHSER=0.0
      ENDIF
      FPHSE=FPHSER*(180.0/SPI)
      WRITE(8,916) DF,FMAG,FPHSE
350   CONTINUE
      ENDIF
C *****
901   FORMAT(5X,F10.5,' S**',I1)
902   FORMAT(5X,F10.5,' Z**',I1)
903   FORMAT(3X,'ALPHA= ',F9.4)
904   FORMAT(3X,'K= ',F9.4)
905   FORMAT(3X,'THE LOWPASS DIGITAL PROTOTYPE FREQ. IS ',F5.3)
914   FORMAT(3X,'MAGNITUDE = ',F8.2,' DB')
915   FORMAT(3X,'PHASE = ',F6.2,' RADIANS')
916   FORMAT(3X,F4.3,10X,F5.3,10X,F6.2)
      STOP
      END

```

NEW11530
 NEW11540
 NEW11550
 NEW11560
 NEW11570
 NEW11580
 NEW11590
 NEW11600
 NEW11610
 NEW11620
 NEW11630
 NEW11640
 NEW11650
 NEW11660
 NEW11670
 NEW11680
 NEW11690
 NEW11700
 NEW11710
 NEW11720
 NEW11730
 NEW11740
 NEW11750
 NEW11760
 NEW11770
 NEW11780
 NEW11790
 NEW11800
 NEW11810
 NEW11820
 NEW11830
 NEW11840
 NEW11850
 NEW11860
 NEW11870
 NEW11880

INITIAL DISTRIBUTION LIST

	No. Copies
1. Defense Technical Information Center Cameron Station Alexandria, Virginia 22304-6145	1
2. Library, Code 0142 Naval Postgraduate School Monterey, California 93943-5002	2
3. Chairman, Code 62 Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943-5000	1
4. Commandant of the Marine Corps Code TE 06 Headquarters, U.S. Marine Corps Washington, DC 20380-0001	1
5. United States Military Academy Department of Electrical Engineering ATTN: MAJ E. Siomacco West Point, New York 10996	1
6. Department of Electrical and Computer Engineering ATTN: Prof. R. Strum, Code 62St Naval Postgraduate School Monterey, California 93943-5000	1
7. Department of Electrical and Computer Engineering ATTN: Prof. J. England, Code 62 Naval Postgraduate School Monterey, California 93943-5000	1
8. Department of Electrical and Computer Engineering ATTN: Prof. P. Moose, Code 62Me Naval Postgraduate School Monterey, California 93943-5000	1
9. Department of Electrical and Computer Engineering ATTN: Prof. D. Kirk, Code 62 Naval Postgraduate School Monterey, California 93943-5000	1