MARTHA User's Manual

Paul Penfield, Jr.

MARTHA

User's Manual

Paul Penfield, Jr.

The MIT Press   Cambridge, Massachusetts, and London, England

This book was designed by the MIT Press Design Department.
It was set in IBM Selectric typewriter
by Williams Graphic Service, Inc.
printed on P & S Offset
by Halliday Lithograph Corp.
and bound by Halliday Lithograph Corp.
in the United States of America.

ISBN 0 262 66015 6 (paperback)
Library of Congress catalog card number: 73-175017

To the real Martha

Contents

(Forms for the convenience of the user are inserted
at the end of this manual)

Modern circuit designers use many sophisticated tools, including the computer. There are many special-purpose and general-purpose programs to analyze electrical networks. No two are exactly alike in their computing algorithms, in their data structure, in the scope of services performed, or in their input-output language.

Originally there were special-purpose programs that could not "talk" to each other except by using humans as interpreters. Thus, for example, a filter designer would use one program to rough out a design, another to analyze it, and possibly still others to perform transient analysis or parameter optimization, or to check on tolerances. Each of these specialized programs required its data in a particular format, and its output was not in a form appropriate for the other programs. Furthermore, these programs often ran on different machines, or used different languages.

The present trend in circuit-design programs is toward a uniform input-output notation that is understood by several programs or subroutines that themselves perform specialized tasks. Among the various programming languages, APL is by far best suited to such an arrangement, and as time goes on will probably supplant FORTRAN, BASIC, and others as the host language for circuit-design programs.

At the very least, an up-to-date system should have the following features:

1. Versatility that is limited by the imagination of the end user, rather than that of the author. This is most easily accomplished by embedding the programs in a general-purpose conversational language such as APL.
2. An open-ended library of special-purpose functions to work with the functions in the system.
3 A uniform language for defining network topology and the constitutive laws of the elements, and ability to program in that language.
4. Several synthesis, analysis, and optimization algorithms.
5. As many user-defined types of elements as needed.
6. A variety of output formats.
7. An implementation that is relatively machine-independent.
8. Facilities for numerically defined functions.
9. Adequate documentation, including on-line summaries.

The author is not aware of any circuit-design system that satisfies all these criteria. At present *MARTHA* fails to satisfy no. 4.

MARTHA was first implemented during the summer of 1970 at M.I.T. Part of the initial implementation was supported by the National

Aeronautics and Space Administration, under Grant NGL22-009-337. The computer time was supplied by M.I.T. and subsequent refinements were not supported, except in minor ways, by any research grant.

Chapter 1

Introduction

MARTHA is a set of general-purpose programs for analyzing linear electrical networks. The networks may be lumped or distributed, reciprocal or nonreciprocal, passive or active. You do not have to be a computer programmer to use *MARTHA*---you need to know only what the network looks like and what information you require.

MARTHA is geared toward "transmission-type" networks, with an input and an output. This includes most filters, amplifiers, and microwave networks, even with multiple feedback paths and branches. The possible output includes two-port parameters like impedance, admittance, and scattering matrices; input and output impedances, admittances, and reflection coefficients; and various types of "gains" including insertion gain, transducer gain, voltage gain, and available gain. These, their real or imaginary parts, their magnitude or phase, may be printed or plotted as functions of frequency or as functions of network parameters, or as functions of each other. You can analyze more than one network at a time.

MARTHA is embedded in the flexible conversational programming language APL. To use *MARTHA* you must have an account with a computer service that supports APL and has *MARTHA* in its library.* If you have never used APL before, Appendix A contains all the information you need to use *MARTHA*.

Because *MARTHA* is embedded in APL, all the features of APL are available if you wish to take advantage of them. In particular, the network topology or various element values can be changed under control of APL programs you write, and the results of calculations done by *MARTHA* are available for further calculations. In other words you may, if you wish, use *MARTHA* without modification as a network analyzer; or you may use the individual functions of *MARTHA* in a more specialized analysis and synthesis system.

*To find out where *MARTHA* is available, write to the Manager of Software Services, The MIT Press, 28 Carleton Street, Cambridge, Mass. 02142.

Chapter 2

How to Use *MARTHA*

2.1 Obtaining the Programs

To use *MARTHA* you must have an account with a computer service that has *MARTHA* in its library. When you set up the account you will be told how to log into the conversational language APL. In general this involves placing a telephone call to the computer, and then requesting APL. In the space below write the telephone number of the computer and the log in procedure, so you won't forget it.

After you have logged into APL, you must still load *MARTHA*. APL programs are stored in "workspaces", and the workspaces associated with *MARTHA* are in a public library, usually (although not always) library 100. You can move the programs into your active workspace by the commands

```
      )CLEAR
      )COPY 100 MARTHA
```

(If the programs are in a library with a number different from 100, use that number instead.)

After you have finished using *MARTHA*, log out of the computer by typing

```
      )OFF
```

If you forget how to use *MARTHA*, a succinct summary is available on-line. You can bring this into your active workspace with the command

```
      )LOAD 100 HOWMARTHA
```

Then type

```
      DESCRIBE
```

for further instructions. Appendix B contains this summary as of the date of this manual.

```
CP/67 VERSION 2.0+33 07/03/71

DIAL APL
 .. CONNECTED ..

)60024
003) 10.29.51 07/12/71 PPENFIELD

  A P L \ 3 6 0

      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED      20.52.47 6/30/71


      )LOAD 100 HOWMARTHA
SAVED 10.17.01 07/12,71

      DESCRIBE


MARTHA    71∘A    1 JULY 1971

MARTHA IS A SET OF FUNCTIONS THAT ANALYZE LINEAR ELECTRICAL NETWORKS,
NORMALLY AS A FUNCTION OF FREQUENCY.  FOR A COMPLETE DESCRIPTION, SEE
PAUL PENFIELD JR., 'MARTHA USER'S MANUAL,' THE MIT PRESS, CAMBRIDGE,
MASS. 02142; 1971.  FOR A SUCCINCT SUMMARY:

      )LOAD 100 HOWMARTHA

THIS WORKSPACE CONTAINS THE FOLLOWING SUMMARIES (BUT NOT THE FUNCTIONS)

DESCRIBE      GENERAL SUMMARY
ELEMENTS      NETWORK ELEMENTS AVAILABLE
WIRING        WIRING FUNCTIONS AVAILABLE
RESPONSES     RESPONSE FUNCTIONS
MODIFIERS     MODIFIERS FOR RESPONSES
FORMATS       FORMS OF OUTPUT
EXTRA         FUNCTIONS TO WORK WITH MARTHA
HINTS
EXAMPLES
CHANGES       SINCE THE USER MANUAL
ERRORS        MARTHA ERROR MESSAGES
ORDERFORM
COMMENTS

TO GET THE FUNCTIONS:

      )COPY 100 MARTHA

MARTHA INCLUDES SEVERAL VARIABLES AND FUNCTIONS THAT THE USER DOES NOT
DIRECTLY USE.  THESE ALL HAVE NAMES CONTAINING TWO UNDERLINED LETTERS.
THERE ARE SIX VARIABLES THAT THE USER CAN CHANGE:

VARIABLE   PRESET TO          MEANING
--------   ---------   ----------------------------------------
F          1  2        FREQUENCY VECTOR, IN HZ
ZG         0           GENERATOR IMPEDANCE IN OHMS
ZL         1E25        LOAD IMPEDANCE IN OHMS
ZN         50          NORMALIZATION IMPEDANCE OF 1-PORT, IN OHMS
ZNIN       50          NORMALIZATION IMPEDANCE AT INPUT, IN OHMS
ZNOUT      50          NORMALIZATION IMPEDANCE AT OUTPUT, IN OHMS

MARTHA INCLUDES 75 OTHER VARIABLES AND FUNCTIONS.  THE MARTHA LIBRARY
CONTAINS OVER 100 ADDITIONAL VARIABLES AND FUNCTIONS IN 5 WORKSPACES:

 WORKSPACE             CONTENTS
-----------  ----------------------------------
100 MARTHAE  ELEMENTS
100 MARTHAW  WIRING FUNCTIONS
100 MARTHAR  RESPONSE FUNCTIONS
100 MARTHAM  MODIFIERS FOR RESPONSE FUNCTIONS
100 MARTHAX  EXTRA FUNCTIONS TO WORK WITH MARTHA

TO GET A SPECIFIC FUNCTION, E.G. VCCS FROM 100 MARTHAE:

      )COPY 100 MARTHAE VCCS

FOR A SUMMARY OF THE CONTENTS OF THE MARTHA LIBRARY, PRINT THE VARIABLE
NAMED 'DESCRIBE' IN EACH WORKSPACE.
```

2.2 *MARTHA* Commands

To use *MARTHA*, you need to specify four things:

1.
Frequency or frequencies to use
2.
Description of the network or networks
3.
List of desired response functions
4.
Choice of printing or plotting the results

After the frequency has been specified, *MARTHA* is run by typing a line of the form

$$\left\{\begin{matrix} \mathit{PRINT} \\ \mathit{PLOT} \\ \mathit{PLOG} \end{matrix}\right\} \quad \texttt{<output list>}\ \mathit{OF}\ \texttt{<network description>}$$

The format of the output is determined by the first word, and the word *OF* is required.

Examples:

```
PRINT Z, Y OF (R 5) P C 300E¯6
PLOT MAG S11, DEG S11 OF FILTER1
```

In the first example the output list is *Z*, *Y* (the impedance and admittance) and the network is (*R* 5) *P C* 300*E*¯6 (a resistor of 5 ohms in parallel with a capacitor of 300 µF). In the second example *MAG S*11, *DEG S*11 is the output list (the scattering-matrix entry S11 expressed in magnitude and phase), and *FILTER*1 is presumably the name of an APL function you have written to describe a network.

2.3 Setting the Frequency

In *MARTHA* the frequency is always denoted by the variable *F*. Initially *F* is set to the vector 1 2, meaning two separate frequencies, 1 Hz and 2 Hz. Normally you will set *F* to something else, and after you do so it

```
      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED    20.52.47 06/30/71

      PRINT Z, Y OF (R 5)P C 300E¯6

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71  10:37
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OP TECHNOLOGY

      F          RE Z         IM Z         RE Y         IM Y
----------- ----------- ----------- ----------- -----------
 1.0000E00   4.9996E00  ¯4.7120E¯02  2.0000E¯01  1.8850E¯03
 2.0000E00   4.9982E50  ¯9.4214E¯02  2.0000E¯01  3.7699E¯03
```

will remain at that new value or values until you reset it to another value. You may change the frequencies as often as you wish, and *MARTHA* will always use the current value of *F*.

The most common frequency specifications are a spot frequency (just one value), a linear sweep, and a logarithmic sweep. For a spot frequency, just set *F* to that value; for example, for 10.5 MHz, type

```
      F←10.5E6
```

Note that in *MARTHA* (as in APL), variables are assigned values by the ← symbol, not by = as in many other programming languages. Also, in *MARTHA* the frequency is always expressed in Hz, not in radians per second or MHz.

A linear sweep can be generated using the index generator ι. (Recall that ι*N* is a vector consisting of the first *N* integers, for example ι5 is 1 2 3 4 5.) For example, to obtain a linear sweep of 20 frequencies starting at 910 Hz and separated by 10 Hz,

```
      F←900+10×ι20
```

A logarithmic sweep can also be generated with the index generator. For example, for 21 frequencies between 100 Hz and 1 MHz

```
      F←100×10★.2×0,ι20
```

Other frequency specifications are possible. For example, a

```
      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED    20.52.47 06/30/71

      ⍝ THIS SYMBOL  ⍝  IS FOR COMMENTS--THE COMPUTER IGNORES ANY LINE TWAT STARTS WITH THAT SYMBOL.

      ⍝ YOU CAN FIND OUT WHAT THE FREQUENCY  F  IS BY MERELY TYPING  F

      F
1 2
      F←10.5E6
      F
10500000

      ι4
1 2 3 4
      ι5
1 2 3 4 5
      F←900+10×ι20
      F
910 920 930 940 950 960 970 980 990 1000 1010 1020 1030 1040 1050 1060 1070 1080 1090 1100

      F←100×10★.2×0,ι20
      F
100  158.4893192  251.1886432  398.1071706  630.9573445  1000  1584.893192  2511.886432  3981.071706  6109.573445
     10000   15848.93192  25118.86432  39810.71706  63095.73445  100000  158489.3192  251188.6402  308107.1706
     630957.3445   1000000

      ⍝ TYPING  ρ  IN FRONT OF  F  WILL TELL YOU HOW MANY FREQUENCIES ARE IN YOUR SWEEP.
      ρF
21

      F←79,81,50+2×ι25
      ρF
27
      F
74  81  52  54  56  58  60  62  64  66  68  70  72  74  76  78  80  82  84  86  88  90  92  94  96  98  100
```

linear frequency sweep with two extra points (presumably of special interest:

```
      F←79,81,50+2×ι25
```

It is not necessary for the frequencies to be arranged in numerical order. You may use negative frequencies if you wish.

In circuit analysis it is often necessary to divide by the frequency (for example in calculating the impedance of a capacitor). Any of your frequencies equal to 0 will be automatically reset to 10^{-25} Hz.

2.4 Describing Networks

MARTHA can analyze either one-port or two-port networks. A two-port network is assumed to be terminated by a generator at the input and a load at the output, as shown in Figure 2.1. The generator and load are not part of the network being defined.

A network is described in *MARTHA* in terms of a sequence of operations used to construct the network from simpler subnetworks, or "sections." In such a construction, the port concept must be valid for each section connected. A port is a pair of terminals of a network with the same current entering one terminal and leaving the other. Thus, any network with only two accessible terminals is automatically a one-port network. However, four-terminal networks do not always behave as two-port networks: for example, consider the two networks of Figure 2.2. It appears that each might be described as a two-port network. However, when wired together as in Figure 2.3, the port concept is not valid because the currents i and i' are not equal. On the other hand, consider the two networks of Figure 2.4. These differ from those in Figure 2.2 only because of the ideal transformers (with turns ratio 1). These transformers ensure that the port concept is valid no matter how the networks are wired. Thus, in Figure 2.5, the port concept is valid.

The analysis done *by MARTHA* is based on the assumption that the port concept is valid. Usually there is no problem, but if the behavior of a network is altered by inserting ideal transformers such as

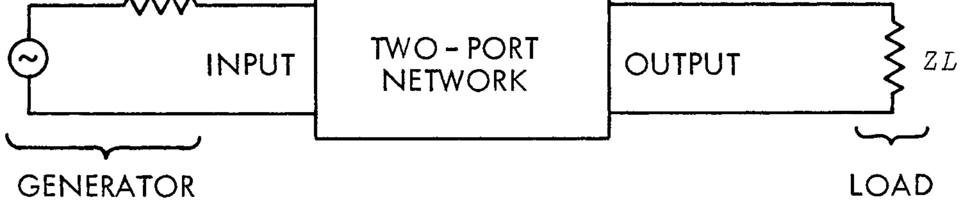


Figure 2.1. Two-port network as terminated by a generator and load.

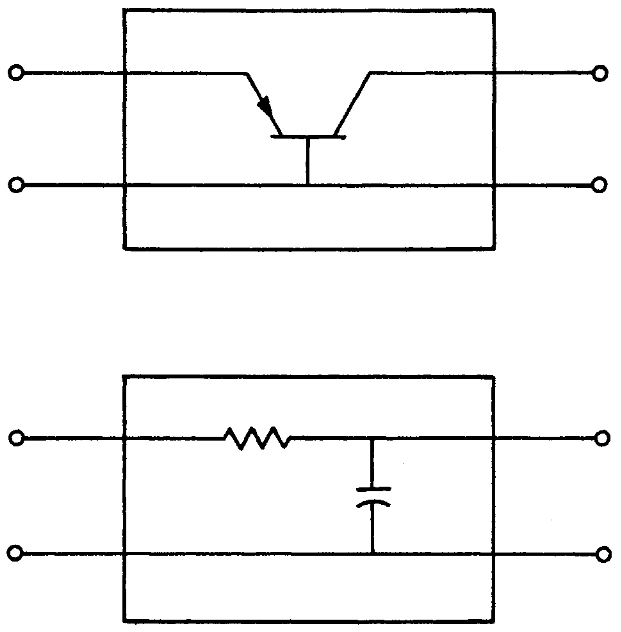


Figure 2.2. Networks that can in many circumstances be ade quately described as two-port networks.

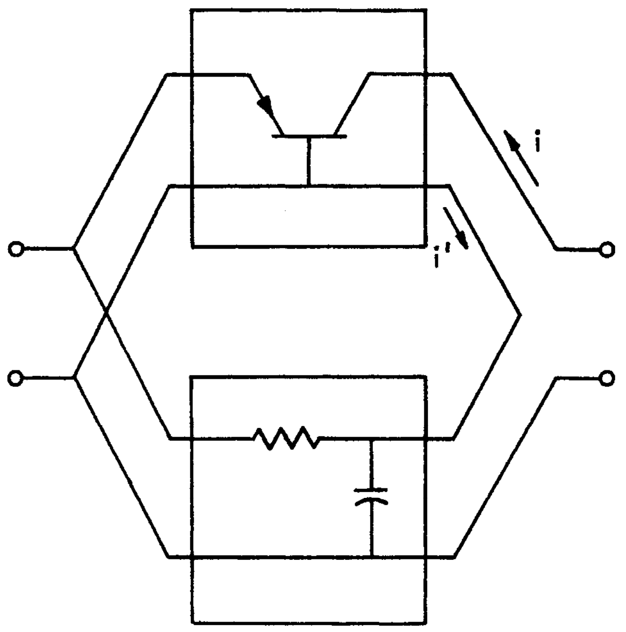


Figure 2.3. Example in which the port concept is not valid. The currents i and i' are not equal.

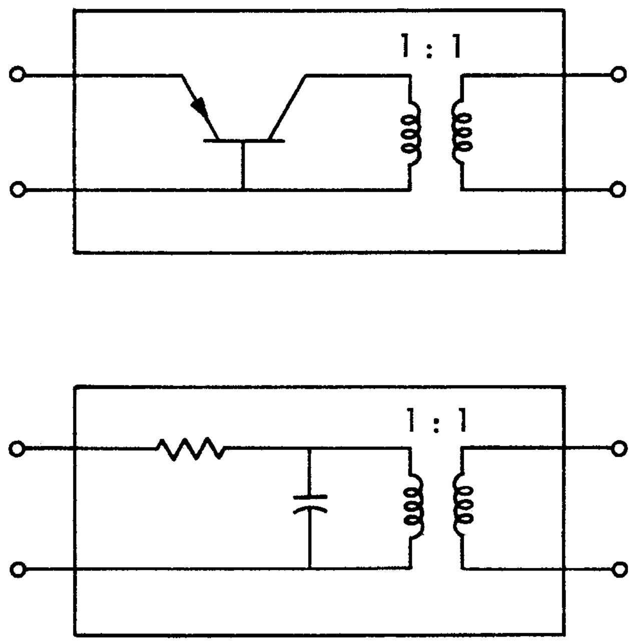


Figure 2.4. Networks that can in all circumstances be adequately described as two-port networks. These differ from the networks in Figure 2.2 because of the 1:1 ideal transformers.

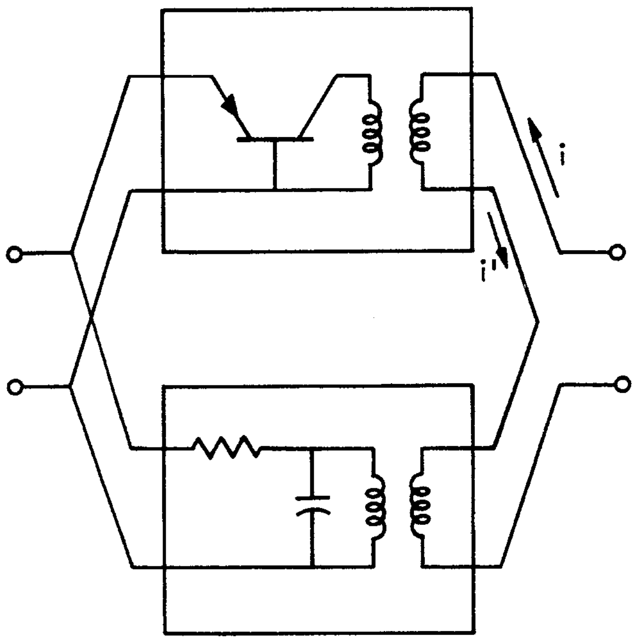


Figure 2.5. Example in which the port concept is valid. The currents i and i' are equal. This network behaves differently from the network in Figure 2.3.

those in Figures 2.4 and 2.5, then the port concept is not valid and *MARTHA* will not work properly. If you really want to analyze the network of Figure 2.3, you can, but the description is different from that implied by Figure 2.3. It is your responsibility as a user to ensure that the port concept is valid for your network: *MARTHA* has no way of checking this.

2.5 Elements

A network consists, ultimately, of elements wired together. Typical elements are resistors, capacitors, inductors, ideal transformers, lengths of transmission line, lengths of waveguide, transistors, operational amplifiers, and so on. You may define the elements of your network at any time and change their parameters at any time. The current definitions will always be used by *MARTHA*. An element need not be redefined when the frequency is changed. You may define more elements than you actually use, if you wish.

You may give names to your elements; for example R1 might be a resistor and C13 a capacitor. *MARTHA* cannot tell, from the name, what type of element it is. You might name one of your resistors *L*3 if you wished, even though this would not be customary electrical-engineering practice.

Elements are defined by one of the element-definition functions *R*, *L*, *C*, *IT*, *OPAMP*, *TEM*, *WG*, *FET*, or *HYBRIDPI*. Each of these has the effect of taking a number or set of numbers and creating an element from it. For example, *R* converts a resistance (in ohms) into a resistor, and *IT* converts a turns ratio into an ideal transformer.

Examples:

```
R1←R 5600
C2←C 3E¯6
L2←L .0015
```

Here a resistor of value 5600 ohms, a capacitor of value 3 µF, and an inductor of value 1.5 mH are being defined, and given the names *R*1, *C*2, and *L*2.

MARTHA includes sixteen types of elements, of which five are one-port elements, and the rest two-port elements.

Resistor. The function that creates a resistor is *R*, and it interprets its argument (the number to its right) as the value of the resistance in ohms.

Example: a 1.5-Megohm resistor:

```
R2←R 1.5E6
```

Capacitor. The function *C* interprets its argument as the value of the capacitance in farads (not microfarads).

Example: a 2.5-pF capacitor

```
C6←C 2.5E¯12
```

Inductor. The function *L*, if given an argument that is a single number, creates an inductor with the value of the argument as the inductance in henries. If the argument is a vector of length three, then *L* creates a mutual inductor, with the three values interpreted as the self-inductance of the left coil, the self-inductance of the right coil, and the mutual inductance (all three expressed in henries). The inductor, of course, is a one-port element whereas the mutual inductor is a two-port element.

Straight-through Section. The straight-through section (Figure 2.6) is a two-port element given by *WTHRU* without any argument (none is necessary since this element has no parameter associated with it). The major purpose of *WTHRU* is to replace a section in a cascade; it is equivalent to an ideal transformer with turns ratio 1.

Polarity Reverser. The polarity reverser (Figure 2.6) is a two-port element given by *WR* without any argument (none is required). This element is useful, in conjunction with the wiring functions *WSP*, *WPS*, and *WSS* defined below, to change a "series" connection into a "difference" connection. It is equivalent to an ideal transformer with turns ratio -1.

Ideal Transformer. The function *IT* creates an ideal transformer, interpreting the argument as the turns ratio (turns on the input side divided by turns on the output side).

Operational Amplifier. Three models for operational amplifiers, of varying degrees of sophistication, are available in *MARTHA*. The function *OPAMP* creates them all. If its argument is a single number, then it is interpreted as the voltage gain. The input impedance is assumed to be infinite, and the output impedance zero. If the argument is a vector of length two, then the first number is interpreted as the open-circuit voltage gain and the second as the (real) output impedance in ohms. The input impedance is assumed to be infinite. If the argument is a vector of length three, then the three are interpreted, in order, as the open-circuit voltage gain, the output impedance in ohms, and the input impedance in ohms. Note that if you want a finite input impedance, you must specify an output impedance (which may be 0 if you wish).

Field-Effect Transistor. The function *FET* creates a model for a common-source field-effect transistor (Figure 2.6). The argument for *FET* has three values, which are the gate-source capacitance C_{gs}, the gate-drain capacitance C_{gd}, both in farads, and the transconductance g_m in mhos.

Example:

```
Q1←FET 10E¯12,2E¯12,2.5
```

Bipolar Transistor. The hybrid-pi model for bipolar transistors (Figure 2.6) is created by the function HYBRIDPI. The argument requires five values, which are the resistances r_x and r_π in ohms, the capacitances C_π and C_μ in farads, and the transconductance gm in mhos. The result is a common-emitter transistor.

Transmission Line. The function *TEM* creates a lossless transmission line. Its argument is a vector of length two; the two numbers are interpreted as, respectively, the characteristic impedance of the line in ohms, and the length in meters.

Example: a fifty-ohm line 3.5 cm. long:

```
T1←TEM 50,3.5E¯2
```

A 75-ohm line 90 feet long:

```
T2←TEM 75,90×.0254×12
```

Note that an arithmetic expression is used as the second element of the vector argument; it is immediately evaluated and the result, 27.432, is interpreted as the length in meters.

If the argument for *TEM* contains only one value instead of two, then the result is not a transmission line, but instead a matched termination for such a line, i.e. a resistor of that many ohms. This may appear trivial, since the same result is obtained from the program *R*. However, it is useful in conjunction with functions in the *MARTHA* library that calculate characteristic impedance of coaxial and microstrip lines.

If you wish you can specify the length of a transmission line as an electrical length in degrees at some reference frequency, instead of as a physical length in meters. The auxiliary function *DEGREESAT* sets up a calculation of the corresponding physical length. It requires two arguments, one at its left (the electrical length in degrees) and one at its right (the reference frequency in Hz). Simply use an expression like 45 *DEGREESAT* 1*E*7 instead of the length in meters.

Example: a 25-ohm line a quarter-wave long at 750 MHz:

```
T3←TEM 25,90 DEGREESAT 750E6
```

MARTHA normally assumes that the dielectric constant is one, and therefore uses the speed of light in free space. If the transmission line is dielectrically loaded, then a different value for the speed of light should be used. The same effect, however, is achieved by using a longer physical length. A calculation of an appropriate length is set up by the function *FORDIEL* followed by the relative dielectric constant.

Examples: 50-ohm lines loaded with a dielectric constant of 9:

```
T4←TEM 50,..25 FORDIEL 9
T5←TEM 50,90 DEGREESAT 750E6 FORDIEL 9
```

In the case of *T*5, the dielectric constant is ignored because the electrical length is specified rather than the physical length.

Waveguide. The function *WG* creates a lossless waveguide in the dominant mode (for rectangular guide, the TE_{10} mode). The argument for *WG* is a vector of length three; the three values in order are the cutoff frequency fc in Hz, the infinite-frequency characteristic impedance $Z_{0\infty}$ in ohms, and the length in meters. Because the waveguide is dispersive, the characteristic impedance is a function of frequency, of the form (for TE waves)

$$Z_0 = \frac{Z_{0\infty}}{\sqrt{1 - (f_c/f)^2}} = \frac{Z_{0\infty}\lambda_g}{\lambda} \tag{2.1}$$

where λ_g is the guide wavelength and λ is the free-space wavelength. The impedance at infinite frequency $Z_{0\infty}$ depends upon the size and shape of the waveguide, and also upon the precise meaning of Z_0. There is no unique way to define voltage and current in a waveguide, and in different circumstances different expressions for $Z_{0\infty}$ might be appropriate. A more complete discussion is given in Section 4.5.

The analysis in *MARTHA* is valid above the cutoff frequency, where Z_0 is real, and also below cutoff where Z_0 is imaginary.

The function *WG* can be used with an argument of length two, rather than three, in which case instead of a waveguide, a matched termination is produced. This termination is a one-port element with frequency-dependent impedance equal to Z_0---real above the cutoff frequency and imaginary below.

If you wish, you can specify the length of the waveguide in terms of electrical degrees at a reference frequency (above fc, of course) by using the function *DEGREESAT*.

Examples: an X-band waveguide, first 15 cm. long, and second 2 wavelengths at 9 GHz:

```
W1←WG 6.557E9,334.9,.15
W2←WG 6.557E9,334.9,720 DEGREESAT 9E9
```

If your waveguide is filled with a dielectric, declare the dielectric constant each time you use *WG*, by using *FORDIEL*. If the examples above had a dielectric constant equal to 9:

```
W3←WG 6.557E9,334.9,.15 FORDIEL 9
W4←WG 6.557E9,334.9,720 DEGREESAT 9E9 FORDIEL 9
```

Summary of Elements. The elements included in *MARTHA* are all illustrated in Figure 2.6. Note that four of the functions that define elements (*L*, *OPAMP*, *TEM*, and *WG*) define different elements according to the length of the argument. The others work with only one argument length, either one, three, or five. If you use an argument of the wrong length, an appropriate message will be printed.

Table 2.1 lists the elements in *MARTHA* and shows for each how the arguments are interpreted. Also given are the equations used by *MARTHA* for each element, and the range of allowed values for each argument. Note that values of zero for some of the arguments are not allowed, but that negative values generally can be used. Thus *MARTHA* can handle models for negative-resistance devices like tunnel diodes, negative waveguide lengths to compensate for junction effects, and unrealizably large mutual inductance.

Other Elements. The elements in Figure 2.6 and Table 2.1 are adequate for many practical circuits. The *MARTHA* library contains several other elements that you may find useful. The library also includes several functions to calculate parameters when other information is known, for example calculation of coaxial and microstrip lines from their physical dimensions. See Sections 4.1 and 4.5.

MARTHA allows user-defined elements. For details, see Section 3.6.

Table 2.1. Elements defined in *MARTHA*. In the equations, *S* is 2πjf. Additional elements in the *MARTHA* library are described in Section 4.1.

ELEMENT	*TYPE*	*NAME*	*ARGUMENT VECTOR*	*REQUIRED*	*EQUATIONS*
RESISTOR	1-*PORT*	*R*	*RESISTANCE RES IN OHMS*	*RES*≠0	*V*=*RES*×*I*
CAPACITOR	1-*PORT*	*C*	*CAPACITANCE CAP IN FARADS*		*I*=*S*×*CAP*×*V*
INDUCTOR	1-*PORT*	*L*	*INDUCTANCE IND IN HENRIES*	*IND*≠0	*V*=*S*×*IND*×*I*
STRAIGHT-THROUGH CONNECTION	2-*PORT*	*WTHRU*	(*NONE*)		*V*1=*V*2; *I*1=-*I*2
POLARITY REVERSE	2-*PORT*	*WR*	(*NONE*)		*V*1=-*V*2; *I*1=*I*2
MUTUAL INDUCTOR	2-*PORT*	*L*	*INPUT SELF-INDUCTANCE L1 IN HENRIES* *OUTPUT SELF-INDUCTANCE L2 IN HENRIES* *MUTUAL INDUCTANCE M IN HENRIES*	*M*≠0	*V*1=*S*×(*L*1×*I*1)+*M*×*I*2 *V*2=*S*×(*M*×*I*1)+*L*2×*I*2
IDEAL TRANSFORMER	2-*PORT*	*IT*	*TURNS RATIO N*	*N*≠0	*V*1=*N*×*V*2; *I*1=-*I*2÷*N*
OPERATIONAL AMPLIFIER	2-*PORT*	*OPAMP*	*OPEN-CIRCUIT VOLTAGE GAIN A* *OUTPUT IMPEDANCE ROUT IN OHMS* *INPUT IMPEDANCE RIN IN OHMS*	*A*≠0 *RIN*≠0	*V*1=*RIN*×*I*1 *V*2=(*A*×*V*1)+*ROUT*×*I*2
OPERATIONAL AMPLIFIER	2-*PORT*	*OPAMP*	*OPEN-CIRCUIT VOLTAGE GAIN A* *OUTPUT IMPEDANCE ROUT IN OHMS*	*A*≠0	*I*1=0 *V*2=(*A*×*V*1)+*ROUT*×*I*2
OPER. AMPLIFIER	2-*PORT*	*OPAMP*	*VOLTAGE GAIN A*	*A*≠0	*I*1=0; *V*2=*A*×*V*1
FIELD-EFFECT TRANSISTOR MODEL, GROUNDED-SOURCE	2-*PORT*	*FET*	*GATE-SOURCE CAPACITANCE CGS IN FARADS* *GATE-DRAIN CAPACITANCE CGD IN FARADS* *TRANSCONDUCTANCE GM IN MHOS*	*GM*≠0	*I*1=*S*×(*CGS*×*V*1)+*CGD*×*V*1-*V*2 *I*2=(*GM*×*V*1)+*S*×*CGD*×*V*2-*V*1
BIPOLAR-TRANSISTOR MODEL, GROUNDED-EMITTER	2-*PORT*	*HYBRIDPI*	*RESISTANCE RX IN OHMS* *RESISTANCE RPI IN OHMS* *CAPACITANCE CPI IN FARADS* *CAPACITANCE CMU IN FARADS* *TRANSCONDUCTANCE GM IN MHOS*	*RPI*≠0 *GM*≠0	*V*1=*VPI*+*RX*×*I*1 *I*1=(*VPI*×(*S*×*CPI*)+÷*RPI*)+ *S*×*CMU*×*VPI*-*V*2 *I*2=(*GM*×*VPI*)+*S*×*CMU*×*V*2-*VPI*
LOSSLESS TRANS-MISSION LINE	2-*PORT*	*TEM*	*CHARACTERISTIC IMPEDANCE Z0 IN OHMS* *LENGTH LEN IN METERS*	*Z*0≠0	*A*=★-*J*×○2×*LEN*×*F*÷3*E*8÷*DIEL*★.5 (*V*2-*Z*0×*I*2)=*A*×*V*1+*Z*0×*I*1 (*V*1-*Z*0×*I*1)=*A*×*V*2+*Z*0×*I*2
TRANSMISSION LINE CHARACTERISTIC IMPEDANCE	1-*PORT*	*TEM*	*CHARACTERISTIC IMPEDANCE Z0 IN OHMS*	*Z*0≠0	*V*=*Z*0×*I*
LOSSLESS WAVEGUIDE DOMINANT MODE *A*=*J*×*LEN*×*ZINF*×*F*÷*Z*0×3*E*8÷*DIEL*★.5 ○2×*A*)×*V*1+*Z*0×*I*1 ○2×*A*)×*V*2+*Z*0×*I*2	2-*PORT*	*WG*	*CUTOFF FREQUENCY FC IN HERTZ* *INFINITE-FREQUENCY CHARACTERISTIC* *IMPEDANCE ZINF IN OHMS* *LENGTH LEN IN METERS*	*ZINF*≠0 ~*FC*∈*F*	*Z*0=*ZINF*÷(1-(*FC*÷*F*)★2)★.5 (*V*2-*Z*0×*I*2)=(★- (*V*1-*Z*0×*I*1)=(★-
WAVEGUIDE CHARAC-TERISTIC IMPEDANCE	1-*PORT*	*WG*	*CUTOFF FREQUENCY FC IN HERTZ* *INF.-FREQ. CHAR. IMP. ZINF IN OHMS*	*ZINF*≠0 ~*FC*∈*F*	*V*=*I*×*ZINF*÷(1-(*FC*÷*F*)★2)★.5

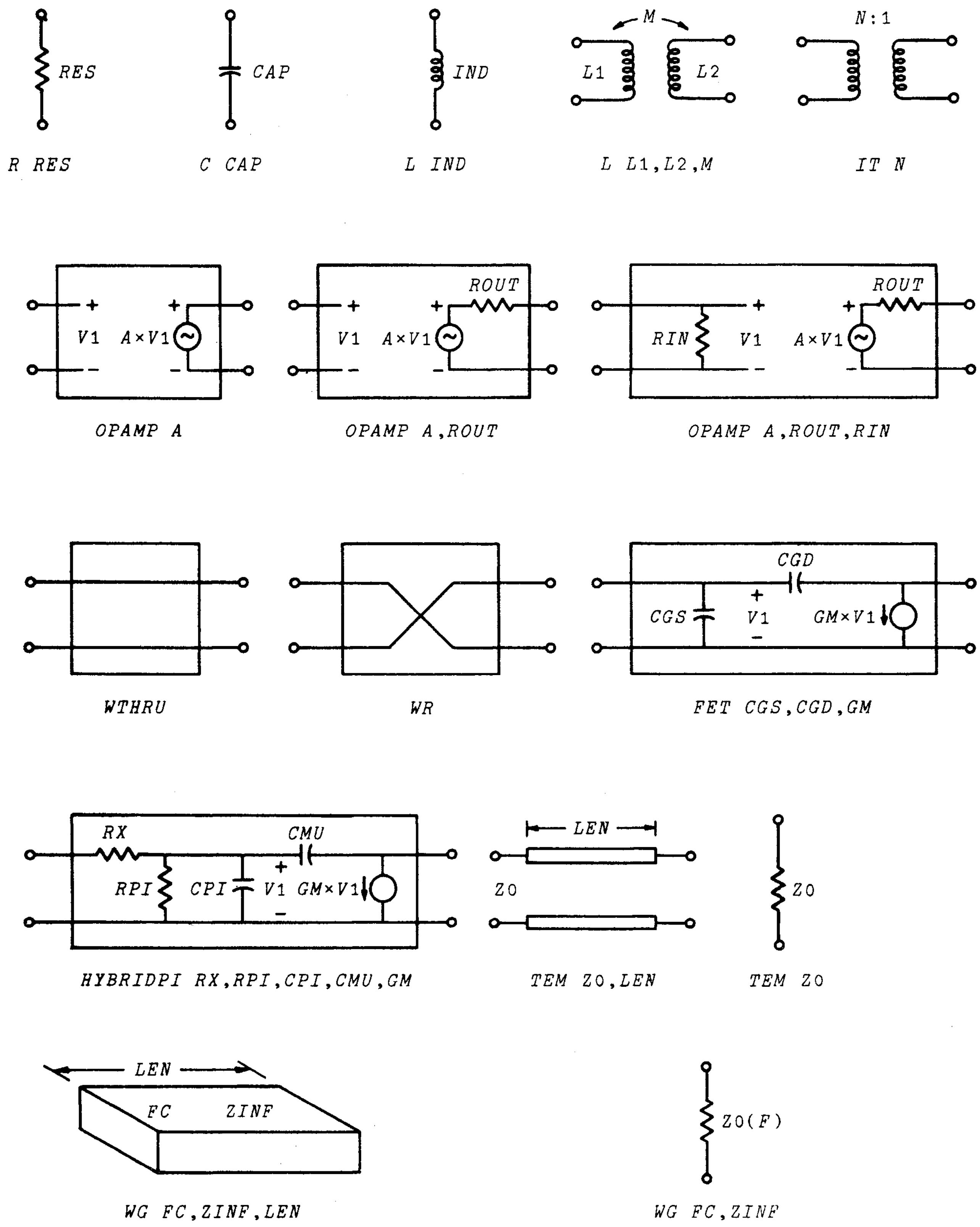


Figure 2.6. Elements defined in *MARTHA*.

2.6 Wiring

MARTHA uses an unusually versatile technique for describing the topology of networks, i.e., how the elements are wired together. The network is defined from simpler subnetworks, or "sections," by means of functions that simulate the wiring. You have great flexibility in defining the network in terms of the elements. Thus, at one extreme, you may write an expression for the entire network directly in terms of elements and their values. At the other extreme, you may define the network as two or three sections wired together, and then define each of these sections in terms of other sections, and so forth, until ultimately each section is defined in terms of elements and their values.

To see how the "wiring functions" are used, consider the one-port network of Figure 2.7. Suppose we have available a "series wiring function" *S* and a "parallel wiring function" *P*. Then the network of Figure 2.7 would be

*R*1 *S*((*L*1 *S* *R*2) *P* *C*1)

This expression is interpreted as follows: *R*1, *L*1, *R*2, and *C*1 are names of elements that have been defined using the element functions *R*, *L*, and *C*. Every element is considered to be a section. Two sections in series with each other (such as *L*1 *S* *R*2) are considered to be a section. So is the result of wiring two sections in parallel such as (*L*1 *S* *R*2) *P* *C*1. By repeated use of the series and parallel wiring functions, you can define any series-parallel network.

For series-parallel networks, the two wiring functions *S* and *P* (which are included in *MARTHA*) are sufficient. For two-port networks, however, you will need others. Fifteen wiring functions are defined in *MARTHA*. Some of these functions are, like *S* and *P*, "dyadic" in that they require two sections, one on each side. The others are "monadic," since they require only one section, which in the expression appears to the right of the function.

The fifteen functions are shown in Figure 2.8. First are the functions S and P which operate on one-port sections and yield, as a result, one-port sections. Next are the functions *WP* and *WS* which are monadic, and which convert one-port sections to two-port sections by placing the section either across the line (*WP*) or in series with the

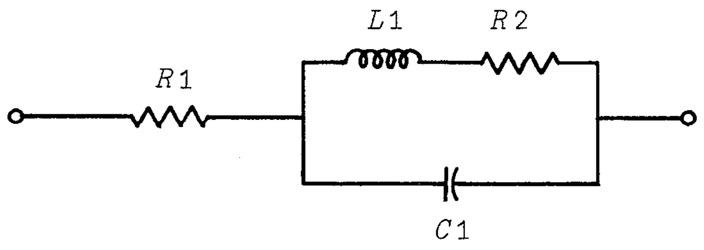


Figure 2.7. Simple one-port section.

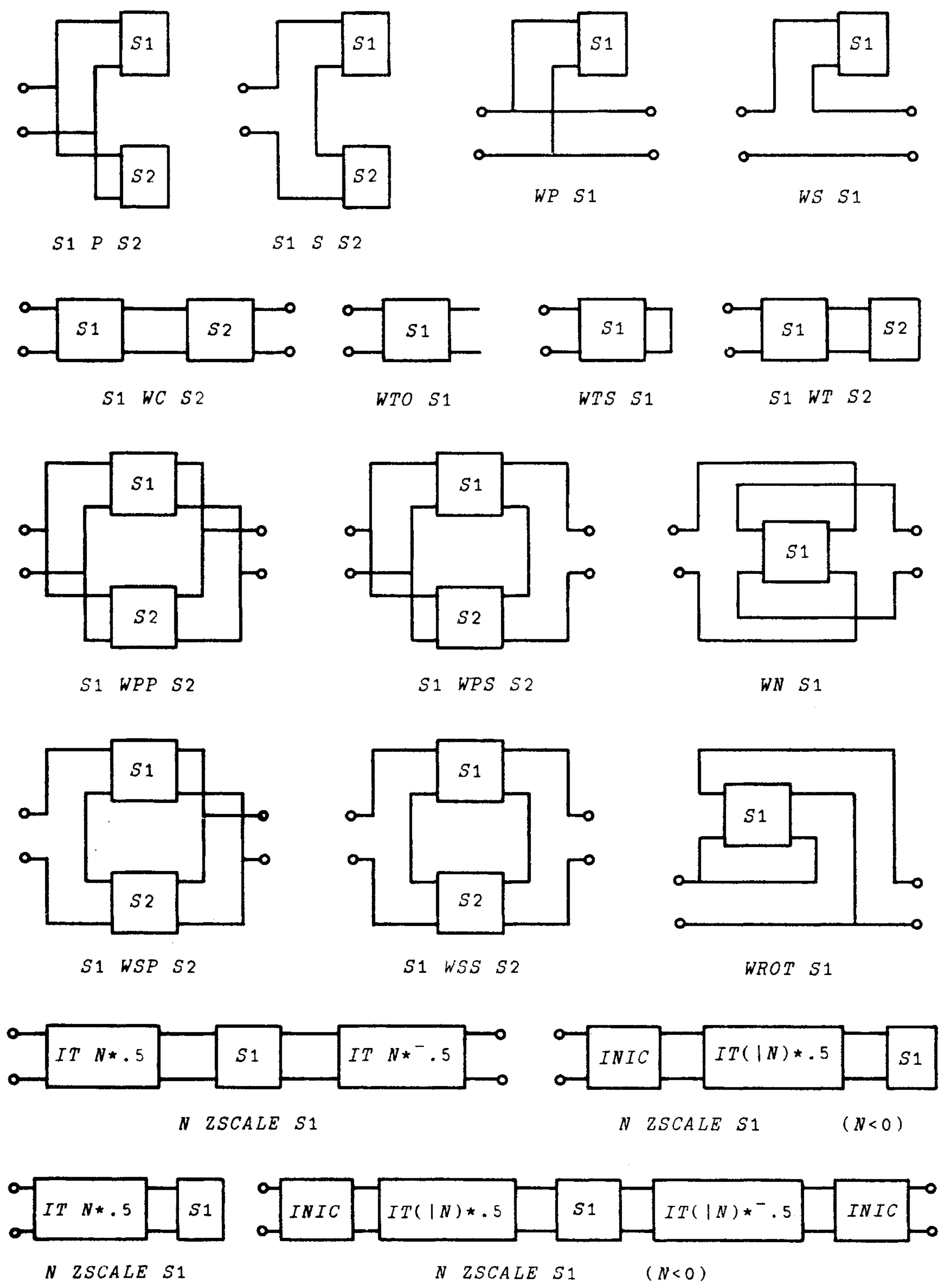


Figure 2.8. Wiring functions defined in *MARTHA*. The element *INIC* is a negative-impedance converter, and is only necessary when a negative impedance scale factor is used.

line (WS). Next is the important cascade function *WC*, which cascades two two-ports, the result being a two-port section. Next are three ways of converting a two-port section into a one-port section by terminating the output: either with a short circuit (*WTS*), or with an open circuit (*WTO*), or with a one-port section as a load (*WT*). Note that *WTO* and *WTS* are monadic and *WT* is dyadic; the argument for *WTO* and *WTS* is a two-port network, and the arguments for *WT* are a two-port on the left, and a one-port on the right.

Next are four functions *WPP*, *WPS*, *WSP*, and *WSS* for combining two two-port sections into a new two-port section by putting the inputs and outputs in series or parallel. Then come two monadic functions for permuting the ports of a two-port section. The first, *WN*, interchanges the input and output, and the second, *WROT*, rotates the section clockwise. You can use these functions to convert grounded-emitter transistors into grounded-collector or grounded-base stages.

The final wiring function is the impedance-scaling function *ZSCALE*. This is a dyadic function; the left argument is a number and the right argument is a section. The result is that section with all impedances scaled by the left argument. Thus, for example,5 *ZSCALE R* 3 is equivalent to *R* 15.

Some of the wiring functions expect one-port sections, and others expect two-port sections as arguments (*WT* expects one of each, and *ZSCALE* accepts either). For your convenience each function is capable of accepting either one-port or two-port sections as arguments. The following two conventions are used:

1. If a function expects a two-port section but finds a one-port section instead, then the two-port section in Figure 2.9 is used.
2. If a function expects a one-port section but finds a two-port section instead, then the output is open-circuited and the input port is used, according to Figure 2.10.

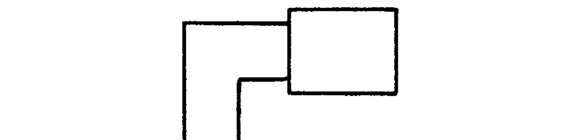
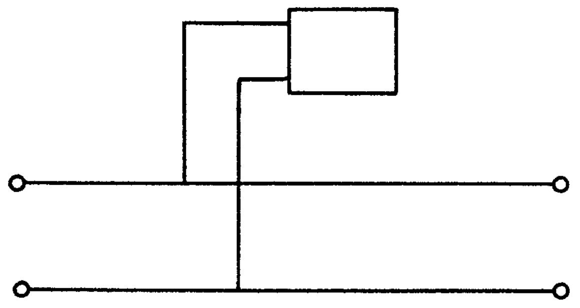



Figure 2.9. Automatic method of forming a two-port section from a one-port section is to place it across the input which is then connected to the output. This is equivalent to using the function *WP*.

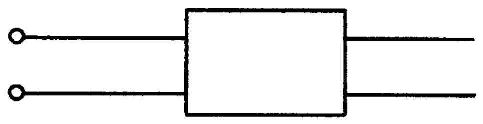


Figure 2.10. Automatic method of forming a one-port section from a two-port section is to open-circuit the output. This is equivalent to using the function *WTO*.

Because of these conventions, you can often omit *WP* and *WTO*.

Like all APL functions, each wiring function takes as its right argument the entire expression to its right, unless it appears inside a pair of parentheses, in which case it takes the entire expression up to the closing parenthesis. In forming expressions for sections in *MARTHA*, be sure to use this convention. Thus, for example, the network of Figure 2.7 could be described by either of the two following expressions:

```
R1 S(C1 P(L1 S R2))

R1 S C1 P L1 S R2
```

2.7 Examples of Network Description

After you have written a few expressions for networks and used the wiring functions, you will find that this technique is simple, straightforward, and easy to use. To become familiar with the element-definition functions and the wiring functions, you may want to look at Figures 2.11 through 2.15, which illustrate many of the important elements and wiring functions. Other examples are given in Section 2.13 of this manual.

2.8 Storing Network Definitions

If you want to analyze a certain network several times, it may be inconvenient to repeat the entire description in each command. You can store the definition of a network or a section under a name of your choice, and then simply refer to it by that name. You can use this stored definition as the network being analyzed, or you can wire it with other sections to form a larger network. The definition is stored using the standard APL function-definition scheme. You simply define a function whose value is set to the expression for the network.

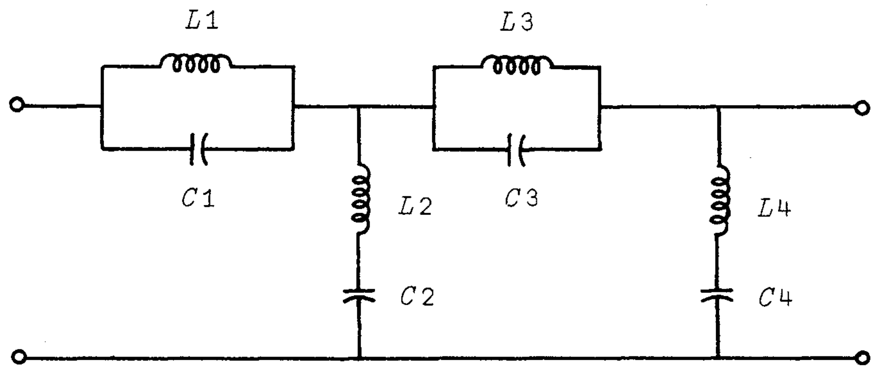


Figure 2.11. Band-stop filter. Assume the elements have already been defined. Expression: *(WS L1 P C1)WC(L2 S C2)WC(WS L3 P C3)WC L4 S C4*

For example, consider the prototype filter of Figure 2.16. Let's write an APL function which stores the definition of that network. We start by opening the definition with the ∇ symbol, then put a variable to represent the return, then the back arrow, then the name of the network:

```
    ∇Z←FILTER
```

The computer responds with [1] which is a request for line no. 1 of the function. In this line we simply set Z to the network:

```
[1] Z←(C .5)WC(WS L 1)WC(C 1)WC(WS L 1)WC(C 1)WC(WS L 1) WC C .5
```

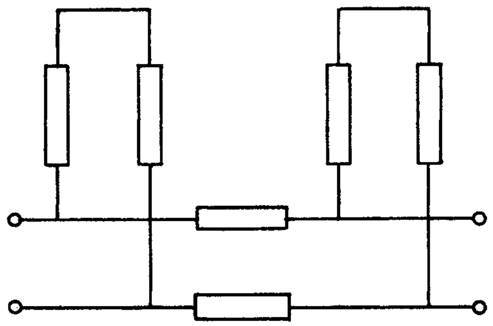


Figure 2.12. Double-stub tuner with 50-ohm lines and adjustable lengths *L*1 and *L*2, and fixed length 5 inches. Expression: (*WTS TEM* 50,*L*1)*WC* (*TEM* 50,5×.0254)*WC WTS TEM* 50,*L*2

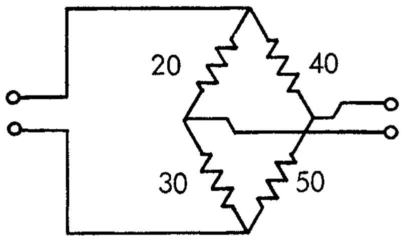


Figure 2.13. Wheatstone bridge, values in ohms. Expression: ((*WS R* 40)*WC R* 50)*WPS*(*WS R* 20) *WC*(*R* 30)*WC WR*

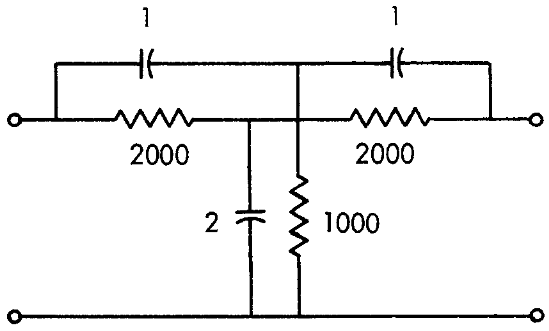


Figure 2.14. Twin-tee filter, values in ohms and µF. Expression: ((*WS R* 2000)*WC*(*C* 2*E*¯6)*WC WS R* 2000)*WPP*(*WS C* 1*E*¯6)*WC* (*R* 1000)*WC WS C* 1*E*¯6

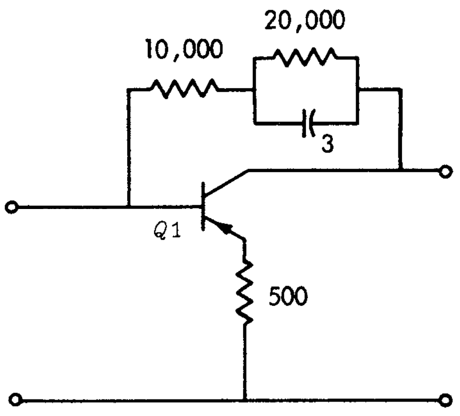


Figure 2.15. Feedback amplifier. Assume *Q*1 is already defined. Expression: (*Q*1 *WSS R* 500)*WPP WS*(*R* 1*E*4)*S*(*R* 2*E*4)*P C* 3*E*¯6

Note that in this definition, the capacitors, are, by convention, assumed to be placed across the line so that the function *WP* would be redundant, although it could be used without harm. After this line is typed in, the computer responds with [2] which is a request for line no. 2 of the function. The network definition is complete, so there is no need for another line. To indicate this, simply close the definition with another ∇. From now on, the name *FILTER* represents the entire network of Figure 2.16, and you do not need to type the definition again.

Alternatively, suppose we decide to view the filter as a cascade of three pi sections as shown in Figure 2.17. The three pi sections are identical, and so it might save effort to define the pi section just once, then wire three of them together. Let's define the pi section with the name *PI*, as follows

```
     ∇ Z←PI
[1]  Z←(C .5)WC(WS L 1)WC C .5
[2]  ∇
```

Then the three-section filter, Figure 2.16, would be defined as

```
     ∇ Z←FILT
[1]  Z←PI WC PI WC PI
[2]  ∇
```

and from now on *FILT* would refer to the filter of Figure 2.16.

Suppose that now we want to analyze a similar filter but with four of these pi sections, as illustrated in Figure 2.18. If *PI* and *FILT* have already been defined, then the definition of the new filter

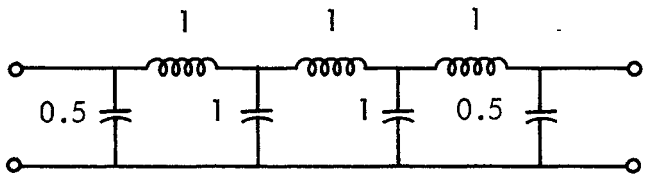


Figure 2.16. Three-section filter.

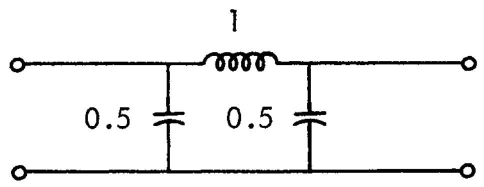


Figure 2.17. Pi section for use in the filter of Figure 2.16

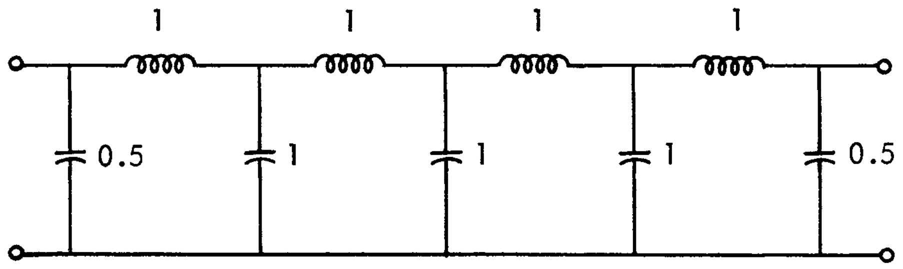


Figure 2.18. Four-section filter.

```
      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED 20:52:47 06/30/71

      ∇Z←FILTER
[1]  Z←(C .5)WC(WS L 1)WC(C 1)WC(WS L 1)WC(C 1)WC(WS L 1)WC C .5
[2]  ∇

      F←.05×⍳10
      PRINT VG OF FILTER

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71   11:7
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

     F          RE VG       IM VG
----------- ----------- -----------
 5.0000E¯02  1.7105E00  ¯2.4037E¯25
 1.0000E¯01 ¯2.9435E00  ¯8.5836E¯25
 1.5000E¯01 ¯1.0198E00  ¯2.3128E¯26
 2.0000E¯01 ¯1.6834E00   2.9305E¯25
 2.5000E¯01  1.5384E00   2.9052E¯25

 3.0000E¯01  2.1901E00  ¯1.2766E¯24
 3.5000E¯01 ¯1.3982E¯01 ¯3.0282E¯26
 4.0000E¯01 ¯2.9639E¯02 ¯3.8930E¯27
 4.5000E¯01 ¯1.0131E¯02 ¯1.0138E¯27
 5.0000E¯01 ¯4.3126E¯03 ¯3.5601E¯28

      ⍝ THE SMALL IMAGINARY PARTS OF THE VOLTAGE GAIN ARE CAUSED BY ZL NOT BEING INFINITE.
      ZL
1E25
      ∇Z←PI
[1]  Z←(C .5)WC(WS L 1)WC C .5
[2]  ∇

      ∇Z←FILT
[1]  Z←PI WC PI WC PI
[2]  ∇

      PRINT VG OF FILT

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71   11:9


     F          RE VG       IM VG
----------- ----------- -----------
 5.0000E¯02  1.7105E00  ¯2.4037E¯25
 1.0000E¯01 ¯2.9435E00  ¯8.5836E¯25
 1.5000E¯01 ¯1.0198E00  ¯2.3128E¯26
 2.0000E¯01 ¯1.6834E00   2.9305E¯25
 2.5000E¯01  1.5384E00   2.9052E¯25

 3.0000E¯01  2.1901E00  ¯1.2766E¯24
 3.5000E¯01 ¯1.3982E¯01 ¯3.0282E¯26
 4.0000E¯01 ¯2.9639E¯02 ¯3.8930E¯27
 4.5000E¯01 ¯1.0131E¯02 ¯1.0138E¯27
 5.0000E¯01 ¯4.3126E¯03 ¯3.5601E¯28

      ∇Z←NEWFILTER
[1]  Z←FILT WC PI
[2]  ∇

      PRINT VG OF NEWFILTER

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71   11:9


     F          RE VG       IM VG
----------- ----------- -----------
 5.0000E¯02  3.2890E00  ¯1.0435E¯24
 1.0000E¯01 ¯1.1995E00  ¯8.3687E¯26
 1.5000E¯01 ¯1.4122E00   1.5965E¯25
 2.0000E¯01  1.5119E00   2.2036E¯25
 2.5000E¯01  1.7038E00  ¯3.7968E¯25

 3.0000E¯01 ¯1.0927E00   1.4401E¯25
 3.5000E¯01  5.7931E¯02  1.2650E¯26
 4.0000E¯01  7.2822E¯03  9.5689E¯28
 4.5000E¯01  1.7400E¯03  1.7412E¯28
 5.0000E¯01  5.5715E¯04  4.5994E¯29
```

can be very simple:

```
       ∇Z←NEWFILTER
[1]    Z←FILT WC PI
[2]    ∇
```

As you can see, the technique of using wiring functions is very flexible. If you are clever in deciding what subnetworks to define separately, you can not only simplify the final expressions, but also allow easy extension to other, similar networks.

You can store your network definitions indefinitely in your own private workspaces, and build up and maintain your own collection of commonly used networks and models. See Section 3.3 for a description of workspace storage.

2.9 Output List

Recall that the general form of a command to *MARTHA* is

$$\left\{\begin{matrix} \textit{PRINT} \\ \textit{PLOT} \\ \textit{PLOG} \end{matrix}\right\} \quad \text{<output list> } \textit{OF} \text{ <network description>}$$

The output list is a list of response functions, separated by commas. The response functions may be modified, and there are some format items that can be inserted in the list.

Examples:

PLOT Z, Y OF (*R* 4)*P C* 3*E*¯6

*PRINT RE Z*11, *MAG Z*11, *DEG Z*11 *OF FILT*

PLOG DB IG, *S*11, *S*21 *OF NEWFILTER*

PLOT 43 *HIGH*, 72 *WIDE IM S*11 *VS RE S*11 *OF PI WC PI*

Here *Z*, *Y*, *Z*11, *IG*, *S*11, and *S*21 are response functions; *RE*, *MAG*, *DEG*, *DB*, and *IM* are modifiers; and 43 *HIGH*, 72 *WIDE*, and *VS* are formatters. The next three sections of this manual describe response functions, modifiers, and formatters in detail.

2.10 Response Functions

MARTHA contains 30 response functions, of which 26 are complex and 4 real. In addition, the *MARTHA* library contains over 40 more.

There are three complex response functions for one-port networks: the impedance Z, the admittance Y, and the reflection coefficient *SC*. Z and Y depend only upon the network, and *SC* depends upon the network and also upon a resistance that is not the resistance of any resistor in the network, but which is used for normalization purposes. This is the "normalization impedance" *ZN*; it is initially set to 50 ohms, but you can change it to any positive value you wish, as often as you wish, and *MARTHA* will always use the current value when evaluating *SC*. To set *ZN* to 64 ohms, for example:

```
      ZN←64
```

note that *ZN* is a resistance rather than a resistor, so set it to 64 rather than to *R* 64.

These three response functions are for one-port networks. If they are applied to two-port networks, the output port is assumed to be open-circuited, in accordance with the convention for wiring func tions given earlier. The other 27 response functions are for two-port networks. If they are applied to one-port networks, the effect is the same as invoking *WP* first.

Twelve of the response functions are complex two-port parameters of the network, namely, the elements of the impedance matrix *Z*11, *Z*12, *Z*21, and *Z*22; the admittance matrix *Y*11, *Y*12, *Y*21, and *Y*22; and the hybrid matrix *H*11, *H*12, *H*21, and *H*22. (The other two-port matrices are in the *MARTHA* library.) These twelve response functions depend only upon the network.

Four of the complex response functions, *S*11, *S*12, *S*21, and *S*22, are entries of the scattering matrix, which depend upon the network and also upon the values of two normalization impedances at the input and output of the network, named *ZNIN* and *ZNOUT*. These are both initi ally set to 50 ohms, but you may reset them to any positive values. They need not be equal to each other or to *ZN*. *MARTHA* will always use the current values when evaluating *S*11, *S*12, *S*21, or *S*22.

The other response functions depend not only upon the network, but also upon the generator and/or load impedances. Two-port networks are assumed to be terminated as in Figure 2.19. The generator impedance is named *ZG*, and is initially set to 0 ohms. The load impedance is named *ZL*, and is initially set to a very high value, 10^{25} ohms. You will ordinarily want to set them to other values that are appropriate for your applications, for example 50 ohms. This is done with state ments like

```
      ZG←25
      ZL←50
```

Negative values for *ZG* and *ZL* are allowed. You may change *ZG* and *ZL* as often as you wish, and *MARTHA* will always use the current values. Note that *ZG* and *ZL* are resistances, not resistors, so do not set them to *R* 50, but to 50.

Three of the complex response functions refer to the input when the output is terminated by the impedance *ZL*. They depend upon the network and also upon the value of *ZL*, but not upon *ZG*. They are the input impedance *ZIN*, and input admittance *YIN*, and the input reflection coefficient *SIN*. *SIN* depends upon *ZNIN* as well as upon the network and *ZL*.

Three of the complex response functions relate to the output port when the input is terminated by the impedance ZG. They depend upon the network and upon *ZG*, but not *ZL*. They are the output impedance *ZOUT*, the output admittance *YOUT*, and the output reflection coefficient *SOUT*. *SOUT* depends also upon the normalization impedance at the output port *ZNOUT*.

The remaining five response functions relate to the two-port network as terminated by generator and load, Figure 2.19. These are the voltage gain, the insertion gain, the available gain, the power gain, and the transducer gain. These are defined as follows: The available power from the generator is

$$P_{ga} = \frac{|E_g|^2}{4Z_g} \tag{2.2}$$

and, similarly, the available power at the output of the network is

$$P_{oa} = \frac{|E_o|^2}{4 \text{ Re } Z_{out}} \tag{2.3}$$

where E_o is the open-circuit output voltage of the network. The available gain *AG* is the ratio of P_{oa} to P_{ga}, and depends upon the network and *ZG*, but not upon *ZL*. The actual power output is

$$P_{out} = \frac{|E_L|^2}{Z_L} \tag{2.4}$$

where E_L is the output voltage with the load in place. The voltage gain *VG* is the ratio of E_L to E_g. The transducer gain *TG* is the ratio of P_{out} to P_{ga}. These are both functions of the network, *ZG*, and *ZL*. The power input to the network is

$$P_{in} = |I_{in}|^2 \text{ Re } Z_{in} \tag{2.5}$$

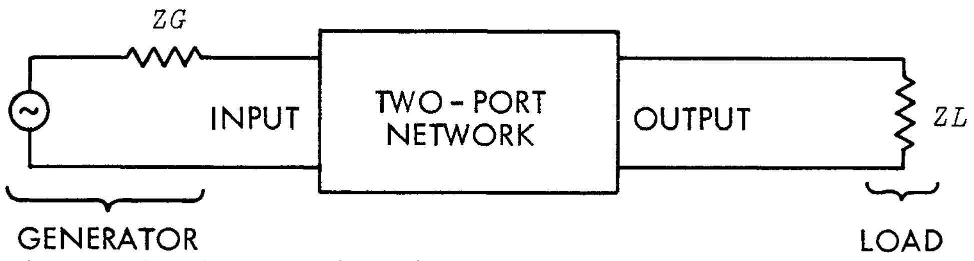


Figure 2.19. Termination of two-port networks assumed when some of the response functions are calculated. The generator and load impedances *ZG* and *ZL* are not part of the network definition.

and the power gain *PG* is the ratio of P_{out} to P_{in}. This depends upon the network and *ZL*, but not upon *ZG*. Finally, the power that would be delivered to the load if the network were not in place (i.e., were replaced by *WTHRU*) is

$$P_{wthru} = \frac{|E_g|^2 Z_L}{(Z_g + Z_L)^2} \qquad (2.6)$$

The insertion gain *IG is* the ratio of P_{out} to P_{wthru}, and depends upon the network, *ZL*, and *ZG*.

Of these five gains, *VG is* complex and the other four are real. If you want these expressed in decibels, this can be done with the modifier *DB* as explained in the next section.

The four power gains *AG*, *IG*, *PG*, and *TG*, may be greater than one for active networks. For passive networks, the corresponding losses are often used. The insertion loss is the reciprocal of *IG*, or when expressed in decibels is the negative of *DB IG*. Thus *IG* can be used when the insertion loss is desired (or the response function *IL* in the *MARTHA* library can be used).

Table 2.2 summarizes the response functions in *MARTHA*.

2.11 Output Modifiers

If you request printing or plotting of any of the complex response functions, you will get the real and imaginary parts. By using the modifiers *RE*, *IM*, *MAG*, *DB*, *RAD*, or *DEG*, you can request the real part, the imaginary part, the magnitude, the magnitude expressed in decibels, the phase angle in radians, or the phase angle in degrees. Each modifier acts on the response function immediately to its right only (contrary to the APL function convention that each function acts on the entire expression to its right).

Four of the modifiers, *RE*, *IM*, *RAD*, and *DEG*, act only on complex responses, and will be ignored if you attempt to modify a real response such as *IG* with them. The other two, *MAG* and *DB*, may be used also on real responses, and will have the effect of taking the absolute value, or expressing the absolute value in decibels, respectively. The modifier *DB*, when modifying a complex response, leads to 20 times the log (to the base 10) of the magnitude. When modifying a real response, it yields 10 times the log (to the base 10) of the absolute value.

The modifiers *MAG* and *DB* can modify even the result of a previous modifier so that, for example, you could get the absolute value of the imaginary part of the impedance:

```
      PRINT MAG IM Z OF (R 1)P(C 1E¯6)P L 1E¯3
```

Table 2.2. Response Functions in *MARTHA*. Of the 30, 26 are complex and 4 real.

NAME	*C/R*	*MEANING*	*DEPENDS ON*
Z	*C*	*IMPEDANCE OF A 1-PORT NETWORK*	*NETWORK*
		V=Z×I	
Y	*C*	*ADMITTANCE OF A 1-PORT NETWORK*	*NETWORK*
		I=Y×V	
SC	*C*	*REFLECTION COEFFICIENT OF 1-PORT*	*NETWORK, ZN*
		B=SC×A	
Z11	*C*	*IMPEDANCE MATRIX*	*NETWORK*
Z12	*C*	*V1=(Z11×I1)+(Z12×I2)*	
Z21	*C*	*V2=(Z21×11)+(Z22×12)*	
Z22	*C*		
Y11	*C*	*ADMITTANCE, MATRIX*	*NETWORK*
Y12	*C*	*I1=(Y11×V1)+(Y12×V2)*	
Y21	*C*	*I2=(Y21×V1)+(Y22×V2)*	
Y22	*C*		
H11	*C*	*HYBRID MATRIX*	*NETWORK*
H12	*C*	*V1=(H11×I1)+(H12×V2)*	
H21	*C*	*I2=(H21×I1)+(H22×V2)*	
H22	*C*		
S11	*C*	*SCATTERING MATRIX*	*NETWORK, ZNIN, ZNOUT*
S12	*C*	*B1=(S11×A1)+(S12×A2)*	
S21	*C*	*B2=(S21×A1)+(S22×A2)*	
S22	*C*		
ZIN	*C*	*INPUT IMPEDANCE V1÷I1*	*NETWORK, ZL*
YIN	*C*	*INPUT ADMITTANCE I1÷V1*	*NETWORK, ZL*
SIN	*C*	*INPUT REFLECTION COEFFICIENT B1÷A1*	*NETWORK, ZL, ZNIN*
ZOUT	*C*	*OUTPUT IMPEDANCE*	*NETWORK, ZG*
		V2÷I2 WHEN OUTPUT EXCITED	
YOUT	*C*	*OUTPUT ADMITTANCE*	*NETWORK, ZG*
		I2÷V2 WHEN OUTPUT EXCITED	
SOUT	*C*	*OUTPUT REFLECTION COEFFICIENT*	*NETWORK, ZG,ZNOUT*
		B2÷A2 WHEN OUTPUT EXCITED	
VG	*C*	*VOLTAGE GAIN V2÷EG*	*NETWORK, ZG, ZL*
AG	*R*	*AVAILABLE GAIN POUT,AV÷PIN,AV*	*NETWORK, ZG*
IG	*R*	*INSERTION GAIN*	*NETWORK, ZG, ZL*
		POUT(NETWORK)÷POUT(WTHRU)	
PG	*R*	*POWER GAIN POUT÷PIN*	*NETWORK, ZL*
TG	*R*	*TRANSDUCER GAIN POUT÷PIN,AV*	*NETWORK, ZG, ZL*

2.12 Output Formats

You may have any text you wish printed at the top of your output. This feature is useful for identifying the network you are analyzing, or recording conditions, or for a caption if you are using the output directly in a report. Just set the variable *TITLE* to whatever text you have in mind, but place a single quote mark ' before and after the text. The text may include carriage returns and any legal APL characters. This use of *TITLE* is a "one-shot" affair, and your title is replaced by a blank as soon as it is used, so that it will not appear by mistake on a later printout.

Aside from this title, when you use the *PRINT* command in *MARTHA*, the format of the output is fixed and you have no control over it. If there is more than one frequency (the usual case) a heading (including *TITLE*) will be printed and then the various responses will appear in separate columns with the frequency at the left as the independent variable. When there is only one frequency, the output is the same except that the heading is omitted.

When you use the *PLOT* or *PLOG* commands, you can control the format in several ways. These two differ in that the independent variable (ordinarily the frequency) is plotted on a linear scale by *PLOT* and on a logarithmic scale by *PLOG*.

You can specify the size of the graphs produced by *PLOT* and *PLOG*. If you do not specify the size, they will be 50 characters wide (each character a tenth of an inch) and 50 lines high (each line a sixth of an inch). This is a convenient size so that, complete with scales and headings, it will fit on a single page. You can ask for a different width and/or a different height by including in the output list the size followed by *WIDE* or *HIGH*.

Example:

```
PLOT 40 WIDE 35 HIGH Y, MAG Z OF MODEL1
```

This is a "one-shot" request, applying only to the single plot in question. Subsequent plots will be 50 x 50 unless you again ask for something different. Such requests will be ignored if the results are printed rather than plotted.

Each dependent variable (item in the output list) is plotted against the independent variable (ordinarily the frequency) and the scales are chosen so that each plot lies completely within the graph but shows reasonable detail. Thus the dependent variables generally have different scales. If you want them all plotted with the same scale, at some sacrifice in detail of some plots, include the format function *SS* in the output list.

Example:

```
PLOT SS ZIN, ZOUT OF AMP
```

The points on the graphs are indicated with symbols that print reasonably well centered in the space. If you want to use different plotting symbols, for example for mnemonic reasons, include them, in the order desired, surrounded by single quotes, followed by the word *SYMBOLS* in the output list.

Example:

```
PLOG 'RIMD' SYMBOLS VG, MAG VG, DEG VG OF FILTER
```

This again, like *WIDE*, *HIGH*, and *SS*, is a "one-shot" request.

Ordinarily the frequency is the independent variable, and all items in the output list are plotted against it. However, *MARTHA* has the ability to plot any response function against any other response function. To use this feature, precede one of the items in the output list with *VS* and it will be used in place of the frequency.

Example:

```
PLOT IM S11 VS RE S11 43 HIGH 72 WIDE OF FILTER
```

2.13 Examples

Nine completely worked out examples are given in this section. These have been chosen so as to illustrate most of the features of *MARTHA* discussed so far, as well as a few to be discussed in Chapter 3.

In each example, the following steps are shown:

1. The frequency *F* and possibly the generator and load impedances *ZG* and *ZL* are set. (In some cases, *ZN*, *ZNIN*, and *ZNOUT* are set also.)
2. The network is defined by using the element-definition functions and the wiring functions. In simple cases this can be done on the same line with the output list; in other cases it is done with stored network definitions.
3. The *PRINT*, *PLOT*, or *PLOG* command is given, with the output list, including modifiers and format functions.

Example 1. Parallel Tuned Circuit, Figure 2.20. Features covered: Catenation for specifying *F; R; L; C; P; S; Z; MAG; DEG; PRINT; OF*. First the active workspace is cleared, and *MARTHA* is copied. Then *F* is specified. Note that the factor 1*E*6 multiplies all the frequencies to its right, in accordance with the idea that the APL function × has as its right-hand argument everything to its right. To verify that the frequency is correct, it is printed out. In this case the circuit is so simple (a capacitor of value .025 μF in parallel with an inductor of value 1 μH) that it is typed on the same line with the output request. The impedance *Z* is requested, and the real and imaginary parts are produced. Note that the parentheses around C .025*E*¯6 are necessary since the capacitor must be defined before the wiring function *P* can act. Because the network is lossless, the impedance is imaginary.

The resonance is close to 1 MHz, so let's expand the detail in that region by redefining the frequency vector. The next printout

```
           ⍝ BEGINNING OF EXAMPLE 1.
     )CLEAR
CLEAR WS
     )COPY 100 MARTHA
SAVED    20.52.47 06/30/71

      F←1E6×.8,.85,.9,.95,1,1.05,1.1,1.15,1.2
      F
800000 850000 900000 950000 1000000 1050000 1100000 1150000 1200000
      PRINT Z OF (C .025E¯6)P L 1E¯6

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71   11:54
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

      F         RE Z       IM Z
 ----------- ---------- -----------
  8.0000E05   .0000E00   1.3646E01
  8.5000E05   .0000E00   1.8614E01
  9.0000E05   .0000E00   2.8295E01
  9.5000E05   .0000E00   5.4627E01
  1.0000E06   .0000E00   4.8186E02

  1.0500E06   .0000E00  ¯7.4864E01
  1.1000E06   .0000E00  ¯3.5596E01
  1.1500E06   .0000E00  ¯2.3671E01
  1.2000E06   .0000E00  ¯1.7900E01

      F←1E6×.95,.96,.97,.98,.99,1,1.01,1.02,1.03,1.04,1.05
      F
950000 960000 970000 980000 990000 1000000 1010000 1020000 1030000 1040000
      1050000
      PRINT Z OF (C .025E¯6)P L 1E¯6

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71   11:55


      F         RE Z       IM Z
 ----------- ---------- -----------
  9.5000E05   .0000E00   5.4627E01
  9.6000E05   .0000E00   6.6711E01
  9.7000E05   .0000E00   8.5497E01
  9.8000E05   .0000E00   1.1813E02
  9.9000E05   .0000E00   1.9034E02

  1.0000E06   .0000E00   4.8186E02
  1.0100E06   .0000E00  ¯9.3347E02
  1.0200E06   .0000E00  ¯2.3884E02
  1.0300E06   .0000E00  ¯1.3750E02
  1.0400E06   .0000E00  ¯9.6813E01

  1.0500E06   .0000E00  ¯7.4864E01

      (÷100)×(1E¯6÷.025E¯6)⋆.5
0.0632455532
      PRINT Z, MAG Z,DEG Z OF (C .025E¯6)P(L 1E¯6)S R .063

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71   11:57


      F        RE Z       IM Z       MAG Z      DEG Z
 ---------- ---------- ---------- ---------- ----------
  9.5000E05   5.2378E00   5.4177E01   5.4429E01   8.4478E01
  9.6000E05   7.6220E00   6.5911E01   6.6350E01   8.3404E01
  9.7000E05   1.2149E01   8.3763E01   8.4639E01   8.1747E01
  9.8000E05   2.2413E01   1.1496E02   1.1615E02   7.8847E01
  9.9000E05   5.4125E01   1.7411E02   1.8233E02   7.2732E01

  1.0000E06   2.3511E02   3.0343E02   3.8385E02   5.2230E01
  1.0100E06   4.3116E02  ¯2.9957E02   5.2502E02  ¯3.4791E01
  1.0200E06   7.6649E01  ¯2.1000E02   2.2355E02  ¯6.9949E01
  1.0300E06   2.7165E01  ¯1.3162E02   1.3439E02  ¯7.8338E01
  1.0400E06   1.3514E01  ¯9.4752E01   9.5711E01  ¯8.1883E01

  1.0500E06   8.0012E00  ¯7.3921E01   7.4353E01  ¯8.3822E01

      ⍝ END OF EXAMPLE 1.
```

shows more detail close to the resonance. Now, for this frequency, let's investigate the effect of a finite Q for the inductor. We model the inductor by a series R-L combination, with a resistance given by

$$R = \frac{1}{Q}\sqrt{\frac{L}{C}} \tag{2.7}$$

where Q is the quality factor at the resonant frequency. To calculate the numerical value for the resistance, we take advantage of the fact that APL will print the value of any expression we can type. A Q of 100 in this example leads up to a resistance of .063 ohms. Now we ask for the impedance, the magnitude of the impedance, and phase in degrees, of the tuned circuit with finite Q. In the expression for the network, the wiring functions *P* and *S* are executed in the order of appearance from right to left, so that the resistor and inductor are put in series before the capacitor is put in parallel.

Example 2. Five-pole Chebyshev low-pass filter, Figure 2.21. Features illustrated: Stored network definition; *ZG*; *ZL*; *ZNIN*; *ZNOUT*; Linear sweep for *F*; Catenate with former *F*; *WC*; *WS*; Implied *WP*; S11; *IG*; *DB*; *RE*; *IM*; *PLOT*; *VS*. The filter is a frequency- and impedance-scaled version of a low-pass prototype with 3dB ripple.[1] The impedance level is 100 ohms, and the cutoff frequency is 159 Hz. The network definition is stored under the name *CHEB*, in an APL function with no arguments. Note that it is not necessary to precede the capacitors by *WP* since this is automatically assumed. After the network has been defined, and *ZG* and *ZL* set, the frequency is defined with with the aid of the index generator ι. The insertion gain (reciprocal of insertion loss) and insertion gain expressed in dB are plotted. Next, the real and imaginary parts of the input scattering matrix entry *S*11 are plotted, after the normalization impedances have been specified. Next, it is desired to plot the imaginary part versus the real part. To make the result more readable, 21 new frequencies are catenated with the previous 25 frequencies. The result resembles a Smith chart, except that the scales are not the same. By the use of the format functions *HIGH* and *WIDE*, the scales can be made proper for

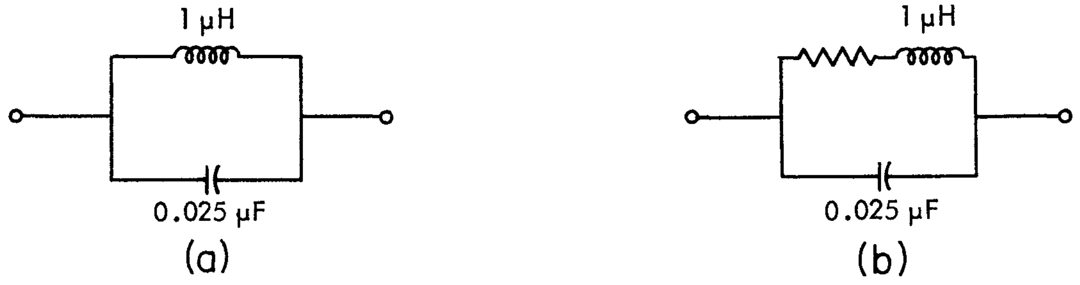


Figure 2.20. Parallel tuned circuit for example 1. (a) with infinite Q; (b) model for finite Q.

```
      ⍝ BEGINNING OF EXAMPLE 2.
      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED       20.52.47 06/30/71

      ∇Z←CHEB
[1]  Z←(C 34.817E¯6)WC(WS L .07618)WC(C 45.381E¯6)WC(WS L .07618)WC C 34.817E¯6
[2]  ∇

      ZG←100
      ZL←100
      ⍳25
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25
      F←8×⍳25
      F
8 16 24 32 40 48 56 64 72 80 88 96 104 112 120 128 136 144 152 160 168 176 184
      192 200
      PLOT IG, DB IG OF CHEB

CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/12/71   12:2
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

○      IG    0.000E00                 5.00E¯01                 1.000E00
×   DB IG   ¯3.00E01                 ¯1.50E01                  0.000E00
 ↓       F     |                        |                        |
    0.0000E00 |----|----|----|----|----|----|----|----|----|----|
              |
              |                                              ○  ×
              |
              |                                        ○      ×
    2.0000E01 |
              |                                 ○            ×
              |
              |                            ○                ×
              |
    4.0000E01 |                         ○                  ×
              |
              |                        ○                   ×
              |
              |                         ○                  ×
    6.0000E01 |
              |                           ○                 ×
              |
              |                                ○             ×
              |
    8.0000E01 |                                       ○       ×
              |
              |                                               ○ ×
              |
              |                                                 ×
    1.0000E02 |
              |                                          ○     ×
              |
              |                                 ○            ×
              |
    1.2000E02 |                           ○                 ×
              |
              |                        ○                   ×
              |
              |                          ○                  ×
    1.4000E02 |
              |                                    ○          ×
              |
              |                                                 ×
              |
    1.6000E02 |                     ○                     ×
              |
              |     ○                            ×
              |
              | ○                        ×
    1.8000E02 |
              |○                  ×
              |
              ○             ×
              |
    2.0000E02 ○----|---×|----|----|----|----|----|----|----|----|
```

direct overlay over a standard size Smith chart; the request would be

PLOT 72 *WIDE* 43 *HIGH IM S*11 *VS RE S*11 *OF CHEB*

Example 3. Double-tuned filter using coupled coils,[2] Figure 2.22. Features illustrated: Network-definition function with an argument; simultaneous analysis of more than one network; interlacing linear frequency sweeps; mutual inductance; *VG*; *SS*. It is desired to analyze this circuit as a function of the degree of coupling, i.e., as a function of the mutual inductance M. The network is defined with an argument which is interpreted as the mutual inductance, so that M can be varied conveniently. Note the use of the function *L* with three

```
      ZNIN←100
      ZNOUT←100
      PLOT S11 OF CHEB

CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/12/71   12:9


o   RE S11  ¯1.000E00                 0.000E00               1.000E00
×   IM S11  ¯1.000E00                 0.000E00               1.000E00
 ↓       F    |                        |                        |
    0.0000E00 |----|----|----|----|----|----|----|----|----|----|
              |
              |                  ×   o
              |
              |               ×  o
    2.0000E01 |
              |             o ×
              |
              |          o     ×
              |
    4.0000E01 |        o         ×
              |
              |       o             ×
              |
              |       o               ×
    6.0000E01 |
              |        o                 ×
              |
              |          o                  ×
              |
    8.0000E01 |               o              ×
              |
              |                     o      ×
              |
              |                      ×  o
    1.0000E02 |
              |               ×       o
              |
              |           ×      o
              |
    1.2000E02 |           × o
              |
              |          o   ×
              |
              |        o         ×
    1.4000E02 |
              |           o            ×
              |
              |                       × o
              |
    1.6000E02 |     ×                  o
              |
              |   ×          o
              |
              |     ×   o
    1.8000E02 |
              |      ×o
              |
              |     o   ×
              |
    2.0000E02 |----o----×----|----|----|----|----|----|----|----|
      F←12,20,76,84,92,94,98,100,102,106,108,140,146,148,149,150,151,154,156,158,164,F
      PLOT IM S11 VS RE S11 OF CHEB

CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/12/71   12:9

o   IM S11  ¯1.000E00                 0.000E00                 1.000E00
 ↓  RE S11    |                        |                        |
   ¯1.0000E00 |----|----|----|----|----|----|----|----|----|----|
              |
              |
              |
              |
   ¯8.0000E¯01|         o
              |        o
              |
              |      o              o o
              |                  o       o
   ¯6.0000E¯01|     o              o
              |              o o            o
              |                        o
              |                              o
              |           o   o
   ¯4.0000E¯01|   o                     o
              |                              o
              |               o
              |                          o
              |   o       o   o              o
   ¯2.0000E¯01|                          o
              |            o   o
              |                         o  o
              |             o    o
              |               o         o
     0.0000E00|     o          o o     o
              |                    o oo
              |
              |        o
              |                  o
    2.0000E¯01|             o
              |
              |
              |
              |
    4.0000E¯01|
              |
              |
              |
              |
    6.0000E¯01|
              |
              |
              |
              |
    8.0000E¯01|
              |
              |
              |
              |
    1.0000E00 |----|----|----|----|----|----|----|----|----|----|
     ⍝  END OF EXAMPLE 2.
```

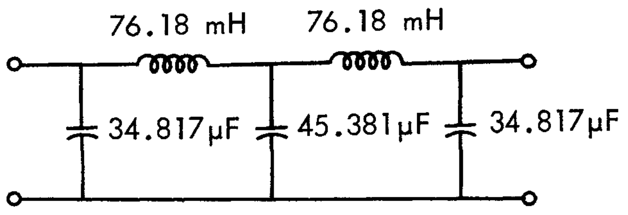


Figure 2.21. Chebyshev low-pass filter, example 2.

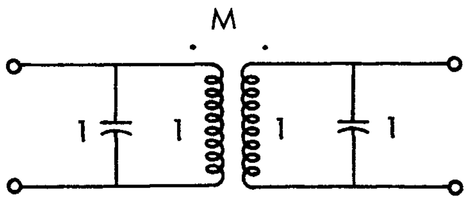


Figure 2.22. Double-tuned filter, example 3.

```
      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED  20:52:47 06/30/71

      ∇Z←DT M
[1]  Z←(C 1)WC(L 1,1,M)WC C 1
[2]  ∇
      ZG←100
      ZL←100
      F←.15+0,.0008×⍳25
      F
0.15 0.1508 0.1516 0.1524 0.1532 0.154 0.1548 0.1556 0.1564 0.1572 0.158
      0.1588 0.1596 0.1604 0.1612 0.162 0.1628 0.1636 0.1644 0.1652 0.166
      0.1668 0.1676 0.1684 0.1692 0.17

      PLOT VG, MAG VG OF DT .01

CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/12/71   12:15
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY


○   RE VG   ¯4.000E¯01                0.000E00                 4.000E¯01
×   IM VG   ¯5.000E¯01               ¯1.500E¯01                2.000E¯01
=  MAG VG    0.000E00                 2.500E¯01                5.000E¯01
 ↓       F    |                        |                        |
    1.5000E¯01|=---|----|----|----|----○----|----|×---|----|----|
              |
              |=                       ○          ×
              |
              |=                       ○          ×
    1.5200E¯01|
              |=                       ○           ×
              |
              | =                      ○           ×
              |
    1.5400E¯01| =                      ○           ×
              |
              |  =                      ○           ×
              |
              |    =                    ○            ×
    1.5600E¯01|
              |       =                   ○            ×
              |
              |               =                ○         ×
              |
    1.5800E¯01|                                 =  ×        ○
              |
              |   ×                                  ○          =
              |
              |     × ○                                        =
    1.6000E¯01|
              |    ○                          =      ×
              |
              |              =  ○                        ×
              |
    1.6200E¯01|       =             ○                 ×
              |
              |    =                  ○              ×
              |
              |  =                    ○             ×
    1.6400E¯01|
              | =                      ○           ×
              |
              | =                      ○           ×
              |
    1.6600E¯01|=                       ○           ×
              |
              |=                       ○          ×
              |
              |=                       ○          ×
    1.6800E¯01|
              |=                       ○          ×
              |
              |=                       ○          ×
              |
    1.7000E¯01|=---|----|----|----|----○----|----|×---|----|----|
```

arguments. The first plot shows the voltage gain *VG* and its magnitude, for the case of critical coupling. For greater detail in the vicinity of the resonance, a second set of frequencies is interlaced with the first. Then several cases are analyzed at once, by using different values of mutual inductance. The format function SS forces them all to be plotted on the same scale for easy visual comparison. It is seen that mutual inductance greater than 0.01 H leads to a double-peaked response, and smaller values lead to a sharper response.

```
      F←F,.156+0,.0008×ι8
      PLOT SS (MAG VG OF DT .002),(MAG VG OF DT .01),(MAG VG OF DT .02),MAG VG OF DT .1

CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/12/71   12:15


○  MAG VG    0.000E00                 2.500E¯01                5.000E¯01
×  MAG VG    0.000E00                 2.500E¯01                5.000E¯01
=  MAG VG    0.000E00                 2.500E¯01                5.000E¯01
∘  MAG VG    0.000E00                 2.500E¯01                5.000E¯01
 ↓      F     |                        |                        |
    1.5000E¯01○×=--|----|----|---∘|----|----|----|----|----|----|
              |
              ○×=                           ∘
              |
              ○×=                                              ∘
    1.5200E¯01|
              ○× =                                    ∘
              |
              ○ × =                     ∘
              |
    1.5400E¯01○ ×  =             ∘
              |
              |○ ×   =        ∘
              |
              |○   ×     = ∘
    1.5600E¯01|○    ×     ∘  =
              | ○     ×   ∘       =
              | ○        ∘                =
              |  ○       ∘    ×                      =
              |   ○      ∘           ×                         =
    1.5800E¯01|     ○   ∘                       ×              =
              |         ∘○                                 =×
              |         ∘     ○                         =       ×
              |         ∘        ○                    =         ×
              |         ∘    ○                         =       ×
    1.6000E¯01|        ○∘                                 ×=
              |     ○   ∘                     ×                =
              |   ○     ∘          ×                          =
              |  ○      ∘    ×                      =
              | ○       ∘                =
    1.6200E¯01|○      ×  ∘       =
              |○    ×    ∘  =
              |○   ×     ∘
              |
              |○ ×   =     ∘
    1.6400E¯01|
              ○ ×  =         ∘
              |
              ○ × =             ∘
              |
    1.6600E¯01○× =                   ∘
              |
              ○×=                               ∘
              |
              ○×=                                              ∘
    1.6800E¯01|
              ○×=                                  ∘
              |
              ○=                    ∘
              |
    1.7000E¯01○=---|----|----∘----|----|----|----|----|----|----|

      ⍝ END OF EXAMPLE 3.
```

Example 4. Active All-Pass Filter,[3] Figure 2.23. Features illustrated: Logarithmic sweep; *OPAMP*; *WPP*; *ZIN*; *ZOUT*; *PLOG*; *HIGH*. The network is defined, complete with all parameter values, as an APL function. The frequency is set to be 10 raised to the power of a linear sweep; the result is a logarithmic sweep going from 10 Hz to 1 MHz. The idea of an all-pass filter is that the response function (in this case, the voltage gain *VG*) should have a constant magnitude, but the phase should vary with frequency. This is indicated in the printout, where the phase changes a total of 360° while, to 5 significant places, the magnitude is constant. The voltage gain (real and imaginary parts, and magnitude) is plotted next. The same scale is used for all three for easy comparison. The request *PLOG* is made so that the independent variable (here the frequency) is plotted on a logarithmic scale. Also, the plot is requested to be only 25 lines high, to save space.

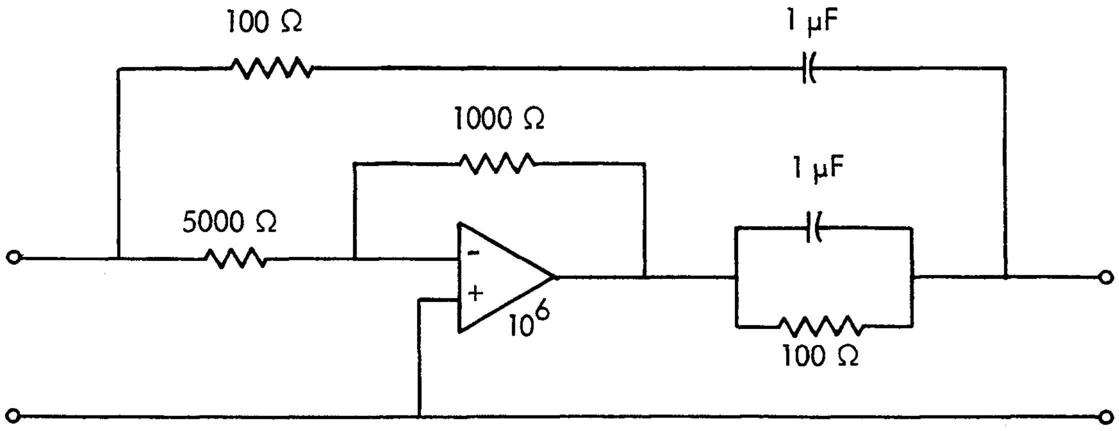
```
      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED  20:52:47 06/30/71

      ZG←.01
      ZL←1E6

      ∇Z←ALLPASS
[1]   Z←(WS(R 100)S C 1E¯6)WPP(WS R 5000)WC((OPAMP ¯1E6)WPP WS R 1000)WC WS(R 100)P C 1E¯6
[2]   ∇

      F←10⋆1+0,.2×⍳25
      F
10 15.84893192 25.11886432 39.81071706 63.09573445 100 158.4893192 251.1886432 398.1071706 630.9573445
      1000 1584.893192 2511.886432 3981.071706 6309.573445 10000 15848.93192 25118.86432 39810.71706
      63095.73445 100000 158489.3192 251188.6432 398107.1706 630957.3445 1000000

      PRINT MAG VG, DEG VG, VG, ZIN, ZOUT OF ALLPASS

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71   15:12
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

      F         MAG VG       DEG VG       RE VG        IM VG        RE ZIN       IM ZIN       RE ZOUT      IM ZOUT
 ----------- ----------- ----------- ----------- ----------- ----------- ----------- ----------- -----------
  1.0000E01   1.9998E¯01  1.7784E02  ¯1.9984E¯01  7.5366E¯03  4.3625E03  ¯1.6365E03   9.9980E01  ¯1.2563E00
  1.5849E01   1.9998E¯01  1.7658E02  ¯1.9962E¯01  1.1939E¯02  3.6648E03  ¯2.1628E03   9.9950E01  ¯1.9904E00
  2.5119E01   1.9998E¯01  1.7458E02  ¯1.9908E¯01  1.8899E¯02  2.6338E03  ¯2.4188E03   9.9876E01  ¯3.1515E00
  3.9811E01   1.9998E¯01  1.7141E02  ¯1.9774E¯01  2.9864E¯02  1.5836E03  ¯2.2044E03   9.9688E01  ¯4.9826E00
  6.3096E01   1.9998E¯01  1.6641E02  ¯1.9438E¯01  4.6976E¯02  8.5025E02  ¯1.6910E03   9.9222E01  ¯7.8488E00

  1.0000E02   1.9998E¯01  1.5857E02  ¯1.8615E¯01  7.3072E¯02  4.6239E02  ¯1.1695E03   9.8078E01  ¯1.2252E01
  1.5849E02   1.9998E¯01  1.4642E02  ¯1.6660E¯01  1.1061E¯01  2.8665E02  ¯7.7104E02   9.5355E01  ¯1.8714E01
  2.5119E02   1.9998E¯01  1.2820E02  ¯1.2367E¯01  1.5716E¯01  2.1190E02  ¯5.0141E02   8.9347E01  ¯2.7199E01
  3.9811E02   1.9998E¯01  1.0265E02  ¯4.3781E¯02  1.9513E¯01  1.7920E02  ¯3.2947E02   7.8031E01  ¯3.5781E01
  6.3096E02   1.9998E¯01  7.0648E01   6.6268E¯02  1.8868E¯01  1.6104E02  ¯2.2410E02   6.1855E01  ¯4.0246E01

  1.0000E03   1.9998E¯01  3.5601E01   1.6260E¯01  1.1642E¯01  1.4441E02  ¯1.6065E02   4.5660E01  ¯3.8391E01
  1.5849E03   1.9999E¯01  3.1979E¯01  1.9998E¯01  1.1162E¯03  1.2503E02  ¯1.1923E02   3.3427E01  ¯3.3380E01
  2.5119E03   1.9998E¯01 ¯3.4958E01   1.6390E¯01 ¯1.1459E¯01  1.0649E02  ¯8.6690E01   2.4276E01  ¯2.8765E01
  3.9811E03   1.9998E¯01 ¯7.0026E01   6.8311E¯02 ¯1.8795E¯01  9.3780E01  ¯5.9742E01   1.6098E01  ¯2.4604E01
  6.3096E03   1.9998E¯01 ¯1.0211E02  ¯4.1968E¯02 ¯1.9553E¯01  8.7101E01  ¯3.9377E01   9.0585E00  ¯1.9618E01

  1.0000E04   1.9998E¯01 ¯1.2780E02  ¯1.2258E¯01 ¯1.5801E¯01  8.4089E01  ¯2.5325E01   4.3549E00  ¯1.4196E01
  1.5849E04   1.9998E¯01 ¯1.4615E02  ¯1.6608E¯01 ¯1.1140E¯01  8.2825E01  ¯1.6106E01   1.8930E00  ¯9.5680E00
  2.5119E04   1.9998E¯01 ¯1.5839E02  ¯1.8592E¯01 ¯7.3646E¯02  8.2311E01  ¯1.0195E01   7.8246E¯01 ¯6.2122E00
  3.9811E04   1.9998E¯01 ¯1.6630E02  ¯1.9429E¯01 ¯4.7360E¯02  8.2104E01  ¯6.4409E00   3.1633E¯01 ¯3.9661E00
  6.3096E04   1.9998E¯01 ¯1.7134E02  ¯1.9770E¯01 ¯3.0111E¯02  8.2022E01  ¯4.0660E00   1.2672E¯01 ¯2.5144E00

  1.0000E05   1.9998E¯01 ¯1.7453E02  ¯1.9907E¯01 ¯1.9057E¯02  8.1989E01  ¯2.5660E00   5.0573E¯02 ¯1.5895E00
  1.5849E05   1.9998E¯01 ¯1.7655E02  ¯1.9961E¯01 ¯1.2039E¯02  8.1976E01  ¯1.6192E00   2.0153E¯02 ¯1.0037E00
  2.5119E05   1.9998E¯01 ¯1.7782E02  ¯1.9983E¯01 ¯7.5997E¯03  8.1971E01  ¯1.0217E00   8.0263E¯03 ¯6.3347E¯01
  3.9811E05   1.9998E¯01 ¯1.7863E02  ¯1.9992E¯01 ¯4.7960E¯03  8.1969E01  ¯6.4463E¯01  3.1958E¯03 ¯3.9974E¯01
  6.3096E05   1.9998E¯01 ¯1.7913E02  ¯1.9995E¯01 ¯3.0263E¯03  8.1968E01  ¯4.0674E¯01  1.2724E¯03 ¯2.5223E¯01

  1.0000E06   1.9998E¯01 ¯1.7945E02  ¯1.9997E¯01 ¯1.9095E¯03  8.1967E01  ¯2.5663E¯01  5.0655E¯04 ¯1.5915E¯01
```

<u>Example 5</u>. Twin-Tee Filter, Figure 2.24. Features illustrated: Hierarchy of network definitions; Use of formulas and global variables in definitions; *SYMBOLS*. This filter has a transmission zero at frequency[4]

$$\frac{1}{2\pi(1\ \mu\text{F})(2000\ \Omega)} = 79.58\ \text{Hz} \tag{2.8}$$

To illustrate the versatility of using wiring functions, the network is

```
      PLOG 25 HIGH SS MAG VG, VG OF ALLPASS

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71   15:13


o  MAG VG  ¯2.000E¯01                0.000E00                 2.000E¯01
×   RE VG  ¯2.000E¯01                0.000E00                 2.000E¯01
=   IM VG  ¯2.000E¯01                0.000E00                 2.000E¯01
 ↓       F   |                        |                        |
   1.0000E01 ×----|----|----|----|----|=---|----|----|----|----o
             ×                         =                       o
             ×                          =                      o
             ×                            =                    o
             |×                             =                  o
   1.0000E02 | ×                               =               o
             |   ×                                  =          o
             |         ×                                  =    o
             |                   ×                            =o
             |                                ×               =o
   1.0000E03 |                                       =    ×    o
             |                        =                        ×
             |          =                                 ×    o
             | =                               ×               o
             |=                  ×                             o
   1.0000E04 |    =    ×                                       o
             |   ×      =                                      o
             | ×             =                                 o
             |×                 =                              o
             ×                    =                            o
   1.0000E05 ×                      =                          o
             ×                      =                          o
             ×                       =                         o
             ×                       =                         o
             ×                        =                        o
   1.0000E06 ×----|----|----|----|----=----|----|----|----|----o

      ⍝ END OF EXAMPLE 4.
```



<u>Figure 2.23</u>. Active all-pass filter, example 4.

```
      ⍝ BEGINNING OF EXAMPLE 5.

      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED  20:52:47 06/30/71

      ∇Z←TWINTEE
[1]  Z←PATH1 WPP PATH2
[2]  ∇

      ∇Z←PATH1
[1]  Z←(WS R 2×RES1)WC(C MULT×CAP1)WC WS R 2×RES1
[2]  ∇

      ∇Z←PATH2
[1]  Z←(WS C CAP1)WC(R RES1)WC WS C CAP1
[2]  ∇
      MULT←2
      CAP1←1E¯6
      RES1←1000
      F←79,81,50+0,2×⍳25
      F
79 81 50 52 54 56 58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98
     100

      PLOT DB VG OF TWINTEE

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/12/71   15:14
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

○   DB VG  ¯6.000E01                ¯3.500E01                ¯1.000E01
 ↓     F     |                        |                        |
   5.0000E01 |----|----|----|----|----|----|----|----|----|-○--|
             |
             |                                              ○
             |
             |                                             ○
   5.5000E01 |
             |                                            ○
             |
             |                                           ○
             |
   6.0000E01 |                                          ○
             |
             |                                         ○
             |
             |                                        ○
   6.5000E01 |
             |                                      ○
             |
             |                                     ○
             |
   7.0000E01 |                                   ○
             |
             |                                 ○
             |
             |                              ○
   7.5000E01 |
             |                          ○
             |
             |                   ○
             |          ○
   8.0000E01 |       ○
             |                  ○
             |                       ○
             |
             |                            ○
   8.5000E01 |
             |                               ○
             |
             |                                 ○
             |
   9.0000E01 |                                   ○
             |
             |                                    ○
             |
             |                                     ○
   9.5000E01 |
             |                                      ○
             |
             |                                       ○
             |
   1.0000E02 |----|----|----|----|----|----|----|----|----|○---|----|
```

defined as two paths in parallel, and then each path is separately defined. Three APL functions in all are used. In the definition of PATH1, the condition on the resistors is met but the capacitor condition is met only if the global variable *MULT* is equal to 2. With this value of *MULT*, the plot of *VG* expressed in dB shows the transmission zero. Next we want to estimate the effect of variations in the value of the capacitor C2, so we want to plot the characteristics when *MULT* is set to, say, 3 rather than 2. To do this easily, we write a function with an argument entitled *VARYMULT* which, first, sets *MULT* to the

```
      ∇Z←VARYMULT A
[1]  MULT←A
[2]  Z←TWINTEE
[3]  ∇

      PLOT '23' SYMBOLS (DB VG OF VARYMULT 2), SS DB VG OF VARYMULT 3

CIRCUIT ANALYSIS BY MARTHA.   71∘A  7/12/71   15:16


2   DB VG  ¯6.000E01                ¯3.500E01                ¯1.000E01
3   DB VG  ¯6.000E01                ¯3.500E01                ¯1.000E01
↓      F     |                        |                        |
   5.0000E01 |----|----|----|----|----|----|----|----|----3-2--|
             |
             |                                           3  2
             |
             |                                          3  2
   5.5000E01 |
             |                                         3  2
             |
             |                                       3   2
             |
   6.0000E01 |                                      3   2
             |
             |                                     3   2
             |
             |                                   3    2
   6.5000E01 |
             |                                  3   2
             |
             |                                 3   2
             |
   7.0000E01 |                                3  2
             |
             |                                32
             |
             |                              2 3
   7.5000E01 |
             |                          2      3
             |
             |                   2               3
             |          2                        3
   8.0000E01 |       2                            3
             |                  2                 3
             |                       2             3
             |
             |                            2         3
   8.5000E01 |
             |                               2       3
             |
             |                                 2      3
             |
   9.0000E01 |                                   2    3
             |
             |                                    2    3
             |
             |                                     2    3
   9.5000E01 |
             |                                      2   3
             |
             |                                       2   3
             |
   1.0000E02 |----|----|----|----|----|----|----|----|2--3|----|

      ⍝  END OF EXAMPLE 5.
```

value of the argument, and then returns the previously defined network. When an analysis is performed, the computer first looks at *VARYMULT*, then at *TWINTEE*, and then in turn at *PATH*1 and *PATH*2. As a user you have control over this sequence by the way you define your networks to call upon sections that themselves have definitions. In the plot, the plotting characters 2 and 3 were used for the two cases, as directed by the format function *SYMBOLS*.

Example 6. Transistor Audio Amplifier. Features illustrated: Calculation of parameter values; User-defined wiring functions; Local variable; *SAME*; *HYBRIDPI*; Darlington connection; *WN*; *WROT*; *WSS*; *WIDE*. This circuit, Figure 2.25, is based on a student project.[5] The major objectives were: (1) input impedance greater than 30,000 ohms; (2) output impedance less than 100 ohms; (3) mid-band voltage gain between 36 and 44; (4) low-frequency 3-dB point below 20 Hz; (5) high-frequency 3-dB point between 20 and 25 kHz; (6) the amplifier must work with transistors with β between 100 and 400. The circuit uses a Darlington transistor connection for the input stage, to get high input impedance. There is no wiring function corresponding to that connection in *MARTHA*, but it can be defined in terms of the functions *WC*, *WN*, and *WROT*. The user-defined wiring function *DARLINGTON is* defined first. Then the amplifier is defined as a function of β. In the hybridpi model, the value of r_π is equal to β/g_m, and g_m depends upon the collector bias current. The collector biases for the network led to the four transistors having the values of g_m indicated, and in the definitions of *Q*1, *Q*2, *Q*3, and *Q*4 an expression for r_π is used, since this depends on β. The last two lines of the function *AMP* define the network. A logarithmic plot of voltage gain is made first; it is specified as 40 lines high to match the frequency band of the amplifier. The gain falls within the specifications, and next we look at the input and output impedances. The network that has most recently been analyzed is saved, for your convenience, so that you can look at different response functions without typing its definition again. It is named *SAME*, and is generally available until the first wiring function is used. Thus the

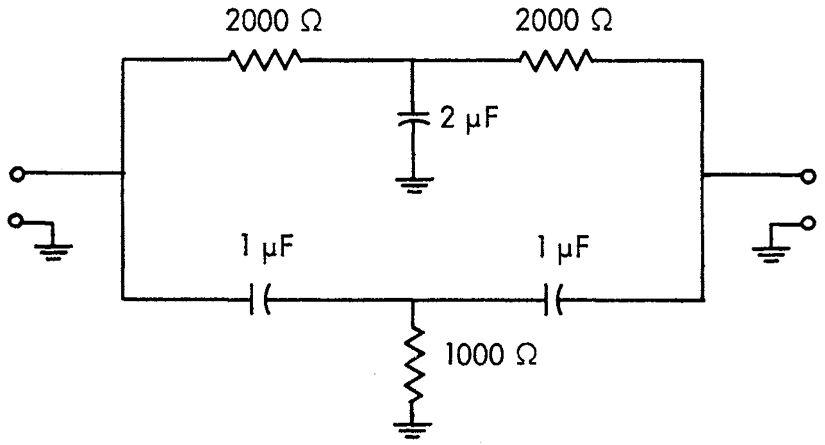


Figure 2.24. Twin-tee filter with nominal values, example 5. In the analysis the upper capacitor is either 2μF or 3μF.

next plot is requested using *SAME* as the name of the network. The impedance levels are well within specifications. Now to check the effect of variations in β on gain, a plot is made of three cases: beta equal to 100, 200, and 400. For maximum visibility the width is set large (100 characters) but the height need not be greater than 20. With this

```
      ⍝ BEGINNING OF EXAMPLE 6.

      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED  20:52:47 06/30/71

      ∇Z←A DARLINGTON B
[1]  Z←WN WROT (WN WROT A) WC WN WROT B
[2]  ∇

      ∇Z←AMP BETA
[1]  Q1←HYBRIDPI 0,(BETA÷.0006),200E¯12,4E¯12,.0006
[2]  Q2←HYBRIDPI 0,(BETA÷.044),200E¯12,4E¯12,.044
[3]  Q3←HYBRIDPI 0,(BETA÷.04),200E¯12,4E¯12,.04
[4]  Q4←HYBRIDPI 0,(BETA÷.05),200E¯12,4E¯12,.05
[5]  Z←(WS C .2E¯6)WC((R 100E3)P R 370E3)WC((R 1800)WSS Q1 DARLINGTON Q2)WC(C 820E¯12)P R 10E3
[6]  Z←Z WC (Q3 WSS (R 630)S(C 100E¯6)P R 3300)WC(R 5600)WC(WN WROT Q4)WC R 6800
[7]  ∇

      F←10★1+0,.2×⍳20
      PLOG 40 HIGH VG, MAG VG OF AMP 100

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   9:29
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

○   RE VG    0.000E00                2.000E01                4.000E01
×   IM VG   ¯2.000E01                5.000E00                3.000E01
=  MAG VG    0.000E00                2.000E01                4.000E01
 ↓      F     |                        |                        |
    1.0000E01 |----|----|----|--○-|----|----|---=|----|--×-|----|
              |
              |                              ○         =
              |
              |                                   ×    ○    =
    3.1623E01 |
              |                              ×              ○ =
              |
              |                          ×                    ○=
              |
    1.0000E02 |                        ×                       =
              |
              |                      ×                         =
              |
              |                    ×                           =
    3.1623E02 |
              |                   ×                            =
              |
              |                   ×                            =
              |
    1.0000E03 |                  ×                             =
              |
              |                ×                               =
              |
              |              ×                                 =
    3.1623E03 |
              |            ×                                  ○=
              |
              |        ×                                   ○ =
              |
    1.0000E04 |    ×                                  ○    =
              |
              |×                             ○       =
              |
              ×                   ○           =
    3.1623E04 |
              |   ×     ○            =
              |
              |    ○  ×       =
              |
    1.0000E05 |-○--|----=-×--|----|----|----|----|----|----|----|
```

scale, 3 dB is 10 spaces, and the allowed limits on mid-band gain, namely 31.13 dB and 32.97 dB, are 6 characters apart. Finally, the 3-dB points can be determined from the last printout, at least for the case with β = 100.

```
      PLOG 20 HIGH DB VG, MAG ZIN, MAG ZOUT OF SAME

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71  9:29

○   DB VG    1.000E01                2.500E01                4.000E01
×  MAG ZIN   6.000E04                9.000E04                1.200E05
=  MAG ZOUT  7.200E01                7.350E01                7.500E01
 ↓      F     |                        |                        |
    1.0000E01 |----|----|----|----|----|----|○---|----=--×-|----|
              |                           ×     ○     =
              |                   ×              ○    =
              |                ×                  ○   =
              |               ×                   ○   =
    1.0000E02 |               ×                   ○   =
              |              ×                     ○  =
              |              ×                     ○  =
              |              ×                     ○  =
              |              ×                     ○  =
    1.0000E03 |              ×                     ○  =
              |              ×                     ○  =
              |              ×                    ○   =
              |              ×                    ○  =
              |             ×                     ○ =
    1.0000E04 |            ×                    =○
              |          ×                ×    ○
              |       ×            =        ○
              |     ×          =        ○
              |    ×            = ○
    1.0000E05 |---×|----|---○|----|----|--=-|----|----|----|----|
```

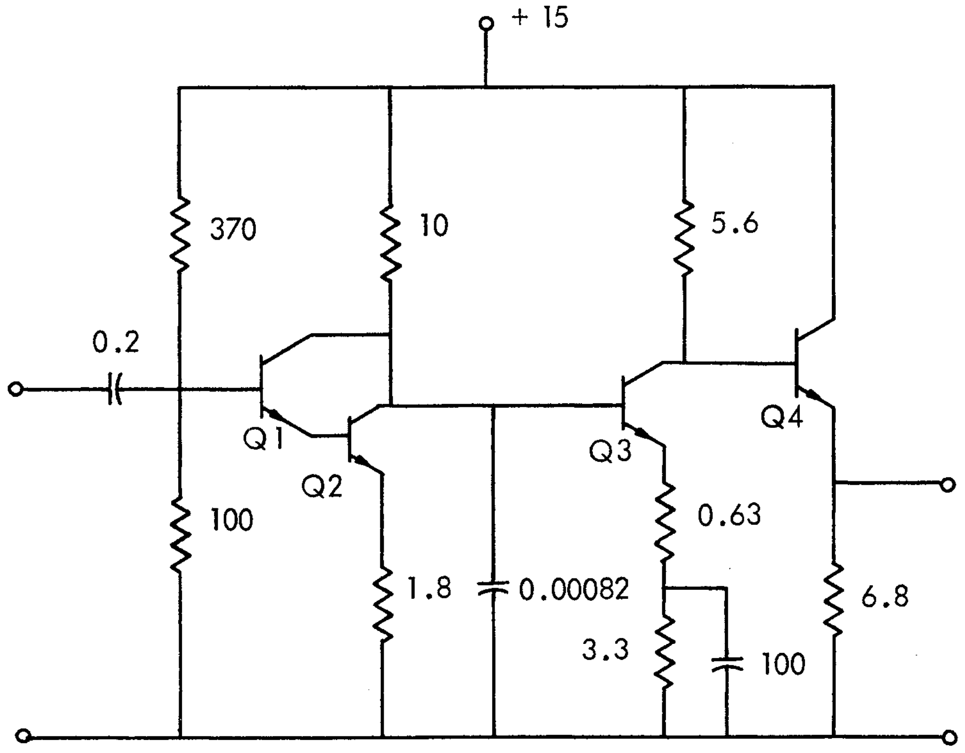


Figure 2.25. Four-transistor amplifier, example 6. Values in kΩ and µF.

Example 7. Microstrip Bandpass Filter[6], Figure 2.26. Features illustrated: *MARTHA* library; *DEGREESAT*; *TEM*; *ZSCALE*; *WTO*; *WTS*; *S*21; *VSWRIN*. Although this filter was ultimately realized in microstrip, the design used electrical parameters, and therefore *MARTHA*'s ability to calculate characteristic impedance of microstrip is not used. First, all transmission-line elements that are distinct are defined; there is no need to define identical elements more than once, although you can if you wish. The lengths are specified in electrical degrees at a

```
      PLOG 20 HIGH 100 WIDE (DB VG OF AMP 100),(DB VG OF AMP 200),DB VG OF AMP 400

CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/13/71   9:35


o   DB VG   1.000E01                                          2.500E01                                          4.000E01
×   DB VG   1.000E01                                          2.500E01                                          4.000E01
=   DB VG   1.000E01                                          2.500E01                                          4.000E01
 ↓    F      |                                                 |                                                 |
   1.0000E01 |----|----|----|----|----|----|----|----|----|----|----|----|-o-x=----|----|----|----|----|----|----|
             |                                                                   o ×=
             |                                                                      o ×=
             |                                                                       o  ×=
             |                                                                        o ×=
   1.0000E02 |                                                                        o × =
             |                                                                        o × =
             |                                                                        o × =
             |                                                                        o × =
             |                                                                        o × =
   1.0000E03 |                                                                        o × =
             |                                                                        o × =
             |                                                                        o ×=
             |                                                                        o ×=
             |                                                                       o ×=
   1.0000E04 |                                                                     o ×=
             |                                                                  o×=
             |                                                           o×=
             |                                                  o=
             |                                       ×=
   1.0000E05 |----|----|----|----|----|-o=-|----|----|----|----|----|----|----|----|----|----|----|----|----|----|

     PRINT DB VG, MAG VG OF AMP 100

CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/13/71   9:35


      F          DB VG       MAG VG
 ----------- ----------- -----------
  1.0000E01   2.8596E01   2.6904E01
  1.5849E01   3.0341E01   3.2888E01
  2.5119E01   3.1243E01   3.6488E01
  3.9811E01   3.1653E01   3.8251E01
  6.3096E01   3.1826E01   3.9022E01

  1.0000E02   3.1897E01   3.9340E01
  1.5849E02   3.1925E01   3.9468E01
  2.5119E02   3.1936E01   3.9518E01
  3.9811E02   3.1940E01   3.9535E01
  6.3096E02   3.1939E01   3.9532E01

  1.0000E03   3.1934E01   3.9509E01
  1.5849E03   3.1919E01   3.9442E01
  2.5119E03   3.1882E01   3.9275E01
  3.9811E03   3.1791E01   3.8863E01
  6.3096E03   3.1568E01   3.7881E01

  1.0000E04   3.1056E01   3.5710E01
  1.5849E04   2.9985E01   3.1568E01
  2.5119E04   2.8084E01   2.5364E01
  3.9811E04   2.5319E01   1.8448E01
  6.3096E04   2.1928E01   1.2485E01

  1.0000E05   1.8195E01   8.1235E00

∩ END OF EXAMPLE 6.
```

reference frequency rather than as a physical length. The network definition is simplified by noting two things. First, the filter is symmetrical, so we can define the left half, and then use the wiring function *WN* to interchange the input and output and the result is the right half (the line in the middle is treated separately). The left half is defined in the first line, and then is used twice in the second line. Second, the open and short stubs lie in matched pairs on either side of the main line, so that the relatively low impedance levels called for can be realized in microstrip. Two identical networks in parallel can be simulated by one of them, with impedance scaled by a factor of 0.5. This is usually easier than writing two

```
      ⍝  BEGINNING OF EXAMPLE 7

      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED  20:52:47 06/30/71

      ZG←50
      ZL←50
      T1←TEM 100,90 DEGREESAT 9.5E9
      T2←TEM 49.3,90 DEGREESAT 9.5E9
      T3←TEM 36,180 DEGREESAT 9.5E9
      T4←TEM 28.2,180 DEGREESAT 9.5E9
      T5←TEM 70.7,90 DEGREESAT 9.5E9

      ∇Z←FILTER
[1]  Z←T5 WC(.5 ZSCALE WTS T2)WC T1 WC(.5 ZSCALE WTO T3)WC T1
[2]  Z←Z WC(.5 ZSCALE WTO T4)WC WN Z
[3]  ∇

      F←1E9×8+0,.1×⍳30
      ZNIN←50
      ZNOUT←50
      PLOT 30 HIGH DB S21, DEG S21 OF FILTER

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   9:29
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

o   DB S21  ¯4.000E01                ¯2.000E01                 0.000E00
×  DEG S21  ¯2.000E02                 0.000E00                 2.000E02
 ↓      F     |                        |                        |
     8.0000E09|----|×-o-|----|----|----|----|----|----|----|----|
              |    ×      o
              |  ×             o
              |                     o                        ×
              |                          o                 ×
     8.5000E09|                               o          ×
              |                                     o ×
              |                                   ×       o
              |                              ×                o
              |                        ×                        o
     9.0000E09|                    ×                            o
              |                ×                                o
              |            ×                                    o
              |        ×                                        o
              |     ×                                           o
     9.5000E09| ×                                               o
              |                                           ×     o
              |                                        ×        o
              |                                    ×            o
              |                                ×                o
     1.0000E10|                            ×                    o
              |                        ×                        o
              |                  ×                            o
              |             ×                             o
              |         ×                           o
     1.0500E10|      ×                        o
              |    ×                     o
              |  ×                  o
              |                o                             ×
              |           o                                ×
     1.1000E10|----|--o-|----|----|----|----|----|----|---×|----|
```

networks separately. The first plot shows the transfer ratio *S*21 in dB, and its phase in degrees. The passband is approximately between 9 and 10 GHz, so let's look in more detail at the passband. One of the useful parameters for passive filters is the voltage standing wave ratio at the input. This response function, like many others, is probably too specialized to justify its taking up space in *MARTHA*. However, it is in the *MARTHA* library, in the workspace 100 *MARTHAR*,

```
      F←1E9×9+.04×0,ι25
      )COPY 100 MARTHAR VSWRIN
SAVED  12:25:05 07/01/71

      PLOT 'VG' SYMBOLS VSWRIN, DB IG OF FILTER


CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   9:36


V   VSWRIN   1.000E00                 1.150E00                 1.300E00
G   DB IG   ¯4.000E¯002              ¯2.000E¯002               0.000E00
  ↓     F     |                        |                        |
    9.0000E09 |--G-|----|----|----|----|----|---V|----|----|----|
              |
              |                    V         G
              |
              |            V                            G
    9.1000E09 |
              |        V                                    G
              |
              |      V                                        G
              |
    9.2000E09 |     V                                         G
              |
              |     V                                         G
              |
              |     V                                         G
    9.3000E09 |
              |    V                                           G
              |
              |    V                                           G
              |
    9.4000E09 |   V                                            G
              |
              | V                                               G
              |
              |V                                                G
    9.5000E09 |
              |V                                                G
              |
              | V                                               G
              |
    9.6000E09 |   V                                            G
              |
              |    V                                           G
              |
              |    V                                           G
    9.7000E09 |
              |     V                                         G
              |
              |     V                                         G
              |
    9.8000E09 |     V                                         G
              |
              |      V                                        G
              |
              |        V                                    G
    9.9000E09 |
              |            V                            G
              |
              |                    V         G
              |
    1.0000E10 |--G-|----|----|----|----|----|---V|----|----|----|

      ⍝ END OF EXAMPLE 7.
```

and you can copy it any time you need it. The plot shows the *VSWR* and insertion gain in the passband.

Example 8. Coaxial Bandpass Filter, Figure 2.27. Features illustrated: *COAX; COAXDISCAP; FORDIEL; TITLE*. In the design of this filter,[7] the coaxial discontinuity capacitances played a major role.

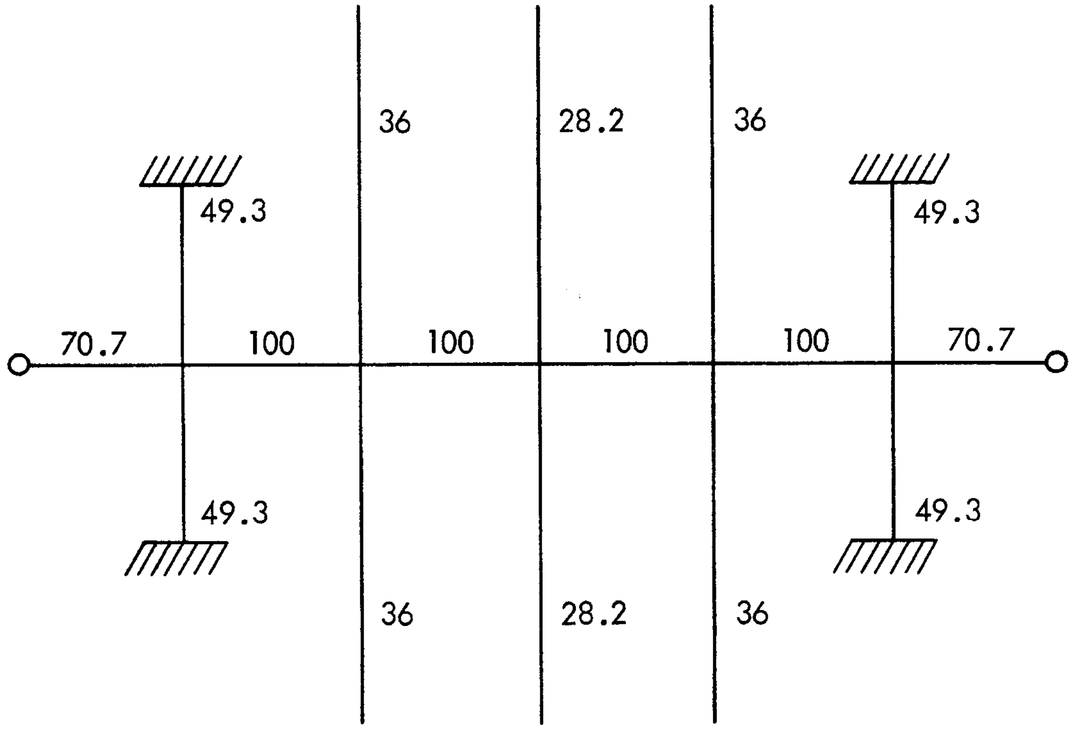


Figure 2.26. Bandpass filter for microstrip, example 7. All lines are either λ/4 or λ/2 at the center frequency, 9.5 GHz. The impedances of the lines, in ohms, are indicated.

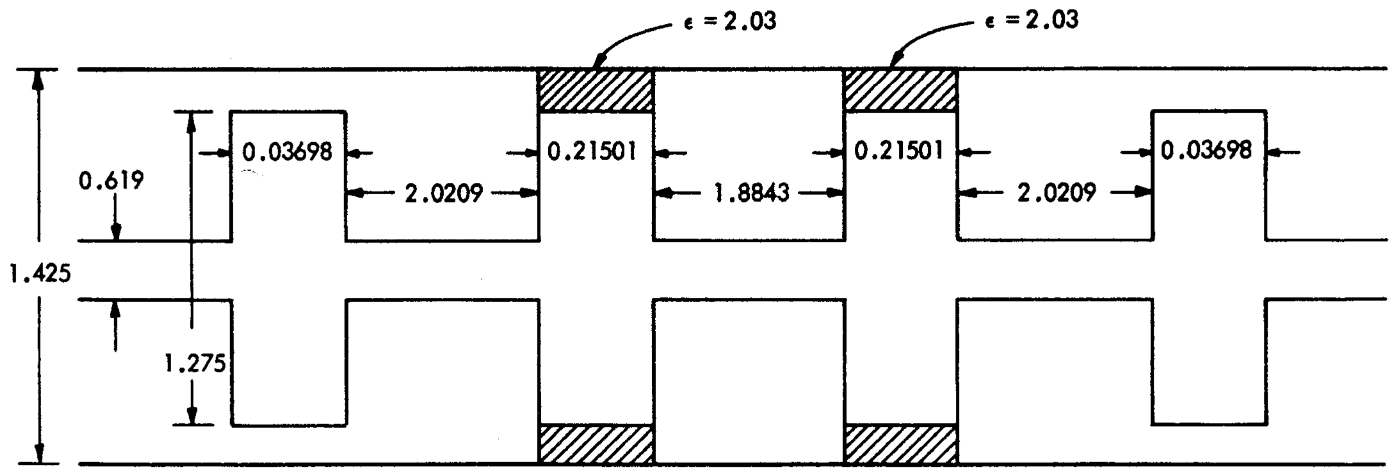


Figure 2.27. Coaxial bandpass filter, example 8. Dimensions in cm. All the disks have the same diameter. The dielectric is Teflon, and does not influence the discontinuity capacitances because it is on the narrow side of each discontinuity.

It is interesting to estimate their importance by analyzing the filter structure, and then repeating the analysis without them. The function *COAXDISCAP*, which is kept in the *MARTHA* library, in workspace 100 *MARTHAX*, calculates the capacitance when the radii are given. The filter definition has as its first line the definition of this capacitor. The filter is symmetric, so its left side is defined in the second line of the function *FILTER*, and then is used twice in the third line. Note that the function *COAX* (discussed in Section 4.5 of this manual) does not produce a transmission line, but merely calculates the charac-

```
      ⍝ BEGINNING OF EXAMPLE 8

      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED  20.52.47 06/30/71
      )COPY 100 MARTHAX COAX
SAVED   12.38.36 07/01/71
      )COPY 100 MARTHAX COAXDISCAP
SAVED   12.38.36 07/01/71

      ∇Z←FILTER
[1]   Z←C COAXDISCAP 0.5×.01×1.425,.619,1.275
[2]   Z←Z WC(TEM COAX .7125,.6375,.0003698)WC Z WC(TEM 50,.020209)WC Z WC(TEM COAX .7125,.6375,.0021501 FORDIEL 2.03)WC Z
[3]   Z←Z WC(TEM 50,.018843)WC WN Z
[4]   ∇

      F←1E9×7+0,.1×⍳30
      ZG←50
      ZL←50
      PLOT 30 HIGH DB IG OF FILTER8

CIRCUIT ANALYSIS BY̲ MARTHA.   71∘A   7/13/71   9:29
MARTHA COPYRIGHT (C̲) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

○   DB IG   ¯4.000E01                ¯2.000E01                 0.00E00
 ↓      F     |                        |                        |
    7.0000E09 |----|----|----o----|----|----|----|----|----|----|
              |               o
              |                 o
              |                   o
              |                     o
    7.5000E09 |                        o
              |                           o
              |                              o
              |                                  o
              |                                       o
    8.0000E09 |                                            o
              |                                                o
              |                                                 o
              |                                                 o
              |                                                 o
    8.5000E09 |                                                 o
              |                                                 o
              |                                                 o
              |                                                 o
              |                                                 o
    9.0000E09 |                                            o
              |                                      o
              |                               o
              |                          o
              |                      o
    9.5000E09 |                  o
              |              o
              |           o
              |        o
              |      o
    1.0000E10 |---o|----|----|----|----|----|----|----|----|----|

       ⍝ HOW LARGE IS THE COAXIAL DISCONTINUITY CAPACITANCE?
       COAXDISCAP .5×.01×1.425,.619,1.1275
3.350146329E¯13
       ⍝ APPROXIMATELY 0.335 PF.
      ∇Z←WITHOUT
[1]   Z←(TEM COAX 0.7125 0.6375 3.698E¯4)WC(TEM 50 0.020209)WC TEM COAX 0.7125 0.6375 ,2.1501E¯3 FORDIEL 2.03
[2]   Z←Z WC(TEM 50 0.018843)WC WN Z
[3]   ∇

      PLOT SS 30 HIGH (DB IG OF FILTER),DB IG OF WITHOUT

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   10:6


o   DB IG   ¯4.000E01                ¯2.000E01                 0.000E00
×   DB IG   ¯4.000E01                ¯2.000E01                 0.000E00
 ↓      F     |                        |                        |
    7.0000E09 |----|----|----o----|----×----|----|----|----|----|
              |               o         ×
              |                 o       ×
              |                   o      ×
              |                     o     ×
    7.5000E09 |                        o   ×
              |                           o ×
              |                              ×
              |                                × o
              |                                 ×     o
    8.0000E09 |                                   ×        o
              |                                     ×          o
              |                                       ×         o
              |                                          ×      o
              |                                            ×    o
    8.5000E09 |                                               × o
              |                                                 ×
              |                                                ×o
              |                                               × o
              |                                            ×    o
    9.0000E09 |                                          × o
              |                                      o ×
              |                               o      ×
              |                          o          ×
              |                      o            ×
    9.5000E09 |                  o               ×
              |              o                  ×
              |           o                   ×
              |        o                     ×
              |      o                     ×
    1.0000E10 |---o|----|----|----|----|--×-|----|----|----|----|

      F←1E9×8+0,.05×⍳20

      )COPY 100 MARTRAR VSWRIN
SAVED   12.25.05 07/01/71

      TITLE←'   FILTER WITH COAXIAL DISCONTINUITY CAPACITANCE ACCOUNTED FOR'

      PLOT 20 HIGH VSWRIN, MAG S21 OF FILTER

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   10:13
   FILTER WITH COAXIAL DISCONTINUITY CAPACITANCE ACCOUNTED FOR

o   VSWRIN   1.000E00                5.000E00                9.000E00
×  MAG S21   6.00E¯01                8.00E¯01                1.000E00
 VS     F     |                        |                        |
    8.0000E09 |--×-|----|----|----|----|----|----|----|----o----|
              |                    ×o
              |         o                          ×
              |  o                                           ×
              o                                                 ×
    8.2500E09 |o                                               ×
              |  o                                          ×
              |  o                                          ×
              |  o                                          ×
              | o                                             ×
    8.5000E09 |o                                                ×
              |o                                                ×
              | o                                               ×
              |  o                                          ×
              |  o                                         ×
    8.7500E09 |  o                                          ×
              | o                                             ×
              |o                                                ×
              |    o                                      ×
              |               o           ×
    9.0000E09 |----×----|----|----|----|----|----|----|o---|----|
```

teristic impedance for *TEM*. The first graph shows the insertion gain of the filter, with a passband between 8 and 9 GHz. To estimate the importance of considering the discontinuity capacitances, the function *WITHOUT* defines exactly the same filter, without these capacitors. The next graph shows both insertion gains plotted on the same scale, and the importance of the capacitors is obvious. Not let's look at the *VSWR* at the input, and the transfer ratio *S*21 in the passband. Let's suppose this graph, and the following one, are for a report or proposal, and we want to indicate on them what the assumptions are. This is done conveniently with the variable *TITLE*, which is normally blank but can be set to any text. It then gets printed out, once, in the next printout. The last graph is similar, for the case without the discontinuity capacitors.

Example 9. Waveguide Matching Filter,[8] Figure 2.28. Features illustrated: Response of one network plotted against response of another; *WG*; *WT*; *Y*; *RECT*1. The waveguide element *WG* is valid for any shape waveguide. For rectangular guide, there are two functions, *RECT*1 and *RECT*2, in the *MARTHA* library to calculate cutoff frequency and characteristic impedance from the guide dimensions; these differ in the formula for characteristic impedance as a function of width. In this example, the width is constant and either formula can be used. The input and output guides are not given lengths, and therefore *WGO* and *WG*4 are waveguide characteristic impedances. The first printout gives the characteristic impedance of each, to indicate the magnitude of the matching problem. Below cutoff, the characteristic impedances are imaginary, and above cutoff they are real. The values in the passband of interest are shown in the graph of characteristic admit-

```
      TITLE←'   FILTER WITH COAXIAL DISCONTINUITY CAPACITANCE OMITTED'

      PLOT 20 HIGH VSWRIN, MAG S21 OF WITHOUT


CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   10:13
   FILTER WITH COAXIAL DISCONTINUITY CAPACITANCE OMITTED


○   VSWRIN   0.000E00                3.000E01                6.000E01
×  MAG S21   2.00E¯01                6.00E¯01                1.000E00
 ↓     F     |                        |                        |
    8.0000E09 |----×----|----|----|----|----|----|----|--○-|----|
              |     ×                             ○
              |       ×                     ○
              |         ×             ○
              |            ×     ○
    8.2500E09 |              ×
              |          ○       ×
              |       ○              ×
              |     ○                     ×
              |   ○                             ×
    8.5000E09 |  ○                                    ×
              | ○                                           ×
              |○                                               ×
              |○                                                ×
              | ○                                            ×
    8.7500E09 | ○                                       ×
              |  ○                                 ×
              |    ○                          ×
              |     ○                     ×
              |       ○               ×
    9.0000E09 |----|----○----|----|×---|----|----|----|----|----|

      ⍝ END OF EXAMPLE 8.
      ⍝ BEGINNING OF EXAMPLE 9

      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED   20:52:47 06/30/71
      )COPY 100 MARTHAX RECT1
SAVED   12:38:36 07/01/71

      WG0←WG RECT1 .0254×6.5,1.3
      WG1←WG RECT1 .0254×6.5,1.479,3.093
      WG2←WG RECT1 .0254×6.5,2.057,3.085
      WG3←WG RECT1 .0254×6.5,2.857,3.175
      WG4←WG RECT1 .0254←6.5,3.25

      F←.1E9×⍳20
      PRINT (Z OF WG0),Z OF WG4

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   10:18
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

      F         RE Z        IM Z        RE Z        IM Z
 ----------- ----------- ----------- ----------- -----------
  1.0000E08   0.0000E00   1.6699E01   0.0000E00   4.1748E01
  2.0000E08   0.0000E00   3.4031E01   0.0000E00   8.5078E01
  3.0000E08   0.0000E00   5.2756E01   0.0000E00   1.3189E02
  4.0000E08   0.0000E00   7.3955E01   0.0000E00   1.8489E02
  5.0000E08   0.0000E00   9.9424E01   0.0000E00   2.4856E02

  6.0000E08   0.0000E00   1.3269E02   0.0000E00   3.3173E02
  7.0000E08   0.0000E00   1.8244E02   0.0000E00   4.5610E02
  8.0000E08   0.0000E00   2.8081E02   0.0000E00   7.0203E02
  9.0000E08   0.0000E00   1.1340E03   0.0000E00   2.8349E03
  1.0000E09   3.5951E02   0.0000E00   8.9877E02   0.0000E00

  1.1000E09   2.6691E02   0.0000E00   6.6727E02   0.0000E00
  1.2000E09   2.3046E02   0.0000E00   5.7614E02   0.0000E00
  1.3000E09   2.1055E02   0.0000E00   5.2637E02   0.0000E00
  1.4000E09   1.9796E02   0.0000E00   4.9491E02   0.0000E00
  1.5000E09   1.8931E02   0.0000E00   4.7327E02   0.0000E00

  1.6000E09   1.8301E02   0.0000E00   4.5752E02   0.0000E00
  1.7000E09   1.7824E02   0.0000E00   4.4560E02   0.0000E00
  1.8000E09   1.7452E02   0.0000E00   4.3630E02   0.0000E00
  1.9000E09   1.7154E02   0.0000E00   4.2886E02   0.0000E00
  2.0000E09   1.6912E02   0.0000E00   4.2281E02   0.0000E00
```

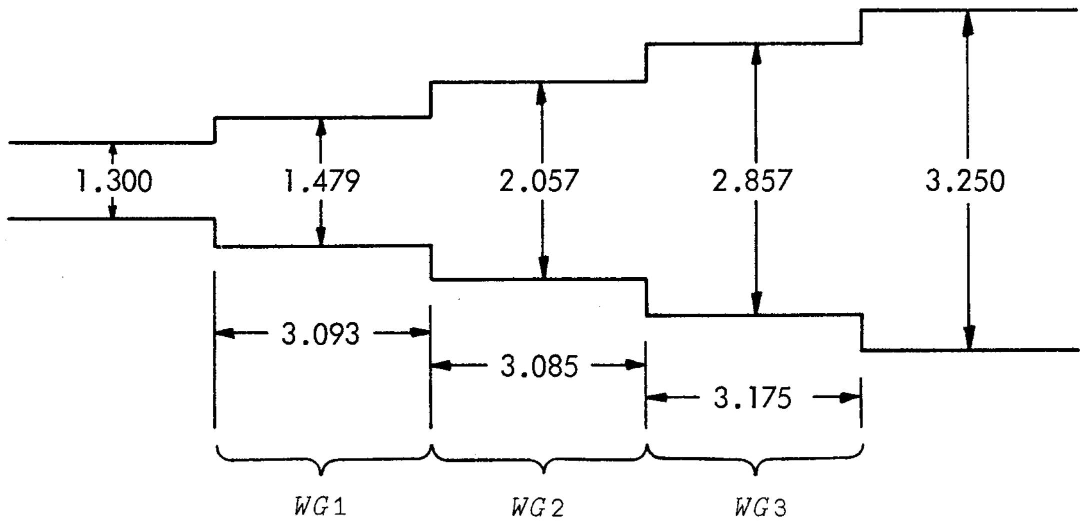


Figure 2.28. Waveguide matching filter, example 9. Dimensions in inches. The guides all have the same width, 6.500 inches, and therefore the same cutoff frequency.

tance. Next the filter, which has a very simple definition, is defined. The actual lengths were calculated accounting for fringing near the junctions, but in this analysis we have neglected this effect. A good match is indicated if the admittance of the filter, when terminated by *WG*4, is close to the characteristic admittance of *WGO*. This is indicated in the printout, and then there is an expanded plot indicating the admittance of *WG*4, the admittance of *WGO*, and the admittance of the filter. The latter two coincide over the passband. A useful way of seeing them coincide is to plot one against the other; this is done in the final graph. The × symbols are a plot of the characteristic admittance against itself, and therefore should lie on a straight line. The circles are the filter input, which is seen to virtually coincide over the passband. If we had included the fringing, the agreement would have been better still.

2.14 In Case of Trouble

Unless you are somehow better than the rest of us, you will make mistakes, especially when you first learn to use *MARTHA*. There are numerous error messages to help you diagnose your problem. *MARTHA* was

```
      F←1E9×1+.02×ι30
      PLOT SS 30 HIGH (Y OF WG0), Y OF WG4

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   10:20

○   RE Y     0.000E00                 3.00E¯03                 6.00E¯03
×   IM Y     0.000E00                 3.00E¯03                 6.00E¯03
=   RE Y     0.000E00                 3.00E¯03                 6.00E¯03
∘   IM Y     0.000E00                 3.00E¯03                 6.00E¯03
 ↓      F     |                        |                        |
    1.0000E09 |----|----|----|----|----|----|----|----|----|----|
              ∘         =              ○
              ∘          =               ○
              ∘          =                 ○
              ∘           =                 ○
    1.1000E09 ∘           =                  ○
              ∘            =                  ○
              ∘            =                   ○
              ∘             =                   ○
              ∘             =                    ○
    1.2000E09 ∘             =                     ○
              ∘              =                     ○
              ∘              =                      ○
              ∘              =                      ○
              ∘               =                      ○
    1.3000E09 ∘               =                       ○
              ∘               =                       ○
              ∘               =                        ○
              ∘               =                        ○
              ∘                =                        ○
    1.4000E09 ∘                =                        ○
              ∘                =                         ○
              ∘                =                         ○
              ∘                =                         ○
              ∘                =                          ○
    1.5000E09 ∘                 =                         ○
              ∘                 =                         ○
              ∘                 =                          ○
              ∘                 =                          ○
              ∘                 =                          ○
    1.6000E09 ∘----|----|----|--=-|----|----|----|----|----|○---|

      ∇Z←FILTER
[1]   Z←WG1 WC WG2 WC WG3
[2]   ∇
```

```
      PRINT (Y OF WG0),MAG Y, DEG Y, Y OF FILTER WT WG4

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   10:23

      F          RE Y        IM Y       MAG Y       DEG Y        RE Y        IM Y
 ----------- ----------- ----------- ----------- ----------- ----------- -----------
  1.0200E09   3.0244E¯03  0.0000E00   3.6517E¯03  1.4038E01   3.5426E¯03  8.8578E¯04
  1.0400E09   3.2366E¯03  0.0000E00   3.8879E¯03  8.9605E00   3.8405E¯03  6.0556E¯04
  1.0600E09   3.4249E¯03  0.0000E00   4.0130E¯03  5.1398E00   3.9969E¯03  3.5952E¯04
  1.0800E09   3.5938E¯03  0.0000E00   4.0723E¯03  2.5515E00   4.0683E¯03  1.8129E¯04
  1.1000E09   3.7466E¯03  0.0000E00   4.1024E¯03  1.0024E00   4.1017E¯03  7.1767E¯05

  1.1200E09   3.8857E¯03  0.0000E00   4.1277E¯03  2.3626E¯01  4.1277E¯03  1.7021E¯05
  1.1400E09   4.0132E¯03  0.0000E00   4.1622E¯03 ¯4.1696E¯03  4.1622E¯03 ¯3.0290E¯07
  1.1600E09   4.1304E¯03  0.0000E00   4.2119E¯03  5.4113E¯02  4.2119E¯03  3.9779E¯06
  1.1800E09   4.2388E¯03  0.0000E00   4.2774E¯03  2.2947E¯01  4.2774E¯03  1.7131E¯05
  1.2000E09   4.3392E¯03  0.0000E00   4.3560E¯03  3.9108E¯01  4.3559E¯03  2.9732E¯05

  1.2200E09   4.4327E¯03  0.0000E00   4.4428E¯03  4.5828E¯01  4.4426E¯03  3.5535E¯05
  1.2400E09   4.5198E¯03  0.0000E00   4.5321E¯03  3.9559E¯01  4.5320E¯03  3.1290E¯05
  1.2600E09   4.6014E¯03  0.0000E00   4.6182E¯03  2.0488E¯01  4.6182E¯03  1.6514E¯05
  1.2800E09   4.6778E¯03  0.0000E00   4.6965E¯03 ¯8.4053E¯02  4.6965E¯03 ¯6.8898E¯06
  1.3000E09   4.7495E¯03  0.0000E00   4.7638E¯03 ¯4.2352E¯01  4.7637E¯03 ¯3.5213E¯05

  1.3200E09   4.8170E¯03  0.0000E00   4.8189E¯03 ¯7.5780E¯01  4.8185E¯03 ¯6.3733E¯05
  1.3400E09   4.8807E¯03  0.0000E00   4.8625E¯03 ¯1.0321E00   4.8617E¯03 ¯8.7589E¯05
  1.3600E09   4.9408E¯03  0.0000E00   4.8974E¯03 ¯1.2003E00   4.8963E¯03 ¯1.0259E¯04
  1.3800E09   4.9976E¯03  0.0000E00   4.9277E¯03 ¯1.2307E00   4.9266E¯03 ¯1.0584E¯04
  1.4000E09   5.0514E¯03  0.0000E00   4.9587E¯03 ¯1.1101E00   4.9578E¯03 ¯9.6068E¯05

  1.4200E09   5.1024E¯03  0.0000E00   4.9960E¯03 ¯8.4565E¯01  4.9954E¯03 ¯7.3735E¯05
  1.4400E09   5.1509E¯03  0.0000E00   5.0450E¯03 ¯4.6536E¯01  5.0448E¯03 ¯4.0975E¯05
  1.4600E09   5.1969E¯03  0.0000E00   5.1109E¯03 ¯1.6333E¯02  5.1109E¯03 ¯1.4569E¯06
  1.4800E09   5.2407E¯03  0.0000E00   5.1978E¯03  4.3821E¯01  5.1977E¯03  3.9753E¯05
  1.5000E09   5.2824E¯03  0.0000E00   5.3087E¯03  8.2328E¯01  5.3081E¯03  7.6277E¯05

  1.5200E09   5.3222E¯03  0.0000E00   5.4448E¯03  1.0573E00   5.4439E¯03  1.0047E¯04
  1.5400E09   5.3601E¯03  0.0000E00   5.6058E¯03  1.0579E00   5.6049E¯03  1.0350E¯04
  1.5600E09   5.3964E¯03  0.0000E00   5.7890E¯03  7.4725E¯01  5.7885E¯03  7.5498E¯05
  1.5800E09   5.4310E¯03  0.0000E00   5.9892E¯03  5.7721E¯02  5.9892E¯03  6.0336E¯06
  1.6000E09   5.4642E¯03  0.0000E00   6.1983E¯03 ¯1.0628E00   6.1972E¯03 ¯1.1496E¯04
```

```
      PLOT 30 HIGH 100 WIDE SS (RE Y OF WG0),(RE Y OF WG4), MAG Y OF FILTER WT WG4

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71  10:27

○   RE Y     1.000E¯03                                         4.000E¯03                                         7.000E¯03
×   RE Y     1.000E¯03                                         4.000E¯03                                         7.000E¯03
=  MAG Y     1.000E¯03                                         4.000E¯03                                         7.000E¯03
 ↓       F     |                                                 |                                                 |
    1.0000E09 |----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
               |  ×                              ○         =
               |    ×                               ○          =
               |     ×                                 ○         =
               |      ×                                   ○       =
    1.1000E09 |       ×                                      ○     =
               |        ×                                      ○   =
               |         ×                                       ○  =
               |          ×                                        ○ =
               |           ×                                          ○=
    1.2000E09 |           ×                                           =
               |            ×                                           =
               |            ×                                             =
               |             ×                                             =
               |              ×                                             ○=
    1.3000E09 |              ×                                              ○=
               |              ×                                                =
               |               ×                                               =○
               |               ×                                                =○
               |                ×                                               = ○
    1.4000E09 |                ×                                                = ○
               |                ×                                                  =○
               |                 ×                                                = ○
               |                 ×                                                   =○
               |                 ×                                                    =○
    1.5000E09 |                  ×                                                   ○=
               |                  ×                                                    ○ =
               |                  ×                                                     ○   =
               |                  ×                                                     ○      =
               |                   ×                                                     ○        =
    1.6000E09 |----|----|----|----×----|----|----|----|----|----|----|----|----|----|---○|----|----|-=--|----|----|
```

designed to ignore or point out in passing certain errors that might not affect the validity of all the results, but to stop if there is no reasonable way to proceed.

The simplest general procedure upon encountering an error message is the following: Type

```
      )SI
```

and you will get a list of functions that have their execution suspended. Get rid of these by typing →(without anything else)

```
      PLOT SS (MAG Y OF FILTER WT WG4),RE Y VS RE Y OF WG0
CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71  10:31

o  MAG Y     3.000E¯03                5.000E¯03                7.000E¯03
×   RE Y     3.000E¯03                5.000E¯03                7.000E¯03
 ↓    RE Y    |                        |                        |
   3.0000E¯03 ×----|--o-|----|----|----|----|----|----|----|----|
              |
              |
              |
              |  ×       o
   3.3000E¯03 |
              |
              |    ×       o
              |
              |
   3.6000E¯03 |      ×     o
              |
              |        ×    o
              |
              |
   3.9000E¯03 |          ×  o
              |
              |            × o
              |
              |             ×o
   4.2000E¯03 |
              |              ×o
              |                ×
              |
              |                 ×
   4.5000E¯03 |                  ×
              |
              |                   ×
              |                    ×
              |                     ×
   4.8000E¯03 |                      ×
              |                      o×
              |                       ×
              |                       o×
              |                       o ×
   5.1000E¯03 |                        o×
              |                         o×
              |                         o××
              |                            ×
              |                            ××o o
   5.4000E¯03 |                             ×    o
              |                             ××     o  o
              |
              |
              |
   5.7000E¯03 |
              |
              |
              |
              |
   6.0000E¯03 |----|----|----|----|----|----|----|----|----|----|

      ⍝ END OF EXAMPLE 9.
```

as many times as necessary, until that list of functions vanishes. Then correct whatever is wrong and try again. The paragraphs below are intended to help you figure out what is wrong. In general, one error may trigger another error, so that you may get more than one error message. If so, pay most attention to the first.

Most error messages produced by *MARTHA* contain the word *NOT* or *ATTEMPTS*; most of those produced by the APL system contain the word *ERROR* or *FULL*. Those produced in response to system commands are APL messages.

The message *ATTEMPT TO DIVIDE BY ZERO* is caused by an attempt to divide by complex zero or take its reciprocal. After this message some of the printed or plotted results may be correct, and others meaningless. There are two possible causes of this message. First, the network or one of its sections might be one for which the computer's internal representation of the network is pathological. To correct this condition you will have to define your network in a different way. Second, you may have requested a response that is actually infinite. An example is

```
      PRINT Z11 OF WS R 7
```

The message starting *NOT A SECTION* is usually caused by forgetting to use one of the element-definition functions.

Example:

```
      PRINT Z OF (C 1E¯6) P 50
```

This message can also be generated if you change the number of frequencies in the middle of a computation, or if you omit parentheses.

Example:

```
      PRINT Z OF C 1E¯6 P R 50
```

An error message beginning *NOT A NETWORK* is usually caused by forgetting the word *OF*, or by changing the number of frequencies during the middle of a calculation.

A message starting *NOT* and then giving a list of items expected, such as resistance or capacitance, is caused by using too many or too few parameters in defining elements. For example, *FET* requires an argument of length exactly three, and *L* requires either one or three. Similarly, the message *NOT A SCALE FACTOR* results from having more than one number in the left argument of the function *ZSCALE*.

A complete list of *MARTHA* error messages is given in Appendix B of this manual.

Among the APL error messages, most include the word *ERROR* or *FULL*.

The message *VALUE ERROR* means that the item in question has not been defined. This is often caused by misspelling names of functions or variables, for example

```
      PIRNT Z11 OF FILTER WC R 50
```

```
    )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED    20.52.47 06/30/71

      PRINT Z11,Y11 OF WS R 7
ATTEMPT TO DIVIDE BY ZERO

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   10:18
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY

      F         RE Z11      IM Z11      RE Y11      IM Y11
 ----------- ----------- ----------- ----------- -----------
  1.0000E00   1.0000E00   1.0000E00   1.4286E¯01  0.0000E00
  2.0000E00   1.0000E00   1.0000E00   1.4286E¯01  0.0000E00
       ⍝ NOTE THAT SOME OF THE RESULTS ABOVE ARE ALL RIGHT DESPITE THE ERROR MESSAGE.

      PRINT Z OF (C 1E¯6)P 50
NOT A SECTION. DID YOU FORGET TO DEFINE ELEMENTS?
VALUE ERROR
P[1]
       ⍝ TO FIND OUT HOW MUCH SPACE IS UNUSED IN YOUR WORKSPACE, BY BYTES, TYPE ⌶22
       ⌶22
13740
       ⍝ THIS MEANS 13740 BYTES. EACH REAL NUMBER TAKES 8 BYTES.
      ⌶22
13916
       ⍝ NOTE THAT BY TYPING → AFTER AN ERROR YOU CAN OFTEN GET RID OF TEMPORARY RESULTS AND MAKE MORE ROOM.

      PRINT Z OF C 1E¯6 P 50
NOT A SECTION. DID YOU FORGET TO DEFINE ELEMENTS?
RANK ERROR
P[1]
      ⌶22
13732
      →
      ⌶22
13916

      PRINT Z, Y OF (R 50)P C 1E¯6
NOT A NETWORK, DID YOU FORGET 'OF'?
VALUE ERROR
PA[2]
      ⌶22
13484
      )SI
PA[2] *
Y[1]
      ⍝ THE REPLY TO  )SI  IS NOT BLANK, SO IT WILL HELP TO TYPE →
      →
      )SI
      ⍝ NOW THE REPLY TO  )SI  IS BLANK, AND ALL TEMPORARY RESULTS ARE ERASED.

      PRINT Z OF WG 334.9
NOT AN FC, ZINF, LENGTH (OR FC, ZINF)
VALUE ERROR
      PRINT Z OF WG 334.9
                 ^
      →
      PRINT Z11 OF FET 3E¯12,.45
NOT A CGS, CGD, GM
VALUE ERROR
      PRINT Z11 OF FET 3E¯12,.45
                   ^
      →
      PRINT Z OF 2 5 ZSCALE R 2.33
NOT A SCALE FACTOR
VALUE ERROR
      PRINT Z OF 2 5 ZSCALE R 2.33
                     ^
      →

      PIRNT Z11 OF (R 25)WC WS C 1E¯6
VALUE ERROR
      PIRNT Z11 OF(R 25)WC WS C 1E¯6
      ^

      PRINT Z OF R 0
DOMAIN ERROR
R[2]
```

The message *DOMAIN ERROR* means that some APL function encounters an argument with which it cannot cope. An example is taking the logarithm of a negative number, or dividing by zero. In using *MARTHA* this error is often caused by setting a resistance to zero. Instead, use a small nonzero value such as 1*E*¯10.

The message *CHARACTER ERROR* indicates an illegal overstruck character, for example an underlined numeral.

The message *DEFN ERROR* is caused by various illegal actions dur-

```
      )SI
R[2]  ⋆
      →
      )SI
      I22
13708

      PRINT Z ~~FOR~~ OF R 35
CHARACTER ERROR
      PRINT Z
             ^
      →

      ZL←R 50
      PRINT ZIN OF (WS R 5)WC R 25
RANK ERROR
PA[6]
      →

      F←.5×⍳70
      PRINT Z11 OF (WS R 31.5)WC C 1E¯4
WS FULL
ES[2]
      I22
8376
      ⍝ TO CLEAR TEMPORARY STORAGE WE SHOULD TYPE → NOW. LET'S SUPPOSE WE FORGET TO DO SO.
      F←.5×⍳50
      PRINT Z11 OF (WS R 31.5)WC C 1E¯4
WS FULL
AS[2]
      )SI
AS[2] ⋆
WS[1]
ES[7] ⋆
WC[1]
      →
      )SI
ES[7] ⋆
WC[1]
      →
      )SI
      PRINT Z11 OF (WS R 31.5)WC C 1E¯4

CIRCUIT ANALYSIS BY MARTHA.   71∘A   7/13/71   10:56


      F         RE Z11      IM Z11
 ----------- ----------- -----------
  5.0000E¯01  3.1500E01  ¯3.1831E03
  1.0000E00   3.1500E01  ¯1.5915E03
  1.5000E00   3.1500E01  ¯1.0610E03
  2.0000E00   3.1500E01  ¯7.9577E02
  2.5000E00   3.1500E01  ¯6.3662E02

  3.0000E00   3.1500E01  ¯5.3052E02
  3.5000E00   3.1500E01  ¯4.5473E02
  4.0000E00   3.1500E01  ¯3.9789E02
  4.5000E00   3.1500E01  ¯3.5368E02
  5.0000E00   3.1500
PRINT[8]
      ⍝ TO STOP THIS LENGTHY PRINTOUT. PRESS THE ATTN BUTTON.
      ⍝ NOTE THAT THE ANALYSIS WITH 50 FREQUENCIES DID NOT WORK AT FIRST, BECAUSE SPACE WAS TAKEN UP BY
      ⍝  RESULTS FROM THE PREVIOUS ERROR. AFTER ALL TEMPORARY RESULTS WERE ERASED, THE ANALYSIS WITH 50
      ⍝  FREQUENCIES WORKED ALL RIGHT.
```

ing function definitions, such as giving a function name that already is in use.

The message *SYNTAX ERROR* indicates an invalid syntax, often caused by missing arguments, or by attempts to use names as variables when they are already in use as functions. A common example is *RE* which you might want to use for the emitter resistor, except that it is already in use as a modifier.

The messages *LENGTH ERROR* or *RANK ERROR* indicate that the size or shape of some variable is incorrect. This can result if you set *ZG*, *ZL*, *ZN*, *ZNIN*, or *ZNOUT* to *R* 50 instead of to 50.

The message *RESEND* is frustrating because it is not your fault. It is usually caused by a noisy telephone line or data set. After this message, type the previous line again.

The message *SYMBOL TABLE FULL* means that too many names (of functions or variables) have been used. You cannot continue without erasing some of the names, then saving the workspace, clearing it, and copying the saved workspace.

The message *WS FULL* means that too many frequencies are being considered simultaneously, so that there is too much intermediate storage. When defining your networks, be tidy and do not leave behind too many sections; elements require very little storage, but sections require more. It may be necessary for you to reduce the number of frequencies and try again. You should not have much trouble with up to 30 frequencies simultaneously. This problem also results if you fail to type → after previous errors, because until you do so the intermediate results calculated just before that error will be retained, using up your workspace.

MARTHA is designed for a workspace of 32k bytes; if you have larger workspaces, you should not have any trouble from *WS FULL* messages.

A message containing the word *ERROR* preceded by the name of a function in *MARTHA* indicates merely that trouble was encountered while that function was being executed. It does not suggest the type of trouble encountered, but the most common types are *WS FULL* or *LENGTH ERROR*. Check the following: *ZG*, *ZL*, *ZN*, *ZNIN*, and *ZNOUT* should be single numbers, not vectors. See if all the functions in *MARTHA* are present---some functions call others and if you erase one of them to give you more room, you might erase one that is needed. Beware of zero values for circuit parameters. The left argument of *SYMBOLS* must be a character string, not a numerical value. Do not define a network in terms of itself, since this is certain to fill up the workspace.

MARTHA works only in *ORIGIN* 1 APL. This is the normal condition and if you do not request anything else, you will have no problem with this.

If you continue to receive error messages, attempt to isolate the simplest possible example where the error occurs. See if it occurs in any of the examples given in this manual. If so, your copy of *MARTHA* may be faulty.

2.15 Summary

MARTHA is a set of co-operating APL functions that, together, comprise a general-purpose analyzer for electrical networks. The functions and variables in *MARTHA* are of 5 general types:

Elements	Wiring Functions	Response Functions	Modifiers	Format Functions
R	*P*	*Z*	*RE*	*PRINT*
L	*S*	*Y*	*IM*	*PLOT*
C	*WP*	*SC*	*MAG*	*PLOG*
TEM	*WS*	*Z11*, *Z12*, *Z21*, *Z22*	*DB*	*TITLE*
WG	*WC*	*Y11*, *Y12*, *Y21*, *Y22*	*RAD*	*SYMBOLS*
FET	*WTO*	*H11*, *H12*, *H21*, *H22*	*DEG*	*SS*
HYBRIDPI	*WTS*	*S11* ,*S12*, *S21*, *S22*		*WIDE*
WTHRU	*WT*	*ZIN*, *ZOUT*		*HGH*
WR	*WPP*	*YIN*, *YOUT*		*VS*
IT	*WPS*	*SIN*, *SOUT*		
OPAMP	*WSP*	*VG*		
	WSS	*AG*		
	WN	*G*		
	WROT	*PG*		
	ZSCALE	*TG*		

In addition there are the variables you set for your particular analysis: *F*, *ZG*, *ZL*, *ZN*, *ZNIN*, and *ZNOUT*. Also there are the useful functions *DEGREESAT* and *FORDIEL*, and the necessary function *OF*, and the variable *SAME* which is described in the next section. Finally there are several variables and functions that you do not use directly; these all have names consisting of two underlined letters.

There is no logical end to the types of elements that may be defined, or the wiring functions that may be useful, or the possible response functions, modifiers, or individual formats, or extra functions to work with *MARTHA*. *MARTHA* has associated with it an open-ended collection of special-purpose functions located in other workspaces in library no. 100, and you can get any of them whenever you need them. This collection is described in Chapter 4 of this manual.

A succinct summary of *MARTHA* is available on-line in the workspace 100 *HOWMARTHA*. This is useful to refresh your memory, and for quick reference. To see this summary, type

```
      )LOAD 100 HOWMARTHA
      DESCRIBE
```

Appendix B of this manual gives this on-line documentation as of the date of this manual. As the *MARTHA* library expands, the on-line documentation will be kept up to date.

One particularly useful function in the library is *WHATIS*, which is stored in the workspace 100 *MARTHAX*. This function tells how any particular variable would be interpreted by *MARTHA*. It is useful if, for example, you have defined a resistor and cannot remember its value. If the resistor is named R1, then merely type

```
WHATIS R1
```

and you will find its value.

3.1 More Than One Network

MARTHA can analyze more than one network at a time. For example, if you wish to compare the real and imaginary parts of *Z*21 of a network with the name *NET*1, with and without a 50-ohm resistor across the output, you can ask for simultaneous analysis of two networks, with and without the resistor. Enclose the output list, *OF*, and the network description of all but the last network in parentheses, and don't forget the comma between them:

```
PRINT (Z21 OF NET1), Z21 OF NET1 WC R 50
```

As another example, consider a filter of several identical sections named *PI* in cascade, as shown in Figure 2.16. You can compare graphically the insertion gain of a one-section, two-section, and three-section filter with a single request

```
PLOT(DB IG OF PI),(DB IG OF PI WC PI),DB IG OF PI WC PI WC PI
```

An even better graph would result by having the same scale used on the three dependent variables:

```
PLOT SS(DB IG OF PI),(DB IG OF PI WC PI),DB IG OF PI WC PI WC PI
```

3.2 Analyzing the Same Network Again

You can reduce the cost of running *MARTHA*, save some time at your terminal, and reduce the possibility of typing errors, in cases where you wish to analyze the same network again. Instead of referring to the network by its name or usual description, call it *SAME*.

Example:

```
PRINT Z11, Z21 OF (TEM 50,90 DEGREESAT .1E9) WT (R20) S C 2E¯6

PLOT Z11, Z21 OF SAME

PLOT MAG Z11, DEG Z11 OF SAME
```

This feature still works when you change the frequency slightly. The numerical values associated with the network will be interpolated (or extrapolated if necessary) to the new values of frequency. This procedure is not completely accurate, of course, but it may be sufficiently so. The old network is destroyed whenever wiring functions are executed, so you usually cannot use *SAME* as part of an expression containing wiring functions. However, you can achieve the same effect by preserving *SAME* under a new name of your choice. The result is a numerically defined two-port element which can be kept and used indefinitely.

```
      PRINT DB IG OF FILTER WC R 25

      OLDFILTER←SAME

      PRINT DB IG OF OLDFILTER WC WS L .015
```

3.3 Saving Your Work

The workspace file system of APL can be used to save your work from day to day, or indefinitely. You can build up a private collection of models, networks, calculated results, *FOF*'s, or functions to work with *MARTHA*. The most convenient way to do this is described in this section.

Groups. In an APL workspace you may define collections of variables and functions (and other groups) to be members of a group. The purpose of groups is to allow you to copy or erase all members of the group by simply naming the group; you do not need to name each object separately. For example, suppose you are working on two projects, A and B, using in project A the frequency vectors *F*1, *F*2, and *F*1*A*, the networks *FILTER*1, *NEWFILTER*, and *COUPLING*, and the elements *T*1, *T*2, and *T*13. You might define a group consisting of all these as follows:

```
      )GROUP PROJA F1 F2 F1A FILTER1 NEWFILTER COUPLING T1 T2 T13
```

The objects used in project B might similarly be put into a group:

```
      )GROUP PROJB C1 C2 C3 T1 T3A FOF1 FOF2 AMP1 AMP2 Q15
```

Note that *T*1 is a member of both groups.

You can define many groups, and the objects in a group need not exist at the time the group is defined.

In the workspace 100 *MARTHA*, all variables and functions are in a group named *MARTHA* except for the six variables that you as a user set (*F*, *ZG*, *ZL*, *ZN*, *ZNIN*, and *ZNOUT*). This allows you to erase *MARTHA* easily but to preserve your present values for those variables.

Copying. There are two ways to copy objects from a workspace. The)*COPY* command copies the entire workspace (or the specific group or object named). If you have an object in your active workspace with the same name as something coming in, it will be erased. The)*PCOPY* command is similar except that objects already in your workspace are protected, so they will not be erased. Instead, objects with the same name are not copied.

Saving. Full details on the file system are in the APL\360: User's Manual[9], and as you become familiar with it you will work out your own method of saving your work. The method suggested here has the advantage that it conserves space and requires you to save fewer workspaces. There is no point in saving any of the functions in *MARTHA*, since these are always available.

Suppose that you have a workspace entitled *MARTHAWORK* with groups *PROJ*1, *PROJ*2, *PROJ*3, and *MODELS* and that today you wish to work on project 2, using some of your models. If you debug them you will want to save the debugged versions. After you log in, do the following

```
      )CLEAR
      )COPY 100 MARTHA
      )COPY MARTHAWORK PROJ2
      )COPY MARTHAWORK MODELS
```

. . . (now do your work) . . .

```
      )ERASE MARTHA
      )PCOPY MARTHA WORK
      )WSID MARTHA WORK
      )SAVE
      )OFF
```

3.4 Write Your Own Wiring Functions

If you find yourself using a particular topology frequently, you may want to define a special-purpose wiring function of your own. For example, consider the half-lattice of Figure 3.1. The resulting two-port section is expressed in terms of the two one-port sections as

```
      (WS S1) WPP WR WC WS S2
```

where the ideal transformer acts as a phase inverter and is equivalent to *WR*. A wiring function that describes this topology would be dyadic, and might look like this:

```
      ∇Z←A HALFLATTICE B
[1]   Z←(WS A)WPP WR WC WS B
[2]   ∇
```

Figure 3.2 shows some additional configurations that might warrant their own wiring functions. The expression is given in each case. Circuit (a) is the Darlington transistor connection, and circuit (b) is similar. Circuits (c) and (d) are identical half-lattices. Circuits

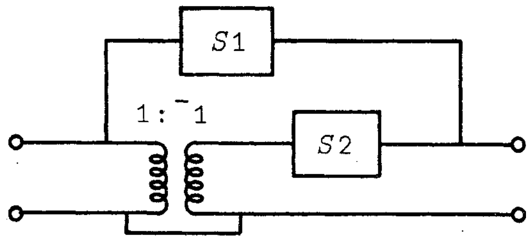
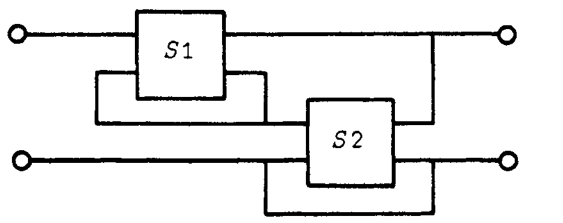
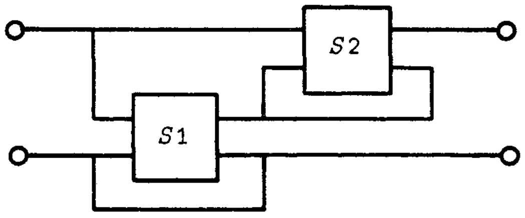


Figure 3.1. Half-lattice example, with ideal transformer.



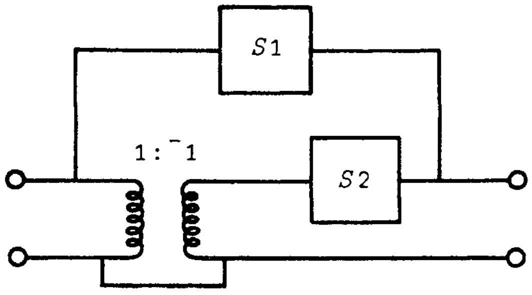
*WN WROT (WN WROT S*1) *WC WN WROT S*2
WROT WROT(*WROT S*2)*WC WROT S*1

(a)



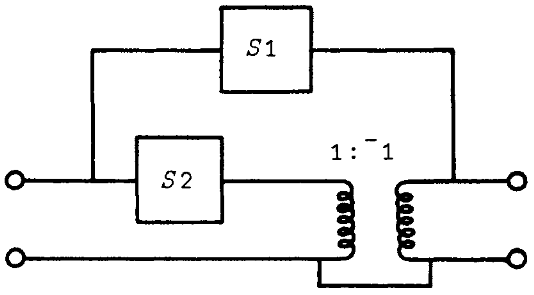
WROT WN(*WROT WN S*1)*WC WROT WN S*2
WROT(*WROT WROT S*2)*WC WROT WROT S*1

(b)



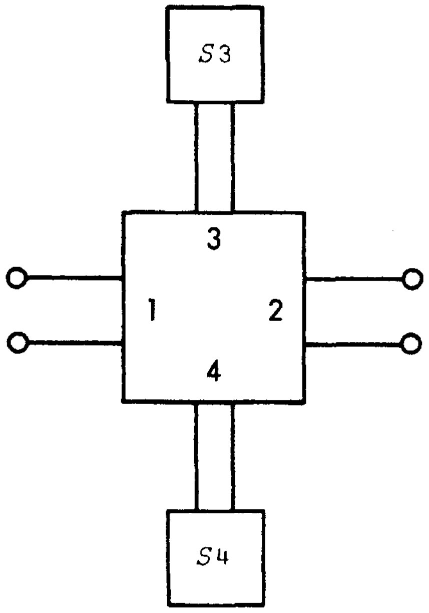
(*WS S*1)*WPP WR WC WS S*2

(c)



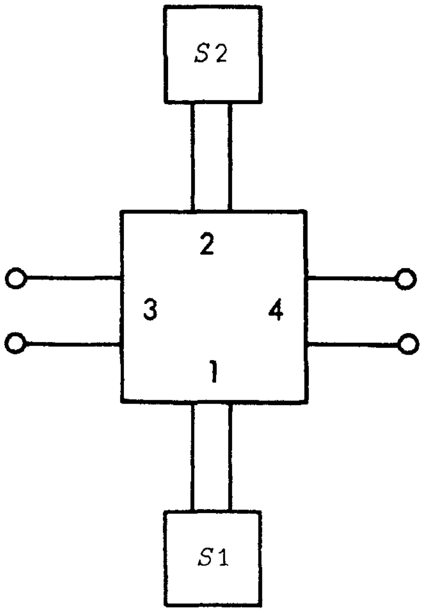
(*WS S*1)*WPP WR WC WS S*2

(d)



2 *ZSCALE*(*WS S*4) *WPP WR WC S*3

(e)



2 *ZSCALE*(*WS S*2)*WPP WR WC WS S*1

(f)

Figure 3.2. Expressions for some circuits. Circuits (e) and (f) are based on the hybrid of Figure 3.3.

(e) and (f) are based on the microwave magic tee[10] (Figure 3.3), which is a four-port matched directional coupler with coupling coefficient 0.707. If ports three and four are terminated in one-port sections *S*3 and *S*4, the result is Circuit (e). If *S*3 and *S*4 are adjustable lengths of waveguide, the result is an E-H tuner. Circuit (f) is similar except that ports one and two are terminated.

3.5 De-embedding

Embedding one network in another consists of using the second to "surround", at least partially, the first. De-embedding is the process of inferring properties of the embedded network, when the embedding network is known and the port properties of the combined network are known.

As a simple example, suppose a one-port network *X* is used to terminate a two-port network *B*. The result is a one-port network *A* which is *B WT X*. If *A* and *B* are known (perhaps by measurement) then *X* can be inferred using *A* to terminate a two-port network that "undoes" the network *B*. This network is ¯1 *ZSCALE WN B* and so *X* can be found from the expression *X*←(¯1 *ZSCALE WN B*)*WT A*

Table 3.1 shows several similar cases. In each *A* and *B* are assumed to be known, and *X* is unknown.

3.6 User-defined Elements

In *MARTHA* you can (if you know how to program in APL) define as many types of elements as you need---elements that cannot be otherwise modeled in *MARTHA*, or elements that are actually written in terms of standard elements in *MARTHA*.

Two types of functions must be written. One defines the elements.

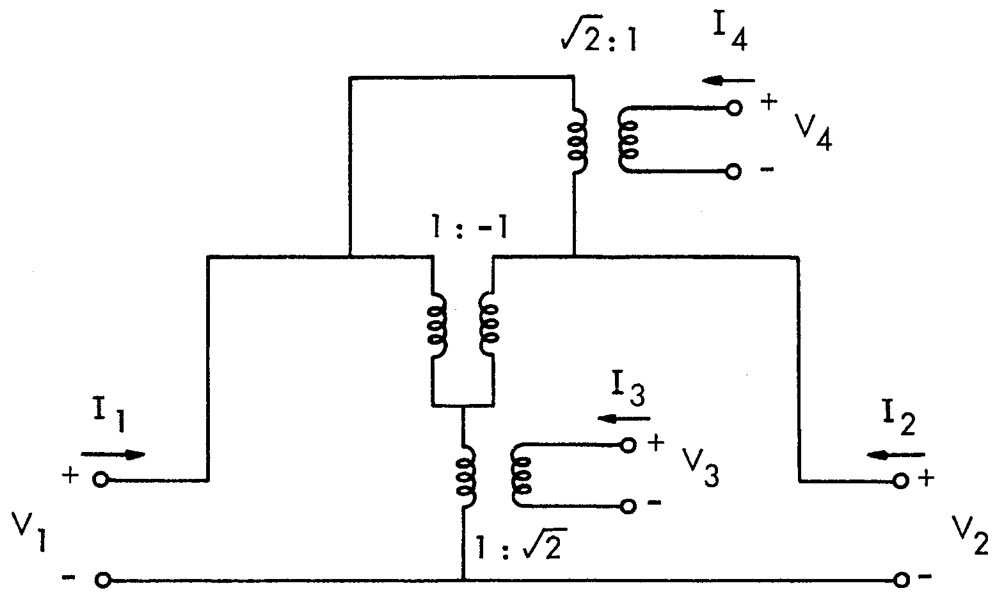


Figure 3.3. Hybrid coil, or magic tee.

All elements in *MARTHA* are independent of frequency, so the element-definition programs should not make use of *F*. An element must be a vector, and the first element of the vector must be 9. The other numbers in the vector carry the parameters of the element. If you define more than one type of element, you must be able to distinguish each type. The easiest method is to make the second number in the vector a code, such as 1, 2, 3, and so on.

For example, consider a lossy inductor. Let's model this as an inductor in series with a resistor, where the resistance value can be calculated from the quality factor Q, the inductance L, and the frequency at which the Q is measured, f_Q:

$$R = \frac{2\pi f_Q L}{Q} \tag{3.1}$$

Then a function that defines an element like this might expect as arguments the inductance, the Q and f_Q:

```
      ∇ Z←LOSSYL A
[1]     Z←9,1,A
[2]   ∇
```

Note that the vector begins with 9, and the second number is a code 1 to identify this as the first type of user-defined element. As

Table 3.1. Formulas for de-embedding networks.

If *A* and *B* are known and *X* is unknown, where	Solve for *X* by
A=X S B OR *A=B S X*	*X←A S* ¯1 *ZSCALE B*
A=X P B OR *A=B P X*	*X←A P* ¯1 *ZSCALE B*
A=WP X	*X←WTO A*
A=WS X	*X←WTS A*
A=X WC B	*X←A WC* ¯1 *ZSCALE WN B*
A=B WC X	*X←*(¯1 *ZSCALE WN B*)*WC A*
A=B WT X	*X←*(¯1 *ZSCALE WN B*)*WT A*
A=B WPP X OR *A=X WPP B*	*X←A WPP* ¯1 *ZSCALE B*
A=B WPS X OR *A=X WPS B*	*X←A WPS WR WC* ¯1 *ZSCALE B*
A=B WSP X OR *A=X WSP B*	*X←A WSP WR WC* ¯1 *ZSCALE B*
A=B WSS X OR *A=X WSS B*	*X←A WSS* ¯1 *ZSCALE B*
A=WN X	*X←WN A*
A=WROT X	*X←WROT WROT A*
A=WROT WROT X	*X←WROT A*
A=N ZSCALE X	*X←*(÷*N*) *ZSCALE A*
A=WAD X	*X←WAD A*
A=WCC X	*X←WCC A*
*A=RES WDUAL*1 *X*	*X←RES WDUAL*1 *A*
*A=RES WDUAL*2 *X*	*X←RES WDUAL*2 *A*

another example, consider the impedance of a short dipole antenna. This impedance is[11]

$$Z = \frac{4\pi}{3}\sqrt{\frac{\mu_0}{\varepsilon_0}}\,\frac{\ln\left(\frac{h}{a}\right) - 1}{2\ln\left(\frac{2h}{a}\right) - 3}\left(\frac{hf}{c}\right)^2 - j\,\frac{\ln\left(\frac{h}{a}\right) - 1}{2\pi^2 f\varepsilon_0 h} \tag{3.2}$$

where h is the half-length of the antenna, a is the radius, and c is the speed of light. The second term is the reactance of a capacitor with capacitance

$$\left(\frac{\pi\varepsilon_0 h}{\ln\left(\frac{h}{a}\right) - 1}\right) \tag{3.3}$$

and the first term is a frequency-dependent resistance. The parameters of the antenna are h and a, and if they are expected as the argument, then a function to define the short dipole element would be

```
      ∇Z←SHORTDIPOLE ARG
[1]   Z←9,2,ARG
[2]   ∇
```

Note the code number 2 to identify this as the second type of user-defined element.

The other function that must be written is one that evaluates the section. When *MARTHA* encounters a user-defined element, it calls upon a function with the name *NEWELEMENT*, which you as a user must supply. This function should accept as an argument the element (i.e., the vector defined by the functions like *LOSSYL* or *SHORTDIPOLE*) including the leading number, 9. Thus for a lossy inductor with inductance .015 H, quality factor 400 at 1 MHz, the argument would be 9 1 .015 400 1*E*6.

The first statement in the function *NEWELEMENT* should be a branch to the portion of the function that pertains to that element. (This statement can be omitted if there is only one type of user-defined element.) This is determined by the code number (in our example, 1 or 2). The description of the network should follow next. In the case of the lossy inductor, the function is simple because the new element can be modeled in terms of elements already in *MARTHA*:

```
      ∇Z←NEWELEMENT A
 [1]  →(1 2=A[2])/B1,B2
 [2]  B1:Z←(L A[3])S R ○2×A[5]×A[3]÷A[4]
 [3]  →0
```

In the case of the radiation impedance, more extensive programming is necessary because the real part is frequency-dependent. To do this programming you have to know how information about sections is stored by *MARTHA*. A section is stored as a matrix *N* × 8, where *N* is the number of frequencies. The eight columns are, in order, the real and imaginary parts of A, B, C, and D. Thus you need to calculate the ABCD matrix of your element. In our case it is easiest to consider the impedance as the series wiring of a capacitor with a one-port impedance R(f), for which the ABCD matrix is: A = D = 1, B = 0, C = 1/R(f). Thus our aim is to generate a matrix with the values of 1/R(f) in the fifth column, one in columns one and seven, and zero elsewhere. First the ones and zeros can be generated by the statement

```
      Z←(N,8)ρ 1  0  0  0  0  0  1  0
```

```
      )CLEAR
CLEAR WS
      )COPY 100 MARTHA
SAVED   20:52:47  06/30/71

      ∇Z←LOSSYL A
[1]   Z←9,1,A
[2]   ∇

      ∇Z←SHORTDIPOLE ARG
[1]   Z←9,2,ARG
[2]   ∇

      ∇Z←NEWELEMENT A;N
[1]   →(1 2=A[2])/B1,B2
[2]   B1:Z←(L A[3])S R ○2×A[5]×A[3]÷A[4]
[3]   →0
[4]   B2:N←ρF
[5]   Z←(N,8)ρ1 0 0 0 0 0 1 0
[6]   Z[;5]←÷(○4÷3)×377×((¯1+⍟A[3]÷A[4])÷¯3+2×⍟2×A[3]÷A[4])×(A[3]×,F÷3E8)★2
[7]   Z←Z S C ○8.854E¯12×A[3]÷¯1+⍟A[3]÷A[4]
[8]   ∇

      C1←C .025E¯6
      L1←L 1E¯6
      LOSSYL1←LOSSYL 1E¯6,100,1E6
      F←1E6×.9+.01×ι20

      PLOT 20 HIGH SS (MAG Z OF C1 P L1), MAG Z OF C1 P LOSSYL1


CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/15/71   10:15
MARTHA COPYRIGHT (C) 1971 MASSACHUSETTS INSTITUTE OF TECHNOLOGY


○  MAG Z     0.000E00                5.000E02                1.000E03
×  MAG Z     0.000E00                5.000E02                1.000E03
 ↓      F     |                        |                        |
    9.0000E05 |----|----|----|----|----|----|----|----|----|----|
              | ×
              | ×
              | ×
              | ×
    9.5000E05 |  ×
              |  ×
              |   ×
              |     ×
              |        ×○
    1.0000E06 |                  ×    ○
              |                         ×                    ○
              |          ×○
              |      ×
              |    ×
    1.0500E06 |   ×
              |  ×
              |  ×
              | ×
              | ×
    1.1000E06 |-×--|----|----|----|----|----|----|----|----|----|
      F←1E6×.5,1,2,4,8,16,32
      PRINT Z, MAG Z, DEG Z OF SHORTDIPOLE .32,.0049


CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/15/71   10:17


      F         RE Z        IM Z       MAG Z       DEG Z
 ----------- ----------- ----------- ----------- -----------
  5.0000E05   2.1173E¯04 ¯1.1369E05   1.1369E05  ¯9.0000E01
  1.0000E06   8.4692E¯04 ¯5.6844E04   5.6844E04  ¯9.0000E01
  2.0000E06   3.3877E¯03 ¯2.8422E04   2.8422E04  ¯9.0000E01
  4.0000E06   1.3551E¯02 ¯1.4211E04   1.4211E04  ¯9.0000E01
  8.0000E06   5.4203E¯02 ¯7.1055E03   7.1055E03  ¯9.0000E01

  1.6000E07   2.1681E¯01 ¯3.5527E03   3.5527E03  ¯8.9997E01
  3.2000E07   8.6725E¯01 ¯1.7764E03   1.7764E03  ¯8.9972E01

      )COPY 100 MARTHAX PSWEEP
SAVED   12:38:36  07/01/71

      ⍝ TO USE THE PARAMETER SWEEPER, WE EDIT LINE [3] OF THE FUNCTION SWEEPPAR.
      ∇SWEEPPAR[3]
[3]  STORE Z OF SHORTDIPOLE PAR,.0049
[4]  ∇

      F←1E6
      F←1E6
      ⍝ THE HALF-HEIGHT OF THE DIPOLE IS THE PARAMETER BEING SWEPT, FROM 2 CM. TO 50 CM.
      PLOT SWEEPPAR .02×⍳25

CIRCUIT ANALYSIS BY MARTHA.   73∘A   7/15/71   10:20


○   RE Z     0.000E00                 1.500E¯03                3.000E¯03
×   IM Z    ¯2.000E05                ¯1.000E05                 0.000E00
 ↓      PAR   |                        |                        |
    0.0000E00 |----|----|----|----|----|----|----|----|----|----|
              |
              ○                    ×
              |
              ○          ×
    5.000E¯02 |
              ○             ×
              |
              |○                ×
              |
    1.000E¯01 |○                   ×
              |
              | ○                     ×
              |
              |  ○                      ×
              |
              |  ○                        ×
              |
              |   ○                        ×
              |
    2.000E¯01 |    ○                         ×
              |
              |      ○                        ×
              |
              |       ○                        ×
    2.500E¯01 |
              |        ○                        ×
              |
              |          ○                      ×
              |
    3.000E¯01 |           ○                      ×
              |
              |             ○                     ×
              |
              |               ○                   ×
    3.500E¯01 |
              |                 ○                  ×
              |
              |                   ○                ×
              |
    4.000E¯01 |                     ○               ×
              |
              |                       ○             ×
              |
              |                          ○           ×
    4.500E¯01 |
              |                            ○         ×
              |
              |                               ○      ×
              |
    5.000E¯01 |----|----|----|----|----|----|----○----×----|----|
```

Then the fifth column can be set by the statement

```
Z[;5]←÷(○4÷3)×377×((¯1+⍟A[3]÷A[4])÷¯3+2×⍟2×A[3]÷A[4])×(A[3]×,F÷3E8)⋆2
```

Finally, this section should be wired in series with a capacitor:

```
      Z←Z S C ○8.854E¯12×A[3]÷¯1+⍟A[3]÷A[4]
```

The entire function is shown in the accompanying example.
Without very much knowledge of programming, you should be able to devise user-defined elements like the lossy inductor, that are modeled by elements already in *MARTHA*, wired together. The radiation impedance is more difficult because it requires, first, evaluation of the ABCD matrix for the device, and second, actual programming to yield the matrix.

3.7 The *MARTHA* Library

The *MARTHA* library consists of functions and variables that are compatible with *MARTHA*, but are not of enough general interest to be included in the workspace 100 *MARTHA*. There are two major kinds of functions in the library. First are additional elements, wiring functions, response functions, and modifiers that are similar in use to those in *MARTHA*. Second are extra functions that work with *MARTHA* and do services not ordinarily done by *MARTHA*. Examples of these are calculation of electrical parameters of coaxial or microstrip transmission lines; storing results calculated by *MARTHA*; and accepting, editing, and interpreting numerically-defined functions of frequency (*FOF*'s).
The library is open-ended in the sense that additional functions can be added at any time. The organization and contents of the library, as of the date of this manual, are summarized in Chapter 4. Some of the services performed by functions in the library are discussed in the next few paragraphs.

3.8 Identification of Objects

Elements, sections and *FOF*'s are stored internally by *MARTHA* in ways that are not always obvious. The function *WHATIS* in the *MARTHA* library, in the workspace 100 *MARTHAX*, will tell how any object would be interpreted by *MARTHA*. To use it, copy it into your active workspace and type *WHATIS* followed by the name of the object you are unsure about.
This function is rather lengthy, and you will probably not want to keep it around after you have finished with it. You can erase it by typing

```
      )ERASE WHATIS
```

The next time you need it you can always copy it from the *MARTHA* library.

3.9 Functions of Frequency

A numerically defined function of frequency (*FOF*) is, basically, a table of values of "defining frequencies" (not necessarily related to *F*) and corresponding values of the function. *A FOF* may have one, two, three, or more columns (besides frequency). You may want to use a *FOF* to hold information about design objectives, and then let *MARTHA* compare proposed designs with your objectives. Or you might use a *FOF* to store measured data about a network you are attempting to model. The numerical values can be typed in or can be the results of calculations done by *MARTHA*. There are functions in the *MARTHA* library for creating, editing, combining, displaying, interpreting, and using *FOF*'s.

Creating a *FOF*. Use the function *MAKEFOF* to create a *FOF*. It requires an argument, and yields as an output the *FOF*.

If the argument of the function *MAKEFOF* is already a *FOF*, it will be assumed you want to edit the *FOF*, as described below.

If the argument is a single number, it is interpreted as the desired number of columns. You will be asked for frequencies to add or change. Since no frequencies have been defined yet, you cannot change the entries for any, but you presumably want to add some new frequencies. List them or give an expression for them. If you have previously defined a variable (other than *F*) with the frequencies in it, just type that variable name. Next you will be asked for the values of the first column. You can type as many values on one line as you want; these will be used in order, and if more are needed, you will be asked for more. You can enter them one at a time if you wish, or several at a time. When the first column is complete, you will be asked for values for column two. Continue in this way until all columns are defined.

Example:

```
      FOF1 ← MAKEFOF 2
2 COLUMNS FOR 0 FREQUENCIES,
FREQUENCIES TO CHANGE OR ADD:
⎕:
      5,3,8

COLUMN 1 FOR F=5
⎕:
      1.5
COLUMN 1 FOR F=3
⎕:
      2.5, 3.5

COLUMN 2 FOR F=5
⎕:
      1.1, 2.1, 3.1
```

When you are asked for the numerical values, you have many options. You can enter specific numbers, as many at a time as you wish. Or you can enter any expression, which is immediately evaluated and used. The frequencies for which entries have not yet been defined in that column are temporarily stored in the vector *F*; thus you can write any specific function of *F* to be used in the remainder of that column. Or you can use an analysis by *MARTHA* to generate numbers for that column by answering something like

```
      RE Z11 OF FILTER1 WC R 75.2
```

The response *RE Z*11 will be evaluated for the frequencies that remain for the network you describe. For the next column you again have the same freedom, and you might reply something like

```
      IM Z11 OF SAME
```

If the argument of the function *MAKEFOF* is neither *a FOF* nor a single number, you will be asked how many columns you want. The argument will then be fashioned into *a FOF* as well as possible, and you will be able to edit it. In particular, if the argument is the output from a regular *MARTHA* analysis, the *FOF* will come out correctly if you answer the right number of columns, allowing two for each complex response.

<u>Example</u>:

```
      FOF2←MAKEFOF Z11, Z12, Z21, Z22 OF MYMODEL
HOW MANY COLUMNS (BESIDES FREQ.)?
⎕:
      8
8 COLUMNS FOR 6 FREQUENCIES.
```

Alternatively, the function *STORE* described in the next section can create *a FOF* from a regular *MARTHA* analysis.

<u>Editing *a FOF*</u>. To edit *a FOF*, again use the function *MAKEFOF*. In this case the argument should be *a FOF* that has already been defined, but in which you have detected some errors in the numerical values.

You will be asked first for a list of frequencies to delete. You may answer *NONE* if that is your intention, or *ALL*, or you may type the numerical values of the ones you want deleted. Next you will be asked for frequencies to change or add. Again you can answer *NONE* or *ALL*, or you can type any expression that, when evaluated, gives the numerical values. Values that coincide with existing frequencies are assumed to be requests for changing, and the rest are new frequencies.

You will now be asked for the entries for column 1, for the frequencies you are changing or adding. You have all the options described in the paragraphs above, namely, typing numerical values, as many at a time as you wish, or typing expressions (possibly involving *F*) or even the results of an analysis by *MARTHA*. In addition, if you do not wish to change any of the remaining numerical values in this column, you can type *OLD* and the old values will be used. Or you can use *OLD* in an expression to create the new values, for example by typing -*OLD* or 2×*OLD* or (*F*★3)×*OLD*.

It is not possible in *MAKEFOF* to edit selectively by column. You must edit all columns for the frequencies that are in error. However, this does not take long because you can say *OLD* for each of the columns that is correct.

To change the value of a frequency (as opposed to the value of one of the entries for one of the columns), delete the offending frequency and add the correct one.

Selecting Columns of a *FOF*. You can create a new *FOF* from an existing *FOF* with any selection of columns, in any order, by using the function *COLUMNSOF*. The same set of defining frequencies is used. The function *COLUMNSOF* has a left argument consisting of a vector of column numbers. The new *FOF* will ordinarily have as many columns as the length of the vector. The right argument is an existing *FOF*, and the columns of that *FOF* will be put into the indicated columns of the new *FOF*.

Examples:

```
FOF5←2 4 3 1 COLUMNSOF FOF1
FOF6←2 2 1 2 COLUMNSOF FOF1
FOF7←1 2 3 0 4 4 COLUMNSOF FOF1
```

Assume FOF1 has at least 4 columns. *FOF*5 is a 4-column *FOF* with the first 4 columns of FOF1 in a new order. *FOF*6 is a 4-column *FOF* with three of the columns identical, and equal to column 2 of FOF1. *FOF*7 has 6 columns, and the fourth column is all zeros. The fifth and sixth columns are identical.

Any numbers in the left argument that are not positive integers less than or equal to the number of columns in the original *FOF* will be ignored, except that, if 0 appears, it will be interpreted as a request for a column of zeros. Thus it is possible that the new *FOF* can have fewer columns than numbers in the left argument.

This function can also generate a *FOF* with no columns, which is, basically, a list of the defining frequencies. For example, ,(⍳0) *COLUMNSOF FOF*1 is a vector consisting of the frequencies at which *FOF*1 is defined.

Combining *FOF*'s. Two *FOF*'s with the same set of defining frequencies, or columns selected from them, can be combined to form a new *FOF* with the same defining frequencies. For example, suppose *FOF*1 and *FOF*2 have the same defining frequencies, arranged in the same order. This condition can be checked by typing

```
(,(⍳0)COLUMNSOF FOF1)=,(⍳0)COLUMNSOF FOF2
```

To combine these two *FOF*'s, proceed as illustrated:

```
      FOF3←MAKEFOF 5
5 COLUMNS FOR 0 FREQUENCIES.
FREQUENCIES TO CHANGE OR ADD:
⎕:
       ,(⍳0)COLUMNSOF FOF1

COLUMN 1 FOR F=1.2E6
⎕:
       1.COLUMNSOF FOF1

COLUMN 2 FOR F=1.2E6
⎕:
       2 COLUMNSOF FOF1

COLUMN 3 FOR F=1.2E6
⎕:
       3 COLUMNSOF FOF1

COLUMN 4 FOR F=1.2E6
⎕:
       1 COLUMNSOF FOF2

COLUMN 5 FOR F=1.2E6
⎕:
       2 COLUMNSOF FOF2
```

Two *FOF*'s with the same number of columns, but presumably different defining frequencies, can be combined using the function *STORE*, to form a *FOF* with the same number of columns but more defining frequencies. *STORE* stores its argument in a variable named *STORED*. If *STORED* already contains a *FOF* with the same number of columns, the new results are appended. Example: to combine two *FOF*'s, say *FOF*1 and *FOF*2, and call the result *FOF*3:

```
      STORED←⍳0
      STORE FOF1
      STORE FOF2
      FOF3←STORED
```

Displaying *FOF*'s. *FOF*'s can be displayed in all the formats that *MARTHA* has available except *VS*. In particular, the format functions *PRINT*, *PLOT*, and *PLOG* can be used, along with *SS*, *SYMBOLS*, *WIDE*, and *HIGH* if desired.

Examples:

```
      PRINT FOF1

      PLOT FOF1

      PLOT 20 WIDE 30 HIGH SS FOF2
```

```
PLOG '12345' SYMBOLS SS 1 2 3 4 5 COLUMNSOF FOF3
```

<u>Including *FOF*'s in the Output List</u>. *A FOF* (or selected columns if the function *COLUMNSOF* is used) can be placed directly in the output list when *MARTHA* is analyzing a network. This is useful if the *FOF* contains the desired response, which then can be compared directly with the calculated response. The response function *OUTFOF* is used for this purpose. It is not used in the same way as the other response functions. At the appropriate place in the output list, type *OUTFOF* preceded by the name of your *FOF*.

<u>Examples</u>:

```
PLOT RE S11, MAG S11, FOF3 OUTFOF, DEG S11 OF MODEL1

PRINT MEAS OUTFOF, VSWR OF FLT

PLOT S11, (2 COLUMNSOF RF)OUTFOF VS(1 COLUMNSOF RF)OUTFOF OF FLT
```

Generally the frequencies of analysis *F* do not coincide with the defining frequencies of the *FOF*. The values of each column of the *FOF* are estimated, at each of the frequencies of *F*, from a linear interpolation between the nearest defining frequency on each side. For those frequencies in *F* outside the range covered by the defining frequencies, a linear extrapolation based on the two nearest defining frequencies is made. If there is only one defining frequency, interpolation is not possible and the *FOF* will be assumed to be independent of *F*.

Each column of the *FOF* will be printed. If there are exactly two columns, it is assumed that they represent the real and imaginary parts of a complex function, and modifiers can be used as with other complex responses.

<u>Example</u>:

```
PRINT MAG (2 4 COLUMNSOF FOF1) OUTFOF, MAG Z21 OF NET1
```

<u>Interpreting *FOF*'s as Networks</u>. You can interpret your *FOF*'s as networks by using any one of five functions (*ZFOF*, *YFOF*, *HFOF*, *SFOF*, and *ABCDFOF*) to convert the *FOF* into an element. The element can then be wired together with other elements or sections using the wiring functions in *MARTHA*. This is useful, for example, if the *FOF* contains experimental measurements.

One-port elements can be created by the functions *ZFOF*, *YFOF*, and *SFOF*. If the argument for *ZFOF* is a two-column *FOF*, the columns are interpreted as the real and imaginary parts of the impedance of a one-port element. If the argument is a one-column *FOF*, that column is interpreted as the imaginary part of the impedance of a lossless one-port element; the real part is assumed to be zero. In a similar way, if the argument for the function *YFOF* is a two-column *FOF*, it is assumed that the columns are the real and imaginary parts of the admittance of a one-port element. If the argument is a one-column *FOF*, it is assumed that that column is the imaginary part of the admittance; the real part is taken to be zero. The function *SFOF* expects a two-column *FOF*, where the two columns are the real and imaginary parts of the reflection coefficient, normalized by *ZN*.

Two-port elements can be created by the functions *ZFOF*, *YFOF*, *SFOF*, *HFOF*, and *ABCDFOF*. In each case, the argument can be a *FOF* with three, four, six, or eight columns (except that *SFOF* cannot interpret a three-column *FOF*). The idea is that in general eight columns are necessary to describe a two-port element. However, for reciprocal elements two of these are redundant. If the element is reciprocal and also symmetric, then only four columns are necessary; if the element is reciprocal and also lossless, then only three columns are necessary. The interpretation of the columns in each case is shown in Tables 3.2, 3.3, 3.4, 3.5, and 3.6. Note that *ZFOF*, *YFOF*, and *SFOF* create both one-port and two-port elements, but *HFOF* and *ABCDFOF* create only two-port elements.

It is a property of all elements in *MARTHA* that they need not be redefined when the frequency is changed. This is true of the elements created by the functions *ZFOF*, *YFOF*, *SFOF*, *HFOF*, and *ABCDFOF*.

In general the defining frequencies of the *FOF*'s will not be the same as those ultimately used in the analysis by *MARTHA*. If the two sets of frequencies are reasonably close and intertwined, there will be no problems. In the analysis *MARTHA* will automatically interpolate between the defining frequencies, if necessary, to find numerical values corresponding to the values of *F*. If any members of *F* lie out-

Table 3.2. Interpretation of columns by the function *ZFOF*

	NUMBER OF COLUMNS					
COLUMN	1	2	3	4	6	8
1	*IM Z*	*RE Z*	*IM Z11*	*RE Z11*	*RE Z11*	*RE Z11*
2		*IM Z*	*IM Z12*	*IM Z11*	*IM Z11*	*IM Z11*
3			*IM Z22*	*RE Z12*	*RE 212*	*RE Z12*
4				*IM Z12*	*IM Z12*	*IM Z12*
5					*RE Z22*	*RE Z21*
6					*IM Z22*	*IM Z21*
7						*RE Z22*
8						*IM Z22*
ASSUMED	0=*RE Z*		0=*RE Z11* 0=*RE Z12* 0=*RE Z22* *Z21*=*Z12*	*Z21*=*Z12* *Z22*=*Z11*	*Z21*=*Z12*	
PROPERTIES OF RESULT	1-*PORT LOSSLESS*	1-*PORT*	2-*PORT RECIPROCAL LOSSLESS*	2-*PORT RECIPROCAL SYMMETRIC*	2-*PORT RECIPROCAL*	2-*PORT*

Table 3.3. Interpretation of columns by the function *YFOF*

COLUMN	*NUMBER OF COLUMNS* 1	2	3	4	6	8
1	*IM Y*	*RE Y*	*IM Y*11	*RE Y*11	*RE Y*11	*RE Y*11
2		*IM Y*	*IM Y*12	*IM Y*11	*IM Y*11	*IM Y*11
3			*IM Y*22	*RE Y*12	*RE* 212	*RE Y*12
4				*IM Y*12	*IM Y*12	*IM Y*12
5					*RE Y*22	*RE Y*21
6					*IM Y*22	*IM Y*21
7						*RE Y*22
8						*IM Y*22
ASSUMED	0=*RE Y*		0=*RE Y*11 0=*RE Y*12 0=*RE Y*22 *Y*21=*Y*12	*Y*21=*Y*12 *Y*22=*Y*11	*Y*21=*Y*12	
PROPERTIES OF RESULT	1-*PORT* *LOSSLESS*	1-*PORT*	2-*PORT* *RECIPROCAL* *LOSSLESS*	2-*PORT* *RECIPROCAL* *SYMMETRIC*	2-*PORT* *RECIPROCAL*	2-*PORT*

Table 3.4. Interpretation of columns by the function *HFOF*

COLUMN	*NUMBER OF COLUMNS* 3	4	6	8
1	*IM H*11	*RE H*11	*RE H*11	*RE H*11
2	*RE H*12	*IM H*11	*IM H*11	*IM H*11
3	*IM H*22	*RE H*12	*RE* 212	*RE H*12
4		*IM H*12	*IM H*12	*IM H*12
5			*RE H*22	*RE H*21
6			*IM H*22	*IM H*21
7				*RE H*22
8				*IM H*22
ASSUMED	0=*RE H*11 0=*IM H*12 0=*RE H*22 *H*21=-*H*12	*H*21=-*H*12 *H*22=(1- *H*12★2)÷ *H*11	*H*21=-*H*12	
PROPERTIES OF RESULT	2-*PORT* *RECIPROCAL* *LOSSLESS*	2-*PORT* *RECIPROCAL* *SYMMETRIC*	2-*PORT* *RECIPROCAL*	2-*PORT*

Table 3.5. Interpretation of columns by the function *SFOF*

	NUMBER OF COLUMNS			
COLUMN	2	4	6	8
1	*RE SC*	*RE S*11	*RE S*11	*RE S*11
2	*IM SC*	*IM S*11	*IM S*11	*IM S*11
3		*RE S*12	*RE* 212	*RE S*12
4		*IM S*12	*IM S*12	*IM S*12
5			*RE S*22	*RE S*21
6			*IM S*22	*IM S*21
7				*RE S*22
8				*IM S*22
ASSUMED		*SEE BELOW*	*S*21=*S*12	
PROPERTIES OF RESULT	1-*PORT*	2-*PORT RECIPROCAL SYMMETRIC*	2-*PORT RECIPROCAL*	2-*PORT*

ASSUMED FOR 4-*COLUMN FOF*: *S*21=*S*12;
*S*22=(*S*11+(1+*S*12★2)×(*ZNIN*-*ZNOUT*)÷*ZNIN*+*ZNOUT*)÷
1+*S*11×(*ZNIN*-*ZNOUT*)÷*ZNIN*+*ZNOUT*

Table 3.6. Interpretation of columns by the function *ABCDFOF*

	NUMBER OF COLUMNS			
COLUMN	3	4	6	8
1	*RE* $\underline{A}BCD$	*RE* $\underline{A}BCD$	*RE* $\underline{A}BCD$	*RE* $\underline{A}BCD$
2	*IM* $A\underline{B}CD$	*IM* $\underline{A}BCD$	*IM* $\underline{A}BCD$	*IM* $\underline{A}BCD$
3	*IM* $AB\underline{C}D$	*RE* $A\underline{B}CD$	*RE* $A\underline{B}CD$	*RE* $A\underline{B}CD$
4		*IM* $A\underline{B}CD$	*IM* $A\underline{B}CD$	*IM* $A\underline{B}CD$
5			*RE* $AB\underline{C}D$	*RE* $AB\underline{C}D$
6			*IM* $AB\underline{C}D$	*IM* $AB\underline{C}D$
7				*RE* $ABC\underline{D}$
8				*IM* $ABC\underline{D}$
ASSUMED	0=*IM* $\underline{A}BCD$ 0=*RE* $A\underline{B}CD$ 0=*RE* $AB\underline{C}D$ $ABC\underline{D}$=(1+ $A\underline{B}CD$× $AB\underline{C}D$)÷ $\underline{A}BCD$	$AB\underline{C}D$=(¯1+ $\underline{A}BCD$★2)÷ $A\underline{B}CD$ $ABC\underline{D}$=$\underline{A}BCD$	$ABC\underline{D}$=(1+ $A\underline{B}CD$× $AB\underline{C}D$)÷ $\underline{A}BCD$	
PROPERTIES OF RESULT	2-*PORT RECIPROCAL LOSSLESS*	2-*PORT RECIPROCAL SYMMETRIC*	2-*PORT RECIPROCAL*	2-*PORT*

side the range of the defining frequencies, a linear extrapolation will be made. If the *FOF* has only one defining frequency, interpolation and extrapolation are impossible and the element will be assumed to be independent of frequency.

3.10 Storing Results

Results calculated by *MARTHA* can be printed or plotted by the functions *PRINT*, *PLOT*, and *PLOG*. Results can also be stored for later printing, plotting, editing, or other interpretation. Results can also be combined with previously stored results, and in this way saved for printing or plotting all together.

Results are stored by the function *STORE*, which works in the following way. The stored results are put into a variable entitled *STORED*, in the form of a numerically defined function of frequency (*FOF*). The *FOF* has as many columns as dependent variables in the output list (two for each complex response). If *STORED* is already a *FOF* with the same number of columns, then the new results are appended to the old results. Otherwise, any old results are lost. Before you store any results, you must initialize the variable *STORED*, by typing

```
      STORED←⍳0
```

After that, you type something of the form

```
      STORE <output list> OF <network description>
```

Thus you can think of *STORE* as a different kind of format function, analogous to *PRINT*, *PLOT*, or *PLOG*.

The argument for *STORE* can be either the results of a *MARTHA* analysis or a *FOF*. Thus *STORE* is useful for combining *FOF*'s with the same number of columns or combining *FOF*'s with the results of a *MARTHA* analysis.

3.11 Analysis with Many Frequencies

There are two ways of analyzing networks with more frequencies than *MARTHA* can ordinarily handle at once. One is useful for printed output, and the other for both printed and plotted output.

If there is just one frequency, i.e.,if 1=×/⍴F, then the function *PRINT* does not print anything, but instead returns a line of text with the answers. In particular, the heading is not printed. The function below takes advantage of this feature by first analyzing the network with a reasonable number of frequencies, then repeatedly reanalyzing it, one frequency at a time. The result is printed output with the heading, and a large number of frequencies.

```
      ∇MANYFREQS
[1]   F←1E6×1+.02×0,⍳30
[2]   PRINT S12, MAG S12 OF NET3
[3]   F←.02E6+¯1↑F
[4]   →2×⍳2E6≥F
[5]   ∇
```

The second technique uses the function *STORE* to store the results of an analysis with a reasonable number of frequencies. Then *F* is changed and the results again stored. At this point the variable *STORED* is a *FOF* containing all the results, and can be printed or plotted. For example, if *F*1 and *F*2 are sets of frequencies, the technique might look like this:

```
      STORED←⍳0
      F←F1
      STORE Z11,Z12 OF AMPLIFIER
      F←F2
      STORE Z11,Z12 OF AMPLIFIER
      PLOT STORED
```

The function *ATATIME*, in the workspace 100 *MARTHAX*, does this automatically. It is a dyadic function; its left argument is a single number specifying the number of frequency points that will be used for each sweep. The right argument is a vector of frequencies. The result is a *FOF* with the desired output.

Example:

```
      PLOT 25 ATATIME 50×⍳65
```

The 65 frequencies will be arranged into sets of up to 25, and the network analyzed three different times with the three sets of frequencies. Information about the output list and the network description is entered by editing line [3] of the function *ATATIME*. For example, suppose your network is called *DIODE* and you want the impedance and admittance of it. Then you edit the function as follows:

```
      ∇ATATIME[3]
[3]   STORE Z, Y OF DIODE
[4]   ∇
```

Note that the first word in line [3] is *STORE*, rather than *PRINT* or *PLOT*. This new line [3] replaces the old line [3], which is a dummy.

ATATIME is generally used along with *STORE*; you can copy both at once by simply typing)*COPY* 100 *MARTHAX FSWEEP*

3.12 Sweeps with Respect to Circuit Parameters

It is often useful to keep the frequency fixed at one value and analyze the network repeatedly for various values of some scalar circuit parameter. The function *SWEEPPAR* sets up such a sweep. This is a monadic function whose argument is a vector of values for the parameter being swept. To use this function:

First, set the frequency to a single value. Next, edit line [3] of the function *SWEEPPAR*, to be of the form

```
      STORE <output list> OF <network description>
```

The network should be a function of the parameter being swept, which is denoted by *PAR*. Several of the element values could, in fact, depend upon *PAR*, and *PAR* may appear in any arithmetic expression to lead to element values. In this way different elements may be made to track together, for example to simulate the effect of temperature.

Now you are ready to use *SWEEPPAR*. The result returned by the function *SWEEPPAR* has the shape of a *FOF*, but (unless you specify differently in line [3] by using the format function *VS*) the independent variable will be the parameter being swept, rather than frequency, which is constant. The result can be printed or plotted. For example,

```
      PLOG 30 HIGH SWEEPPAR 1000×1,2⋆⍳10
```

You can, if you wish, store the results of the parameter sweep:

```
      STORED←⍳0
      STORE SWEEPPAR ⍳20
```

Since *SWEEPPAR* is normally used in conjunction with *STORE*, a group has been defined with these two in it. You can copy them both at the same time by typing

```
     )COPY 100 MARTHAX PSWEEP
```

The *MARTHA* Library

The *MARTHA* library is a collection of functions that are compatible with *MARTHA*, but are not in the workspace 100 *MARTHA*. They are stored in the following workspaces:

```
100 MARTHAE
100 MARTHAW
100 MARTHAR
100 MARTHAM
100 MARTHAX
```

The final letters in the names stand for "elements", "wiring", "responses", "modifiers", and "extra". You can get any of the functions at any time you need them. For example, to get the response function *VSWR* (which calculates the voltage standing-wave ratio of a one-port network), which is stored in the workspace 100 *MARTHAR*, you simply type

```
)COPY 100 MARTHAR VSWR
```

You can copy several such functions, one after another.

This library is open-ended, in the sense that new functions can be put in from time to time. Each of these workspaces has a variable entitled *DESCRIBE* which can be printed for up-to-date information.

4.1 Elements

Additional elements, besides the 16 in *MARTHA*, are stored in the workspace 100 *MARTHAE*. These include basic building blocks for model making; specialized elements; composite tee and pi structures; specific interpretations of numerical functions of frequency (*FOF*'s); and some miscellaneous elements of primarily academic interest.

Basic Building Blocks. Included in this workspace are 16 kinds of controlled sources. The current-controlled current source *CCCS* requires an argument with one value, which is the current gain. The input port is short-circuited, and the output current is set to the current gain times the input current. The current-controlled voltage source *CCVS* is similar: the input is short-circuited, and the output voltage is set equal to the input current times the value of the argument. Also available are the voltage-controlled current source *VCCS* and the voltage-controlled voltage source *VCVS*, each of which results in the input being open-circuited. Each requires an argument of length one.

Controlled sources involving charge (integral of the current) and flux (integral of the voltage) are also available. Each of the four types of variable can control any one of the four types of variable, so there are in all 16 controlled sources, four of which are the common ones mentioned above. The others are named *CCFS*, *CCQS*, *VCFS*, *VCQS*, *FCCS*, *FCFS*, *FCQS*, *FCVS*, *QCCS*, *QCFS*, *QCQS*, and *QCVS*. All require an argument of length one.

A gyrator is created by the function *GYRATOR*. Its argument, of length one, is interpreted as the characteristic resistance of the gyrator.

Two negative-impedance converters are available. One, *VNIC*, is a voltage-inverting converter, and the other, *INIC*, is a current-inverting converter. Neither requires an argument.

Specialized Elements. A "quick and dirty" transistor model *BETAIC* is available. It requires an argument with length two; the two values are interpreted as the current gain 3, and the collector bias current I_C. Parasitic capacitances are not included, so *BETAIC* is not useful at high frequencies.

The element *ATTENUATOR* requires an argument of length two. The two values are the characteristic impedance of the attenuator in ohms, and the attenuation in decibels. For example, a 50-ohm 10-dB attenuator,

```
      A1←ATTENUATOR 50,10
```

Since the attenuation is in dB, moderate values (up to, say, a few hundred) will work but larger values can lead to overflows.

The element *ISOLATOR* requires an argument with either one, two, or three values. It simulates an isolator, either perfect or imperfect. If the argument has length three, the first value is interpreted as the characteristic impedance of the isolator, the second value as the loss in dB in the forward direction (from input to output), and the third value as the loss in dB in the reverse direction (from output to input). Normally the reverse loss will be greater than the forward loss, but if you want an isolator pointing backward you can make the forward loss high and the reverse loss zero. If the argument has length two, the two values are the characteristic impedance and the forward loss; the reverse loss is taken to be infinite. If the argument has length one, it is the characteristic impedance, and the reverse loss is assumed to be infinite and the forward loss zero. Note that if you wish a finite reverse loss, you must specify a value for the forward loss, although you can specify it as zero if you wish.

Composite Tee and Pi Structures. The function *CPI* requires an argument with length three; the three values are the capacitances, in farads, of the input, middle, and output capacitors of a pi network. The functions *LPI* and *RPI* are similar---each requires an argument of length three, with the three values interpreted as the inductance (or resistance) of the input, middle, and output inductors (or resistors) of a pi network.

Example: *RPI* 3,6,7 is equivalent to

```
      (WP R 3)WC(WS R 6)WC WP R 7
```

The functions *CTEE*, *LTEE*, and *RTEE* each require an argument with length three, the values being the capacitance, inductance, or resistance of each of the three components in a tee network.

Example: *RTEE* 13,2,5 is equivalent to

(*WS R* 13) *WC* (*WP R* 2) *WC WS R* 5

Functions of frequency. *MARTHA* has provisions for creating, editing, displaying and interpreting numerical functions of frequency (*FOF*'s). One purpose of *FOF*'s is to represent terminal measurements of a network. There are five functions in the workspace 100 *MARTHAE* that convert *FOF*'s into elements; these are *ZFOF*, *YFOF*, *HFOF*, *SFOF*, and *ABCDFOF*. These are used if the measurements represent, respectively, impedance, admittance, hybrid, scattering, or transmission matrices of the network.

Details on the use of these five functions are given in Section 3.9.

Exotic Elements. There are several elements that are probably of more academic than practical interest. You may have fun playing around with them. In so doing you may discover practical uses for some of them.

Probably the most unusual is the element *NULLOR*, which requires no argument[12]. It has the unusual property that the input voltage and current are both zero, and the output voltage and current are both arbitrary.

L. O. Chua has introduced three classes of two-port elements that are linear, even though their major use is in connection with nonlinear elements. These are rotators, reflectors, and scalors. There are three rotators[13], named *RROTATOR*, *LROTATOR*, and *CROTATOR*. They all require an argument with length two: an angle of rotation in degrees, and a scale factor, either resistance, inductance, or capacitance. The R-rotator has the property that the v-i curve at the input is a replica of the v-i curve of a nonlinear resistor at the output, rotated through an angle. The C-rotator rotates curves of nonlinear capacitors, and the L-rotator rotates curves of nonlinear inductors. Rotators are reciprocal.

There are three reflectors[14] named *RREFLECTOR*, *LREFLECTOR*, and *CREFLECTOR*. They each require an argument of length two: an angle in degrees and a scale factor. Their defining property is that, instead of rotating curves, they reflect them about the specified angle. Reflectors are nonreciprocal.

There are three scalors[14], named *ISCALOR*, *VSCALOR*, and *PSCALOR*. The function *ISCALOR* requires an argument with length one, which is interpreted as a current scale factor. This two-port element has the property that the voltage at the input and output are the same, but the input current is a scaled replica of the output current, with the specified scale factor. Similarly, the function *VSCALOR* requires an argument with length one, and it is interpreted as a voltage scale factor. This two-port element has the property that the input and output currents are the same, but the input voltage is a scaled replica of the output voltage. Finally, the function *PSCALOR* requires an argument with length two: a voltage scale factor and a current scale factor. This two-port element is equivalent to a cascade of a *VSCALOR* and an *ISCALOR*.

Describe. All these functions are in the workspace 100 *MARTHAE*. From time to time new elements may be added. The variable *DESCRIBE* in that workspace can be printed at any time and contains a description of all the elements in *MARTHA*, including an up-to-date description of recently added elements. Appendix B contains this description as of the date of this manual.

4.2 Wiring Functions

Additional wiring functions, besides the 15 in *MARTHA*, are available in the workspace 100 *MARTHAW*. These include some functions that are not, strictly speaking, wiring functions, even though they are used in the same way.

Adjoint. The adjoint of a section is another section which has an impedance matrix equal to the transpose of the impedance matrix of the original network. Thus the adjoint of a one-port network is simply that network: similarly the adjoint of any reciprocal network is the same as that network. The monadic "wiring function" *WAD* takes the adjoint of any section it acts on.

Dual. The dual of a section, as defined here, is another section with impedance matrix equal to the admittance matrix of the original section, multiplied by the square of a resistance scale factor. The dyadic "wiring functions" *WDUAL*1 and *WDUAL*2 have as their left arguments a scale factor in ohms, and as their right arguments a section. The result of the operation is the dual of the right argument. The function *WDUAL*1 is for one-port sections, and *WDUAL*2 is for two-port sections.

Complex Conjugate. The "complex conjugate" of a network may be defined as that network with all impedances, admittances, and scattering coefficients replaced by their complex conjugates. The monadic "wiring function" *WCC* takes the complex conjugate of its argument, which may be either a one-port or two-port section. This operator is useful in forming conjugate-matched loads. Also, it can be used in conjunction with *ZSCALE* to form a network whose imaginary parts of impedance and admittance are unchanged, but whose real parts are multiplied by ¯1:

```
WCC ¯1 ZSCALE S1
```

Matched Termination. The "wiring function" *WTM* converts a two-port into a one-port by terminating its output port with a "matched impedance" (the impedance that is equal to the output impedance when the generator impedance is equal to the resulting input impedance). This terminating impedance is also called the image impedance Z_{I2}, and the resulting input impedance is the image impedance Z_{I1}, which is, in general, different. These image impedances are those that would result from an infinite cascade of back-to-back pairs of identical two-ports.

Thus *WTM S*1 is equivalent to the one-port impedance Z_{I1}. If the two-port *S*1 is lossless, then the image impedances are either real or imaginary, but in general they may be complex.

To print the image impedance of a two-port *S*1:

*PRINT Z OF WTM S*1

Describe. The variable *DESCRIBE* in the workspace 100 *MARTHAW* contains a summary of all wiring functions, including any that have been added to the library after this manual was written. Appendix B contains this summary as of the date of this manual.

4.3 Response Functions

Additional response functions, besides the 30 in *MARTHA*, are available in the workspace 100 *MARTHAR*. This workspace contains many two-port parameters, a few miscellaneous responses, and a function that allows numerically defined functions of frequency (*FOF*'s) to be included in the output list.

Two-Port Parameters. A two-port network has four terminal variables, V_1 and I_1 at the input, and V_2 and I_2 at the output, as indicated in Figure 4.1. There are two equations describing the linear two-port, and these usually may be written by choosing two of the four variables as independent, and expressing the other two in terms of them. There are six ways of choosing two objects from a set of four, so this means that there are six such representations for linear two-ports. These involve the impedance matrix Z, the admittance matrix Y, the hybrid matrices H and G, the forward transmission, or ABCD, matrix, and the reverse transmission matrix R:

$$\begin{aligned} V_1 &= Z_{11}I_1 + Z_{12}I_2 \\ V_2 &= Z_{21}I_1 + Z_{22}I_2 \end{aligned} \qquad (4.1)$$

$$\begin{aligned} I_1 &= Y_{11}V_1 + Y_{12}V_2 \\ I_2 &= Y_{21}V_1 + Y_{22}V_2 \end{aligned} \qquad (4.2)$$

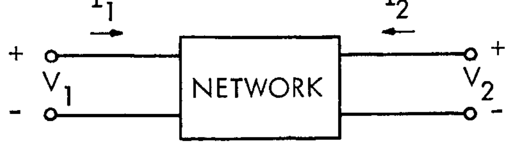


Figure 4.1. Voltages and currents at the ports of a two-port network.

$$V_1 = H_{11}I_1 + H_{12}V_2$$

$$I_2 = H_{21}I_1 + H_{22}V_2 \qquad (4.3)$$

$$I_1 = G_{11}V_1 + G_{12}I_2$$

$$V_2 = G_{21}V_1 + G_{22}I_2 \qquad (4.4)$$

$$V_1 = AV_2 - BI_2$$

$$I_1 = CV_2 - DI_2 \qquad (4.5)$$

$$V_2 = R_{VV}V_1 - R_{VI}I_1$$

$$I_2 = R_{IV}V_1 - R_{II}I_1 \qquad (4.6)$$

The response functions *Z*11, *Z*12, *Z*21, *Z*22, *Y*11, *Y*12, *Y*21, *Y*22, *H*11, *H*12, *H*21, and *H*22 are in *MARTHA*. The response functions *G*11, *G*12, *G*21, *G*22, $\underline{A}BCD$, $A\underline{B}CD$, $AB\underline{C}D$, $ABC\underline{D}$, *RVV*, *RVI*, *RIV*, and *RII* are in the workspace 100 *MARTHAR*. All of these are complex. The underlined letters are typed by hitting the backspace key and then the underscore, which is located above the F.

Scattering Two-port Parameters. Wave variables at the input and output of a two-port network are related to voltages and currents by

$$A_1 = \frac{V_1 + Z_{nin}I_1}{\sqrt{Z_{nin}}} \qquad (4.\ 7)$$

$$B_1 = \frac{V_1 - Z_{nin}I_1}{\sqrt{Z_{nin}}} \qquad (4.8)$$

$$A_2 = \frac{V_2 + Z_{nout}I_2}{\sqrt{Z_{nout}}} \qquad (4.9)$$

$$B_2 = \frac{V_2 - Z_{nout}I_2}{\sqrt{Z_{nout}}} \qquad (4.10)$$

There are six two-port matrices relating the wave variables:

$$B_1 = S_{11}A_1 + S_{12}A_2$$

$$B_2 = S_{21}A_1 + S_{22}A_2 \qquad (4.11)$$

$$A_1 = K_{11}B_1 + K_{12}B_2$$

$$A_2 = K_{21}B_1 + K_{22}B_2 \qquad (4.12)$$

$$B_1 = U_{11}A_1 + U_{12}B_2$$

$$A_2 = U_{21}A_1 + U_{22}B_2 \qquad (4.13)$$

$$A_1 = W_{11}B_1 + W_{12}A_2$$

$$B_2 = W_{21}B_1 + W_{22}A_2 \qquad (4.14)$$

$$B_1 = T_{BA}A_2 + T_{BB}B_2$$

$$A_1 = T_{AA}A_2 + T_{AB}B_2 \qquad (4.15)$$

$$B_2 = R_{BA}A_1 + R_{BB}B_1$$

$$A_2 = R_{AA}A_1 + R_{AB}B_1 \qquad (4.16)$$

The first is the scattering matrix; the response functions *S*11, *S*12, *S*21, and S22 are in *MARTHA*. The other responses, *K*11, *K*12, *K*21, *K*22, *U*11, *U*12, *U*21, *U*22, *W*11, *W*12, *W*21, *W*22, *TAA*, *TAB*, *TBA*, *TBB*, *RAA*, *RAB*, *RBA*, and *RBB*, are in the workspace 100 *MARTHAR*. All of these are complex.

Miscellaneous Responses. The open-circuit voltage gain *OCVG* is the ratio of V_2 to V_1 when the input is driven and the output is open-circuited. The short-circuit current gain *SCCG* is the ratio of $-I_2$ to I_1 when the input is driven and the output is short-circuited. These two response functions are complex.

The gain response functions *AG*, *IG*, *PG*, and *TG* in *MARTHA* are often less than one for passive networks. Many engineers prefer to think in terms of the corresponding losses, which in each case are the reciprocal of the gains. Thus the insertion loss is the reciprocal of the insertion gain; the available loss is the reciprocal of the available gain; the transducer loss is the reciprocal of the transducer gain, and the power loss is the reciprocal of the power gain. Usually there is no problem in simply using the gain functions, especially since the gains when expressed in dB are the negatives of the corresponding losses. However, if you wish to use the loss functions instead, they are in the workspace 100 *MARTHAR*, under the names *AL*, *IL*, *PL*, and *TL*. Note that *TL* is not what is sometimes called the transmission loss; it is the transducer loss. These four responses are real.

The voltage standing-wave ratio at the ports of the network are often of interest. For one-port networks, the ratio is

$$\text{VSWR} = \left|\frac{1 + |S_c|}{1 - |S_c|}\right| \tag{4.17}$$

where S_C is the reflection coefficient. The response function *VSWR* calculates this quantity, which depends upon the network and upon the normalization impedance for one-port networks *ZN*. For two-port networks, there are two such ratios, one at the input when the input is driven, and one at the output when the output is driven and the input is connected to the generator impedance *ZG*. These are given by formulas like (4.17) except with S_C replaced by either S_{in} or S_{out}. They are calculated by the response functions *VSWRIN* and *VSWROUT*. These three response functions are real.

Functions of Frequency. One purpose of numerical functions of frequency (*FOF*'s) is to store measured response of networks, or else desired response, in order to compare it with response calculated for a model or a proposed design. The response function *OUTFOF* can prepare a *FOF* for inclusion in the output list. Details are in Section 3.9.

Describe. The variable *DESCRIBE* in the workspace 100 *MARTHAR* contains an up-to-date summary of all response functions including any that have been added to the library after this manual was written. Appendix B of this manual contains this variable as of the date of this manual.

4.4 Modifiers

Additional modifiers, besides the six in *MARTHA*, are available in the workspace 100 *MARTHAM*.

Phase Delay. The phase delay of a response function is the negative of the ratio of the angle of the response function in radians, to the frequency, in radians per second. For transmission lines, the phase delay of the transfer voltage gain *VG* is equal to the time of propagation of waves down the line.

If the phase angle of the response function is greater than 360°, then the phase delay is larger than the period. *MARTHA* cannot tell if it is greater than 360°, and in fact the modifiers *DEG* and *RAD* give the angle only modulo 360° (or 2π). Similarly, the modifier *PD* gives the phase delay modulo the period. If you analyze the network from low frequencies up, you can watch the phase angle and see when it appears to jump between +180° and -180°. Count the number of such jumps and add that many periods to the calculated phase delay.

The modifier *PD* operates only on complex responses; if applied to a real response it will be ignored.

Combination Modifiers. You may obtain both the magnitude and phase of a complex response function by asking separately for each; for example,

*PLOT MAG S*21, *DEG S*21 *OF FILTER*

If you make such requests frequently, you might want to use a single modifier to request both magnitude and phase (in either radians or degrees) or magnitude in decibels and phase (in either radians or degrees). The four modifiers *MAGRAD*, *MAGDEG*, *DBRAD*, and *DBDEG* in the workspace 100 *MARTHAM* are available for these four cases. For example,

*PRINT DBRAD S*11 *OF FILT*

produces the same results as

*PRINT DB S*11, *RAD S*11 *OF FILT*

These four modifiers operate on complex responses; if you apply them to real responses they will be ignored.

Reciprocal. The modifier *REC* takes the reciprocal of the response function to its right. For complex responses, the result is complex; for real responses, the result is real.

Describe. The variable *DESCRIBE* in the workspace 100 *MARTHAM* contains an up-to-date description of all the modifiers, including any that have been added to the library after this manual was written. Appendix B of this manual contains this variable as of the date of this manual.

4.5 Miscellaneous Extra Functions

Several miscellaneous extra functions or variables that are intended to work with *MARTHA* are collected together in the workspace 100 *MARTHAX*. These include various calculations of specific kinds of transmission lines and waveguides, and several utility functions.

Transmission-Line Calculations. Recall that the element *TEM* requires the characteristic impedance in ohms, followed by the length. In place of the physical length a phrase such as 90 *DEGREESAT* 1.5*E*9 can be used. In case you prefer to specify length as so many wavelengths at a reference frequency, rather than in degrees, this can be done using the function *WAVESAT*. Example: *TEM* 50,90 *DEGREESAT* 1.5*E*9 and *TEM* 50,.25 *WAVESAT* 1.5*E*9 are identical. The function *WAVESAT* can also be used with waveguides.

MARTHA can calculate the characteristic impedance from the physical dimensions of your coaxial or microstrip transmission lines.

Use a phrase in place of the characteristic impedance. For coaxial lines the phrase is *COAX* followed by two numbers which are interpreted as the inner and outer radii.

Example:

T1←TEM COAX .068,.157,.36

The inner and outer radii may be in either order. The characteristic impedance is calculated according to the standard formula

$$Z_0 = \frac{1}{2\pi}\sqrt{\frac{\mu_0}{\varepsilon}}\ \ln\left(\frac{r_{out}}{r_{in}}\right) \tag{4.18}$$

Since only the ratio is used, the radii can be in any consistent units. Note that *COAX* does not itself produce a transmission-line element. It simply calculates Z_0 and passes it, together with the other arguments, along to *TEM*.

If two coaxial lines are connected, a reasonably good model of the discontinuous junction is a "discontinuity capacitance." The physical structure is shown in Figures 4.2 and 4.3 for the two possible cases, namely that the inner conductor is continuous or the outer conductor is continuous. The value of capacitance to use is a function of the common radius and the two discontinuous radii, and the dielectric constant of the region with the larger gap between the conductors. The dielectric constant of the narrow-gap region is not involved. The value of the capacitance is calculated by the function *COAXDISCAP*, which requires as argument a vector of length three. The first number should be the common radius, and the next two the two values of the radius of the discontinuous conductor, in either order. All the radii should be in meters. If a large-gap region has a dielectric constant different from one, remember to declare it with the function *FORDIEL*.

Examples:

C1←C COAXDISCAP .057,.102,.161
C2←C COAXDISCAP .173,.110,.077 *FORDIEL* 4.25

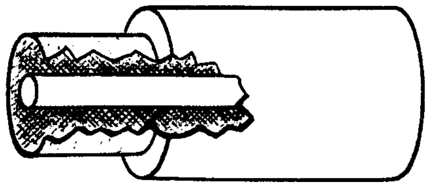


Figure 4.2. Coaxial discontinuity with inner conductor continuous. The dielectric constant to the right of the discontinuity should be used, not that to the left.

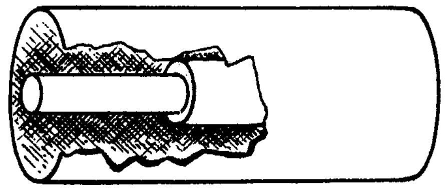


Figure 4.3. Coaxial discontinuity with outer conductor continuous. The dielectric constant to the left of the discontinuity should be used, not that to the right.

Note particularly that *COAXDISCAP* does not create a capacitor; it merely calculates the capacitance. The analytical approximations of Somlo[15] are used in the calculations. Typical values of capacitance are less than 1 pF., and Somlo gives an estimate of the maximum error in his formulas. The model of a capacitor alone is accurate only at frequencies with a wavelength greater than about 10 times the largest diameter involved, as discussed by Somlo[15].

Microstrip Calculations. A microstrip transmission line is not, strictly speaking, TEM, and it is a low-frequency approximation to consider it so. Several authors have discussed the high-frequency dispersion of microstrip[16]-[18]. Typically deviations from TEM behavior occur at frequencies greater than 1 GHz. If the TEM approximation is adequate for your purposes, a set of formulas given by Wheeler[19] is available to calculate characteristic impedance and effective dielectric constant as a function of strip width and substrate thickness. You can use these formulas in the following way. In place of the characteristic impedance, substitute the phrase *MICROSTRIP* followed by the strip width, followed by the substrate thickness. Since only the ratio is used, any consistent set of units can be used. Remember to declare the dielectric constant of the substrate with the function *FORDIEL*.

Examples:

```
T3←TEM MICROSTRIP .08,.22,.006 FORDIEL 7.6
T4←TEM MICROSTRIP .11,.065 FORDIEL 9.3
```

In these examples, *T*3 is a transmission line .6 cm long, and *T*4 is a resistor whose value is equal to the characteristic impedance of the microstrip line.

In these calculations no account is taken of end effects. Wheeler's two formulas[19] are used, one for ratios of width to thickness greater than 1.8 and the other for ratios less than 1.8. No account is taken of the high-frequency dispersive effects or the effects of the enclosure, if any. The dielectric constant above the line is assumed to be one, and the strip thickness to be negligible.

Waveguide Calculations. The characteristic impedance of a waveguide is assumed by *MARTHA* to be of the form

$$Z_0 = \frac{Z_{0\infty}}{\sqrt{1 - (f_c/f)^2}} = \frac{Z_{0\infty}\lambda_g}{\lambda} \qquad (4.19)$$

where f_c is the cutoff frequency, $Z_{0\infty}$ is the impedance at infinite frequency, λ is the free-space wavelength, and λ_g the guide wavelength. The cutoff frequency is a well defined quantity which can, at least in principle, be calculated from the cross-sectional shape and size of the waveguide. For example, for rectangular guides it is equal to the speed of light divided by twice the larger of the two dimensions. However, $Z_{0\infty}$ depends upon the size and shape of the waveguide, and also upon the precise meaning of Z_0. In general, there is no unique way to define voltage and current in waveguide, and different authors use different expressions for $Z_{0\infty}$. If Z_0 is interpreted as the wave impedance,[20] then

$$Z_{0\infty} = \sqrt{\frac{\mu_0}{\varepsilon}} \tag{4.20}$$

For rectangular guide, other definitions may be more suitable. Some of these are of the form

$$Z_{0\infty} = K \frac{b}{a} \sqrt{\frac{\mu_0}{\varepsilon}} \tag{4.21}$$

where a and b are the usual waveguide dimensions. Any choice of the constant K is consistent with Marcuvitz's equivalent circuit[21] for a step change in the value of b. Ishii[22] has discussed several possible choices for K, including 1, $\pi/2$, $\pi^2/8$, and 2. Getsinger and Maggiacomo[23] have used K = 2 because it most closely matches a probe or packaged diode[24] mounted across the narrow dimension of the guide. If (4.21) is used for junctions of waveguides with different values of a, then a transformer is required as part of the junction model[25].

An alternate definition of characteristic impedance for rectangular guide is of the form of (4.19) with

$$Z_{0\infty} = Kb \sqrt{\frac{\mu_0}{\varepsilon}} \tag{4.22}$$

where K is a constant with dimensions of inverse length. This has the same advantages as a formula advocated by Riblet[26]. Any choice of K is consistent with Marcuvitz's formulas for step changes in both a and b[21],[25] (provided the change in a is not too great) without transformers.

The function *RECT*1 calculates f_c and Z_0 for rectangular waveguide using (4.21) with K = 2. To use this, merely substitute *RECT*1 followed by the two waveguide dimensions, in either order, in place of f_c and $Z_{0\infty}$, in the argument of the function *WG*.

Examples:

```
W1←WG RECT1 (.0254×.4),(.0254×.9),.17
W2←WG RECT1 .0037 , .0014, 90 DEGREESAT 2E9
```

Note that *RECT*1 does not produce a waveguide element, it merely calculates f_c and $Z_{0\infty}$ and passes them, along with the rest of its argument, to *WG*.

The function *RECT*2 calculates f_c and $Z_{0\infty}$ for rectangular waveguide using (4.22) with K = 1 meter^{-1}. This function is used in exactly the same way as *RECT*1.

The dimensions of the common standard waveguide sizes are stored in the workspace 100 *MARTHAX*. The EIA classification system is used. For example, WR-90 is standard X-band waveguide, 0.400 x 0.900 inches. Table 4.1 shows the various sizes. To use one of these, simply substitute the name, such as *WR*90, in place of the two waveguide dimensions.

Table 4.1. Standard waveguide sizes. Inside dimensions are given in inches and meters. The dimensions in meters are stored in the *MARTHA* library.

NAME	*SIZE IN INCHES*	*SIZE IN METERS*
*WR*2300	23.000 *BY* 11.500	.5842 *BY* .2921
*WR*2100	21.000 *BY* 10.500	.5334 *BY* .2667
*WR*1800	18.000 *BY* 9.000	.4572 *BY* .2286
*WR*1500	15.000 *BY* 7.500	.3810 *BY* .1905
*WR*1150	11.500 *BY* 5.750	.2921 *BY* .14605
*WR*975	9.750 *BY* 4.875	.24765 *BY* .123825
*WR*770	7.700 *BY* 3.850	.19558 *BY* .09779
*WR*650	6.500 *BY* 3.250	.1651 *BY* .08255
*WR*510	5.100 *BY* 2.550	.12954 *BY* .06477
*WR*430	4.300 *BY* 2.150	.10922 *BY* .05461
*WR*340	3.400 *BY* 1.700	.08636 *BY* .04318
*WR*284	2.840 *BY* 1.340	.072163 *BY* .034036
*WR*229	2.290 *BY* 1.145	.058166 *BY* .029083
*WR*187	1.872 *BY* .872	.0475488 *BY* .0221488
*WR*159	1.590 *BY* .795	.040386 *BY* .020193
*WR*137	1.372 *BY* .622	.0348488 *BY* .0157988
*WR*112	1.122 *BY* .497	.0284988 *BY* .0126238
*WR*90	.900 *BY* .400	.02286 *BY* .01016
*WR*75	.750 *BY* .375	.01905 *BY* .009525
*WR*62	.622 *BY* .311	.0157988 *BY* .0078994
*WR*51	.510 *BY* .255	.012954 *BY* .006477
*WR*42	.420 *BY* .170	.010668 *BY* .004318
*WR*34	.340 *BY* .170	.008636 *BY* .004318
*WR*28	.280 *BY* .140	.007112 *BY* .003556
*WR*22	.224 *BY* .112	.0056896 *BY* .0028448
*WR*19	.188 *BY* .094	.0047752 *BY* .0023876
*WR*15	.148 *BY* .074	.0037592 *BY* .0018796
*WR*12	.122 *BY* .061	.0030988 *BY* .0015494
*WR*10	.100 *BY* .050	.00254 *BY* .00127
*WR*8	.080 *BY* .040	.002032 *BY* .001016
*WR*7	.065 *BY* .0325	.001651 *BY* .0008255
*WR*5	.051 *BY* .0255	.0012954 *BY* .0006477
*WR*4	.043 *BY* .0215	.0010922 *BY* .0005461
*WR*3	.034 *BY* .0170	.0008636 *BY* .0004318

Examples:

```
W3←WG RECT1 WR90,.17
W4←WG RECT2 WR340,90 DEGREESAT 3E9
```

These waveguide dimensions like *WR*90 can be used with either *RECT*1 or *RECT*2; they do not in themselves calculate cutoff frequency or impedance.

Other expressions for Z_0 would be appropriate for ridge waveguide[27], circular guide, or other shapes. The waveguide analysis done by *MARTHA* is consistent with any formulas for $Z_{0\infty}$, so long as the frequency dependence is given by (4.19). The analysis is valid above the cutoff frequency, where Z_0 is real, and also below cutoff, where Z_0 is imaginary.

The waveguide analysis in *MARTHA* is only valid where there is a single mode (either propagating or cutoff). The functions *RECT*1 and *RECT*2 assume that the larger of the two dimensions is the waveguide width, and calculate the cutoff frequency accordingly. Because of this you do not have to remember in which order to type the two dimensions. If you really want the smaller of the two to be the width, make both dimensions negative instead of positive, so that the one you intend will be algebraically larger. The functions *RECT*1 and *RECT*2 are written so that the cutoff frequency and impedance always end up positive.

Other Functions. The functions *MAKEFOF*, *COLUMNSOF*, *STORE*, *ATATIME*, *SWEEPPAR*, and *WHATIS* are located in the workspace 100 *MARTHAX*. These functions are described in Sections 3.8, 3.9, 3.10, 3.11, and 3.12.

Describe. The variable *DESCRIBE* in the workspace 100 *MARTHAX* can be printed for an up-to-date description of its contents, including recently added functions. Appendix B contains this description as of the date of this manual.

Chapter 5

Future Plans for *MARTHA*

5.1 Bugs

MARTHA has been thoroughly tested, and as far as is known, will work as indicated in this manual. If you discover a bug, please do the following:

1. Isolate the bug as far as you can, by running the simplest possible case where the bug shows up.
2. Specifically, see whether it shows up for any of the examples in this manual. If so, this probably indicates a fault only for your machine's version.
3. Save your printed output, in as simple a case as possible, and send it, with a description of the bug, to the Manager of Software Services, The MIT Press, Cambridge, Mass. 02142.

5.2 Additions to the Library

The *MARTHA* library is open-ended, and additions can be made from time to time. Each of the workspaces in the library has a variable entitled *DESCRIBE* which can be printed for up-to-date summaries of the contents of the library.

5.3 Extensions to *MARTHA*

Extensions to *MARTHA* currently under consideration include additional elements, additional wiring functions, additional response functions, additional modifiers, and additional types of format. Also under consideration are generalizations to four-port elements such as directional couplers; calculation of different properties of networks, such as poles and zeros; development of synthesis libraries to work with *MARTHA*.

5.4 User's Comments

The MIT Press welcomes your suggestions for improvements to *MARTHA*. Which projects and improvements receive high priority will depend partly on criticism received from users. Please send your comments and suggestions to the Manager of Software Services, The MIT Press, Cambridge, Mass. 02142.

For your convenience a sheet for your comments is provided at the end of the manual.

Appendix A

Introduction to APL

If you have never used APL and are not a programmer, this appendix gives you all you need to know to start using *MARTHA*. For a more complete description of APL refer to the APL\360 Primer[28], or the APL\360: User's Manual[9] or a book on programing in APL.

A.1 Keyboard

Many special symbols are used in APL, and a special typing element is used. The keyboard is shown in Figure A-1. When typing you may correct errors on the input line by backspacing to the point of the error, hitting the ATTN key, and then proceeding with the corrected portion of the line. If you make another mistake, do this again, until finally the input line is perfect.

A.2 Constants

In APL numerical constants are typed as integers or real numbers with a decimal point.

Examples:

```
50
17.2357
```

For very large or very small numbers a power of ten is specified by following the number with the letter *E* (no space before *E*) followed in turn by an integer. Examples: 2537.2 may be written in any of the following ways:

```
2537.2                    2537.2E0
2.5372E3                  .25372E4
```

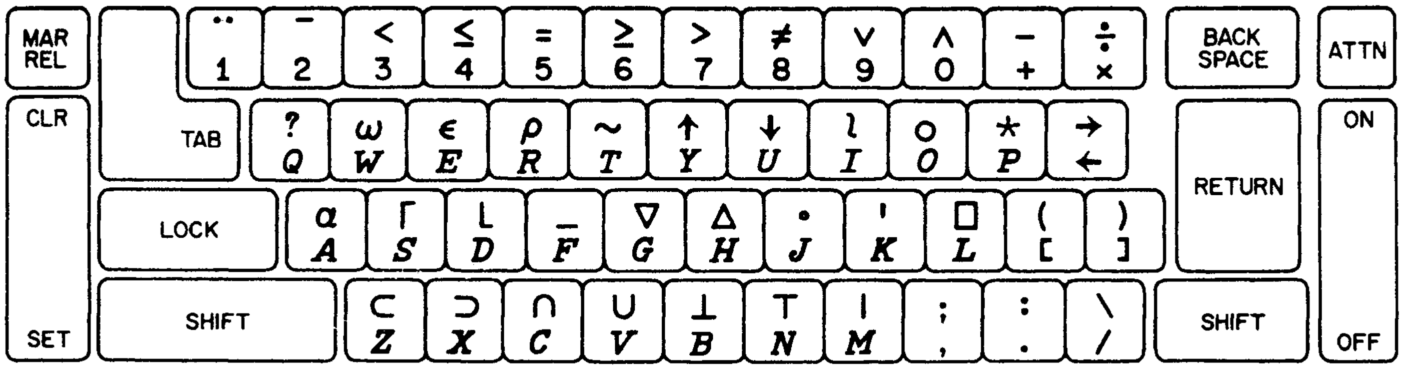


Figure A-1. Keyboard used in APL. Some of the symbols are made by overstriking. Reprinted with permission from [281. Copyright 1969 by IBM Corporation.

Negative_ numbers are indicated by preceding the number with the negation sign ¯. This is not the same as the subtraction sign, and prints higher in the line. It is located above 2 on the keyboard. A negative exponent is similarly indicated.

Examples: -2537.2 can be written in any of the following ways:

```
      ¯2537.2
      ¯2.5372E3
      ¯25372E¯1
```

APL has no provision for complex numbers. You need not consider integers to be different from other real numbers.

A.3 Variables

You may give names to variables. These names may contain letters or digits, may be of any length, but must begin with a letter. Names may also contain underlined letters (type the letter, hit the backspace, and type the underscore which is located above the letter F). When using *MARTHA*, you must avoid names already defined in *MARTHA*.

To define a variable or to reset its value, type its name, followed by ←, followed by the value or an expression that you wish evaluated. Note that ← is used for assignment rather than =. To eliminate a variable, type)*ERASE* followed by the name of the variable. To get a list of variables (including those in *MARTHA*), type)*VARS*. To find the value of a variable, simply type its name.

A.4 Expressions

In using *MARTHA* you may want to type in arithmetic expressions which, when evaluated, yield your desired answers. For example, if *MARTHA* expects a length in meters and you know the value is 1.35 inches, you may type

```
      1.35×.0254
```

since an inch is .0254 meters. You may find occasion to use the following arithmetic functions: + (addition), - (subtraction), × (multiplication, ÷ (division), ⋆ (exponentiation), and ⍟ (logarithm). Note particularly that ⋆ is not used for multiplication. The symbol for logarithm is made by overstriking ⋆ and ○, and the expression *A*⍟*B* means the logarithm of *B* to the base *A*. Ordinarily in arithmetic the + and - functions are not performed until after the × and ÷ functions, and ⋆ is performed first. In APL however, the convention is that the functions are performed in the order they appear, from right to left, except that parentheses around subexpressions force the functions inside to be performed first. For example, 2×3+4 and 2×(3+4) both evaluate to 14, whereas (2×3)+4 evaluates to 10.

Your expressions may contain constants and variables. The current value of a variable is always used when the expression is evaluated.

```
      )CLEAR
CLEAR WS

      ⍝ THIS SYMBOL AT THE BEGINNING OF A LINE INDICATES THAT THE LINE IS A COMMENT, TO BE IGNORED BY THE COMPUTER,
      ⍝ APL INDENTS SIX SPACES BEFORE YOU CAN TYPE ANYTHING. THUS THE USER'S LINES ALL APPEAR INDENTED.

      50
50
      ⍝ APL PRINTS THE VALUE OF ANY EXPRESSION YOU TYPE, IF YOU DON'T ASK IT TO DO ANYTHING ELSE WITH IT.
      ⍝ IN THIS CASE THE 'EXPRESSION' WAS VERY SIMPLE, AND THE VALUE WAS 50.

      50.00000
50
      49.99999999
 49.99999999
      17.2357
 17.2357
      2537.2
 2537.2
      2537.2E0
 2537.2
      2.5372E3
 2537.2
      .25372E4
 2537.2
      25372E¯1
 2537.2
      ¯2537.2
 ¯2537.2
      ¯2.5372E3
 ¯2537.2
      − 25372E¯1
 ¯2537.2

      A←50
      ⍝ IN THIS CASE APL DID NOT PRINT THE VALUE BECAUSE IT WAS ASSIGNED TO A VARIABLE, A.
      A
50
      A←35
      A
35
      NAMESMAYBEQUITELONG←10.2
      A←NAMESMAYBEQUITNLONG
      A
 10.2
      )VARS
 A      NAMESMAYBEQUITELONG
      )ERASE NAMESMAYBEQUITELONG
      )VARS
 A
      )ERASE A
      )VARS
      ⍝THE BLANK REPLY MEANS THERE ARE NO VARIABLES DEFINED.
      1.35×.0254
 0.03429
      INCH←.0254
      1.35×INCH
 0.03429
      INCH×1.35
 0.03429
      39.36×INCH
 0.999744
      10÷2
 5
      2⋆5
 32
      2+3
 5
      ¯2+3
 1
      -2+3
 ¯5
      ⍝ NOTICF THE DIFFERENCE BETWEEN ¯2+3 AND -2+3
      2⍟64
 6
      2×3+4
 14
      2×(3+4)
 14
      (2×3)+4
 10
      A←7
      B←6
```

The functions above are "dyadic," in the sense that they have two arguments, located to the right and the left. Each of them also has a "monadic" meaning, if the left argument is absent. These meanings are: + (unchanged in value), - (subtracted from 0), × (signum, i.e., either ¯1, 0, or 1 depending on whether the argument is negative, zero, or positive), ÷ (reciprocal), ⋆ (e raised to the power of the argument) and ⍟ (natural logarithm).

A.5 Vectors

You may wish to consider several numbers together and give them a single name. An example is a set of frequencies at which you wish *MARTHA* to do analysis. In APL, vectors are used for this. A vector constant consists of a set of numbers separated by one or more blank spaces. A variable may be set equal to a vector as well as to a scalar (single number). The functions described above extend to vectors in two ways: First, if one argument is a vector and the other a scalar the result is a vector of the same length where each element of the vector is operated on separately, using the same scalar. Second, if both arguments are vectors of the same length, the operations are performed element-by-element.

You can find the length of a vector (i.e. the number of elements in it) by the monadic operator ⍴ (located above the *R* key). Thus if the vector is 1 3 5 6 ¯17, then its length is 5 and is found by typing ⍴1 3 5 6 ¯17.

Two vectors can be catenated with the , function, so that the result is a vector consisting of the elements of the first vector followed by the elements of the second vector. The length of the result is the sum of the two lengths.

A particularly useful vector is that generated by the index generator ⍳ (located above the *I* key). This monadic function creates a vector whose length is equal to its argument (a nonnegative integer) and which consists of integers starting with 1. Thus ⍳3 is equal to 1 2 3 and ⍳*N* consists of the first *N* integers. This is useful in generating frequency sweeps for *MARTHA*.

To refer to a particular element of a vector, type its name followed by the index of the desired element in square brackets. Thus if *A* is equal to the vector 1 ¯5 13 3 7 then *A*[3] is 13, *A*[5] is 7, and *A*[*A*[4]] is 13.

A.6 User-defined Functions

To give your APL functions names, follow the rules for variable names. Type)*FNS* to get a list of all functions in your active workspace. You define functions by going from the normal "execution mode" to "definition mode", and back again, each time by typing ∇ (located above the letter *G*). Your most common use of functions will be to define networks, so that will be illustrated here. Suppose you want your network to be called *NET*. Then open the definition and type a name to be used as the value returned, followed by ← followed by the name of the

```
      A+B-1
12
      A-2×B
¯5
      -3
¯3
      ¯3
¯3
      -¯3
3
      B-A
¯1
      ×B-A
¯1
      ×¯5
¯1
      ×5
1
      ×5-5
0
      ÷5
0.2
      ⋆1
2.718281828
      ⋆0
1
      ⍟1
0
      ⍟2.71828
0.9999993273
      A←12
      ÷A
0.08333333333
      ÷÷A
12
      ⋆A
162754.7914
      ⍟⋆A
12
      ⍟A
2.48490665
      ⋆⍟A
12

      1 3 5 6 ¯17
1  3  5  6  ¯17
      ⍴1 3 5 6 ¯17
5
      A←1 2 4 ¯3
      A
1  2  4  ¯3
      A+1
2  3  5  ¯2
      A×3
3  6  12  ¯9
      B←2×A
      B
2  4  8  ¯6
      A+B
3  6  12  ¯9
      A×B
2  8  32  18
      A←0 1 2 3 4 5
      2⋆A
1  2  4  8  16  32
      A⋆2
0  1  4  9  16  25
      B←2 1 2 1 2
      7⋆B
49  7  49  7  49
      A←1 2 3 4 5
      A⋆B
1  2  9  4  25
      A+B
3  3  5  5  7
      ⍴A
5
      ⍴A+B
5
      A,B
1  2  3  4  5  2  1  2  1  2
      ⍴A,B
10
      ⍴A,3,5
7
```

function. APL will respond with [1] indicating you are expected to type the first line of the function. This might, for example, be setting the name of the value returned to an expression. This expression is not evaluated, but instead is stored as part of the function. APL will next ask you for the second line by typing [2]. You type the second line and keep on this way until you are finished. Then type ∇ to close the definition.

To look at a function, you should open the definition, name the function, request that it be typed by typing [⎕] (the quad symbol is above the *L*), and then close the definition. For example, to print out the program *NET*

```
∇NET[⎕]∇
```

To change a line in the definition, open the definition and request that line, for example ∇*NET*[2]. APL will reply [2] and you type in the new line [2]. APL will then type [3], assuming you want to change line [3]. If so, do it; if not, type ∇ to close the definition (the previous line [3] and succeeding lines will not be lost). For more sophisticated function editing, refer to the APL\360: User's Manual[9].

After a function has been defined, its name may be used in expressions. When it is encountered, APL will execute the lines of the function, in order, and evaluate the value to be returned, and then use it. A function may be called within another function, or even within itself. Thus, for example, one network definition may refer to other sections which are themselves defined by functions.

A.7 Error Messages

When APL is unable to continue for any reason, it prints out an error message and then points to the spot where it got into trouble. Until you learn to take advantage of the troubleshooting capabilities of APL, the easiest way to recover from errors, after an error message, is to type → to clear your workspace of temporary variables, correct the trouble, and try again.

```
      ⍳3
1  2  3
      ⍳21
1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21
      N←17
      ⍳N
1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17

      A←1 ¯5 13 3 7
      ⍴A
5
      A[3]
13
      A[5]
7
      A[A[4]]
13

      ∇RET←FCN
[1]  RET←A+3
[2]  ∇
      ∇FCN[⎕]∇
   ∇  RET←FCN
[1]   RET←A+3
   ∇
      A
1  ¯5  13  3  7
      FCN
4  ¯2  16  6  10
      A←⍳6
      FCN
4  5  6  7  8  9
      ⍴FCN
6
      ⍴A
6

      ∇R←NEWFCN
[1]  R←FCN×2
[2]  ∇
      A←⍳3
      A
1  2  3
      FCN
4  5  6
      NEWFCN
8  10  12
      A←0 ¯3,A
      A
0  ¯3  1  2  3
      NEWFCN
6  0  8  10  12
      NEWFCN÷16
0.375  0  0.5  0.625  0.75

      )VARS
A       B       INCH    N
      )FNS
FCN     NEWFCN
```

Appendix B

On-Line Documentation

The next few pages of this manual contain a copy of the *MARTHA* on-line documentation, as of the date of this manual. Most of this is in the workspace 100 *HOWMARTHA*, but some of it is in the workspaces in the *MARTHA* library.

```
MARTHA    71∘A    1 JULY 1971

MARTHA IS A SET OF FUNCTIONS THAT ANALYZE LINEAR ELECTRICAL NETWORKS,
NORMALLY AS A FUNCTION OF FREQUENCY.  FOR A COMPLETE DESCRIPTION, SEE
PAUL PENFIELD JR., 'MARTHA USER'S MANUAL,' THE MIT PRESS, CAMBRIDGE,
MASS. 02142; 1971.  FOR A SUCCINCT SUMMARY:

      )LOAD 100 HOWMARTH

THIS WORKSPACE CONTAINS THE FOLLOWING SUMMARIES (BUT NOT THE FUNCTIONS)

DESCRIBE      GENERAL SUMMARY
ELEMENTS      NETWORK ELEMENTS AVAILABLE
WIRING        WIRING FUNCTIONS AVAILABLE
RESPONSES     RESPONSE FUNCTIONS
MODIFIERS     MODIFIERS FOR RESPONSES
FORMATS       FORMS OF OUTPUT
EXTRA         FUNCTIONS TO WORK WITH MARTHA
HINTS
EXAMPLES
CHANGES       SINCE THE USER MANUAL
ERRORS        MARTHA ERROR MESSAGES
ORDERFORM
COMMENTS

TO GET THE FUNCTIONS:

      )COPY 100 MARTHA

MARTHA INCLUDES SEVERAL VARIABLES AND FUNCTIONS THAT THE USER DOES NOT
DIRECTLY USE.  THESE ALL HAVE NAMES CONTAINING TWO UNDERLINED LETTERS.
THERE ARE SIX VARIABLES THAT THE USER CAN CHANGE:

VARIABLE   PRESET TO          MEANING
--------   ---------   -----------------------------------------
F          1  2        FREQUENCY VECTOR, IN HZ
ZG         0           GENERATOR IMPEDANCE IN OHMS
ZL         1E25        LOAD IMPEDANCE IN OHMS
ZN         50          NORMALIZATION IMPEDANCE OF 1-PORT, IN OHMS
ZNIN       50          NORMALIZATION IMPEDANCE AT INPUT, IN OHMS
ZNOUT      50          NORMALIZATION IMPEDANCE AT OUTPUT, IN OHMS

MARTHA INCLUDES 75 OTHER VARIABLES AND FUNCTIONS.  THE MARTHA LIBRARY
CONTAINS OVER 100 ADDITIONAL VARIABLES AND FUNCTIONS IN 5 WORKSPACES:

 WORKSPACE             CONTENTS
-----------   ---------------------------------
100 MARTHAE   ELEMENTS
100 MARTHAW   WIRING FUNCTIONS
100 MARTHAR   RESPONSE FUNCTIONS
100 MARTHAM   MODIFIERS FOR RESPONSE FUNCTIONS
100 MARTHAX   EXTRA FUNCTIONS TO WORK WITH MARTHA

TO GET A SPECIFIC FUNCTION, E.G. VCCS FROM MARTHAE:

      )COPY 100 MARTHAE VCCS

FOR A SUMMARY OF THE CONTENTS OF THE MARTHA LIBRARY, PRINT THE VARIABLE
NAMED 'DESCRIBE' IN EACH WORKSPACE.
```

```
MARTHA    71∘A    1 JULY 1971

ELEMENTS IN MARTHA:

     ELEMENT         TYPE     NAME               ARGUMENT VECTOR              REQUIRED           EQUATIONS
------------------  ------  --------  --------------------------------------  --------  -----------------------------
RESISTOR            1-PORT  R         RESISTANCE  RES  IN OHMS                          V=RES×I

CAPACITOR           1-PORT  C         CAPACITANCE  CAP  IN FARADS                       I=S×CAP×V

INDUCTOR            1-PORT  L         INDUCTANCE  IND  IN HENRIES             IND≠0     V=S×IND×I

STRAIGHT-THROUGH    2-PORT  WTHRU     (NONE)                                            V1=V2; I1=-I2
 CONNECTION

POLARITY REVERSE    2-PORT  WR        (NONE)                                            V1=-V2; I1=I2

MUTUAL INDUCTOR     2-PORT  L         INPUT SELF-INDUCTANCE  L1  IN HENRIES   M≠0       V1=S×(L1×I1)+M×I2
                                      OUTPUT SELF-INDUCTANCE  L2  IN HENRIES            V2=S×(M×I1)+L2×I2
                                      MUTUAL INDUCTANCE  M  IN HENRIES

IDEAL TRANSFORMER   2-PORT  IT        TURNS RATIO  N                          N≠0       V1=N×V2; I1=-I2÷N

OPERATIONAL         2-PORT  OPAMP     OPEN-CIRCUIT VOLTAGE GAIN  A            A≠0       V1=RIN×I1
 AMPLIFIER                            OUTPUT IMPEDANCE  ROUT  IN OHMS         RIN≠0     V2=(A×V1)+ROUT×I2
                                      INPUT IMPEDANCE  RIN  IN OHMS

OPERATIONAL         2-PORT  OPAMP     OPEN-CIRCUIT VOLTAGE GAIN  A            A≠0       I1=0
 AMPLIFIER                            OUTPUT IMPEDANCE  ROUT  IN OHMS                   V2=(A×V1)+ROUT×I2

OPER. AMPLIFIER     2-PORT  OPAMP     VOLTAGE GAIN  A                         A≠0       I1=0; V2=A×V1

FIELD-EFFECT        2-PORT  FET       GATE-SOURCE CAPACITANCE CGS  IN FARADS  GM≠0      I1=S×(CGS×V1)+CGD×V1-V2
 TRANSISTOR MODEL,                    GATE-DRAIN CAPACITANCE  CGD  IN FARADS            I2=(GM×V1)+S×CGD×V2-V1
 GROUNDED-SOURCE                      TRANSCONDUCTANCE  GM  IN MHOS

BIPOLAR-TRANSISTOR  2-PORT  HYBRIDPI  RESISTANCE  RX  IN OHMS                 RPI≠0     V1=VPI+RX×I1
 MODEL, GROUNDED-                     RESISTANCE  RPI  IN OHMS                GM≠0      I1=(VPI×(S×CPI)+÷RPI)+
 EMITTER                              CAPACITANCE  CPI  IN FARADS                           S×CMU×VPI-V2
                                      CAPACITANCE  CMU  IN FARADS                       I2=(GM×VPI)+S×CMU×V2-VPI
                                      TRANSCONDUCTANCE  GM  IN MHOS

LOSSLESS TRANS-     2-PORT  TEM       CHARACTERISTIC IMPEDANCE  Z0  IN OHMS   Z0≠0      A=★-J×○2×LEN×F÷3E8÷DIEL★.5
 MISSION LINE                         LENGTH  LEN  IN METERS                            (V2-Z0×I2)=A×V1+Z0×I1
                                                                                        (V1-Z0×I1)=A×V2+Z0×I2

TRANSMISSION LINE   1-PORT  TEM       CHARACTERISTIC IMPEDANCE  Z0  IN OHMS   Z0≠0      V=Z0×I
 CHARACTERISTIC
 IMPEDANCE

LOSSLESS WAVEGUIDE  2-PORT  WG        CUTOFF FREQUENCY  FC  IN HERTZ          ZINF≠0    Z0=ZINF÷(1-(FC÷F)★2)★.5
 DOMINANT MODE                        INFINITE-FREQUENCY CHARACTERISTIC       ~FC∈F     A=J×LEN×ZINF×F÷Z0×3E8÷DIEL★.5
                                       IMPEDANCE  ZINF  IN OHMS                         (V2-Z0×I2)=(★-○2×A)×V1+Z0×I1
                                      LENGTH  LEN  IN METERS                            (V1-Z0×I1)=(★-○2×A)×V2+Z0×I2

WAVEGUIDE CHARAC-   1-PORT  WG        CUTOFF FREQUENCY  FC  IN HERTZ          ZINF≠0    V=I×ZINF÷(1-(FC÷F)★2)★.5
TERISTIC IMPEDANCE                   INF.-FREQ. CHAR. IMP.  ZINF  IN OHMS    ~FC∈F


 ∘      ∘       ∘     ∘--------∘  ∘-      -∘  ∘--- ---∘  ∘--   --∘  ∘------  --/\/\--∘  ∘--  --/\/\--∘  ∘--  ------∘
 |      |       |                   \    /      .|M|.     .|N:1|.       | +  | ROUT       +  | ROUT       +  |
 \      |       ⊃                    \  /        ⊃ ⊂       ⊃   ⊂        \    |               |               |
 /     ---      ⊃                     \/         ⊃ ⊂       ⊃   ⊂        /V1  ⊥+         V1  ⊥+          V1  ⊥+
 \RES  ---CAP   ⊃IND                  /\       L1⊃ ⊂L2     ⊃   ⊂     RIN\   (~)A×V1         (~)A×V1         (~)A×V1
 /      |       ⊃                    /  \        ⊃ ⊂       ⊃   ⊂        /    ⊤-              ⊤-              ⊤-
 |      |       |                   /    \       | |       |   |        | -  |            -  |            -  |
 ∘      ∘       ∘     ∘--------∘  ∘-      -∘  ∘--- ---∘  ∘--   --∘  ∘------  --------∘  ∘--  --------∘  ∘--  ------∘

R RES  C CAP   L IND    WTHRU         WR      L L1,L2,M    IT N      OPAMP A,ROUT,RIN    OPAMP A,ROUT     OPAMP A


∘-------------||------------∘  ∘--/\/\----------------||------------∘  ∘-|‾‾‾‾‾|-∘    ∘     -----------        ∘
GATE |    +   CGD  |    DRAIN  BASE RX |     |    +   CMU | COLLECTOR     -----       |     |\        \        |
     |             ⊥                   \     |            ⊥              |←LEN→|      \     | \ FC,ZINF \      \
 CGS---           (↓)GM×VGS            / CPI---          (↓)GM×VPI                    /     ---\        \      /
    ---  VGS       ⊤               RPI \    ---  VPI      ⊤             Z0            \Z0    \  -----------    \Z0(F)
     |             |                   /     |            |                           /       \ |         |    /
     |    -        |   SOURCE          |     |    -       |   EMITTER     _____       |        \|←--LEN--→|    |
∘---------------------------∘  ∘------------------------------------∘  ∘-|_____|-∘    ∘         -----------    ∘

       FET CGS,CGD,GM                HYBRIDPI RX,RPI,CPI,CMU,GM        TEM Z0,LEN   TEM Z0   WG FC,ZINF,LEN  WG FC,ZINF
```

ELEMENTS IN 100 *MARTHAE*:

ELEMENT	*TYPE*	*NAME*	*ARGUMENT VECTOR*	*REQUIRED*	*EQUATIONS*
SIMPLE TRANSISTOR	*2-PORT*	*BETAIC*	*CURRENT GAIN BETA*	*BETA*≠0	*I2=-BETA×I1*
MODEL			*COLLECTOR BIAS CURRENT IC IN AMPS*	*IC*≠0	*V1=I1×BETA÷40×IC*
ATTENUATOR	*2-PORT*	*ATTENUATOR*	*CHARACTERISTIC IMPEDANCE Z0 IN OHMS*	*Z0*≠0	*(V2+Z0×I2)=(10★A÷20)×V1-Z0×I1*
			ATTENUATION A IN DECIBELS	1000>|*A*	*(V1+Z0×I1)=(10★A÷20)×V2-Z0×I2*
ISOLATOR	*2-PORT*	*ISOLATOR*	*CHARACTERISTIC IMPEDANCE Z0 IN OHMS*	*Z0*≠0	*(V2+Z0×I2)=(10★A÷20)×V1-Z0×I1*
			FORWARD LOSS B IN DECIBELS	1000>|*B*	*(V1+Z0×I1)=(10★B÷20)×V2-Z0×I2*
			REVERSE LOSS A IN DECIBELS	1000>|*A*	
ISOLATOR WITH INF.	*2-PORT*	*ISOLATOR*	*CHARACTERISTIC IMPEDANCE Z0 IN OHMS*	*Z0*≠0	*V1=Z0×I1*
REVERSE LOSS			*FORWARD LOSS B IN DECIBELS*	1000>|*B*	*(V1+Z0×I1)=(10★B÷20)×V2-Z0×I2*
IDEAL ISOLATOR	*2-PORT*	*ISOLATOR*	*CHARACTERISTIC IMPEDANCE Z0 IN OHMS*	*Z0*≠0	*V1=Z0×I1; (V2-Z0×I2)=V1+Z0×I1*
GYRATOR	*2-PORT*	*GYRATOR*	*CHARACTERISTIC RESISTANCE Z0 IN OHMS*	*Z0*≠0	*V1=-Z0×I2; V2=Z0×I1*
NEGATIVE-IMPEDANCE	*2-PORT*	*VNIC*	*(NONE)*		*V1=-V2; I1=-I2*
CONVERTER					
NEGATIVE-IMPEDANCE	*2-PORT*	*INIC*	*(NONE)*		*V1=V2; I1=I2*
CONVERTER					
CURRENT-CONTROLLED	*2-PORT*	*CCCS*	*CURRENT GAIN AC*	*AC*≠0	*V1=0; I2=-AC×I1*
CURRENT SOURCE					
CURRENT-CONTROLLED	*2-PORT*	*CCFS*	*TRANSINDUCTANCE LM IN HENRIES*	*LM*≠0	*V1=0; V2=S×LM×I1*
FLUX SOURCE					
CURRENT-CONTROLLED	*2-PORT*	*CCQS*	*TRANSFER TIME TM IN SECONDS*	*TM*≠0	*V1=0; I2=-S×TM×I1*
CHARGE SOURCE					
CURRENT-CONTROLLED	*2-PORT*	*CCVS*	*TRANSRESISTANCE RM IN OHMS*	*RM*≠0	*V1=0; V2=RM×I1*
VOLTAGE SOURCE					
FLUX-CONTROLLED	*2-PORT*	*FCCS*	*TRANSFER INVERSE INDUCTANCE HM IN*	*HM*≠0	*I1=0; I2=-HM×V1÷S*
CURRENT SOURCE			*INVERSE HENRIES*		
FLUX-CONTROLLED	*2-PORT*	*FCFS*	*FLUX GAIN AF*	*AF*≠0	*I1=0; V2=AF×V1*
FLUX SOURCE					
FLUX-CONTROLLED	*2-PORT*	*FCQS*	*TRANSCONDUCTANCE GM IN MHOS*	*GM*≠0	*I1=0; I2=-GM×V1*
CHARGE SOURCE					
FLUX-CONTROLLED	*2-PORT*	*FCVS*	*TRANSFER INVERSE TIME FM IN*	*FM*≠0	*I1=0; V2=FM×V1÷S*
VOLTAGE SOURCE			*INVERSE SECONDS*		
CHARGE-CONTROLLED	*2-PORT*	*QCCS*	*TRANSFER INVERSE TIME FM IN*	*FM*≠0	*V1=0; I2=-FM×I1÷S*
CURRENT SOURCE			*INVERSE SECONDS*		
CHARGE-CONTROLLED	*2-PORT*	*QCFS*	*TRANSRESISTANCE RM IN OHMS*	*RM*≠0	*V1=0; V2=RM×I1*
FLUX SOURCE					
CHARGE-CONTROLLED	*2-PORT*	*QCQS*	*CHARGE GAIN AQ*	*AQ*≠0	*V1=0; I2=-AQ×I1*
CHARGE SOURCE					
CHARGE-CONTROLLED	*2-PORT*	*QCVS*	*TRANSFER INVERSE CAPACITANCE SM*	*SM*≠0	*V1=0; V2=SM×I1÷S*
VOLTAGE SOURCE			*IN DARAFS*		
VOLTAGE-CONTROLLED	*2-PORT*	*VCCS*	*TRANSCONDUCTANCE GM IN MHOS*	*GM*≠0	*I1=0; I2=-GM×V1*
CURRENT SOURCE					
VOLTAGE-CONTROLLED	*2-PORT*	*VCFS*	*TRANSFER TIME TM IN SECONDS*	*TM*≠0	*I1=0; V2=S×TM×V1*
FLUX SOURCE					
VOLTAGE-CONTROLLED	*2-PORT*	*VCQS*	*TRANSCAPACITANCE CM IN FARADS*	*CM*≠0	*I1=0; I2=-S×CM×V1*
CHARGE SOURCE					
VOLTAGE-CONTROLLED	*2-PORT*	*VCVS*	*VOLTAGE GAIN AV*	*AV*≠0	*I1=0; V2=AV×V1*
VOLTAGE SOURCE					
NULLOR	*2-PORT*	*NULLOR*	*(NONE)*		*V1=0; I1=0*
RESISTOR TEE	*2-PORT*	*RTEE*	*FIRST RESISTANCE RIN IN OHMS*	*RMID*≠0	*V1=(RIN×I1)+RMID×I1+I2*
			SECOND RESISTANCE RMID IN OHMS		*V2=(ROUT×I2)+RMID×I1+I2*
			THIRD RESISTANCE ROUT IN OHMS		
RESISTOR PI	*2-PORT*	*RPI*	*FIRST RESISTANCE RIN IN OHMS*	*RIN*≠0	*I1=(V1÷RIN)+(V1-V2)÷RMID*
			SECOND RESISTANCE RMID IN OHMS	*ROUT*≠0	*I2=(V2÷ROUT)+(V2-V1)÷RMID*
			THIRD RESISTANCE ROUT IN OHMS		

INDUCTOR TEE	*2-PORT*	*LTEE*	*FIRST INDUCTANCE LIN IN HENRIES* *SECOND INDUCTANCE LMID IN HENRIES* *THIRD INDUCTANCE LOUT IN HENRIES*	*LIN≠0* *LMID≠0* *LOUT≠0*	*V1=S×(LIN×I1)+LMID×I1+I2* *V2=S×(LOUT×I2)+LMID×I1+I2*
INDUCTOR PI	*2-PORT*	*LPI*	*FIRST INDUCTANCE LIN IN HENRIES* *SECOND INDUCTANCE LMID IN HENRIES* *THIRD INDUCTANCE LOUT IN HENRIES*	*LIN≠0* *LMID≠0* *LOUT≠0*	*I1=((V1÷LIN)+(V1-V2)÷LMID)÷S* *I2=((V2÷LOUT)+(V2-V1)÷LMID)÷S*
CAPACITOR TEE	*2-PORT*	*CTEE*	*FIRST CAPACITANCE CIN IN FARADS* *SECOND CAPACITANCE CMID IN FARADS* *THIRD CAPACITANCE COUT IN FARADS*		*V1=((I1÷CIN)+(I1+I2)÷CMID)÷S* *V2=((I2÷COUT)+(I1+I2)÷CMID)÷S*
CAPACITOR PI	*2-PORT*	*CPI*	*FIRST CAPACITANCE CIN IN FARADS* *SECOND CAPACITANCE CMID IN FARADS* *THIRD CAPACITANCE COUT IN FARADS*		*I1=S×(V1×CIN)+(V1-V2)×CMID* *I2=S×(V2×COUT)+(V2-V1)×CMID*
R-ROTATOR	*2-PORT*	*RROTATOR*	*ANGLE T IN DEGREES* *RESISTANCE RES IN OHMS*	*RES≠0*	*V1=(V2×2○T)+RES×I2×1○T* *I1=(V2÷RES÷1○T)-I2×2○T*
L-ROTATOR	*2-PORT*	*LROTATOR*	*ANGLE T IN DEGREES* *INDUCTANCE IND IN HENRIES*	*IND≠0*	*V1=(V2×2○T)+S×IND×I2×1○T* *I1=(V2÷S×IND÷1○T)-I2×2○T*
C-ROTATOR	*2-PORT*	*CROTATOR*	*ANGLE T IN DEGREES* *CAPACITANCE CAP IN FARADS*	*CAP≠0*	*V1=(V2×2○T)-I2÷S×CAP÷1○T* *I1=(-V2×S×CAP×1○T)-I2×2○T*
R-REFLECTOR	*2-PORT*	*RREFLECTOR*	*ANGLE T IN DEGREES* *RESISTANCE RES IN OHMS*	*RES≠0*	*V1=(V2×2○2×T)-RES×I2×1○2×T* *I1=(V2÷RES÷1○2×T)+I2×2○2×T*
L-REFLECTOR	*2-PORT*	*LREFLECTOR*	*ANGLE T IN DEGREES* *INDUCTANCE IND IN HENRIES*	*IND≠0*	*V1=(V2×2○2×T)-S×IND×I2×1○2×T* *I1=(V2÷S×IND÷1○2×T)+I2×2○2×T*
C-REFLECTOR	*2-PORT*	*CREFLECTOR*	*ANGLE T IN DEGREES* *CAPACITANCE CAP IN FARADS*	*CAP≠0*	*V1=(-V2×2○2×T)-I2÷S×CAP÷1○2×T* *I1=(V2×S×CAP×1○2×T)-I2×2○2×T*
I-SCALOR	*2-PORT*	*ISCALOR*	*CURRENT SCALE FACTOR A*		*V1=V2; I1=-A×I2*
V-SCALOR	*2-PORT*	*VSCALOR*	*VOLTAGE SCALE FACTOR A*		*V1=A×V2; I1=-I2*
P-SCALOR	*2-PORT*	*PSCALOR*	*VOLTAGE SCALE FACTOR AV* *CURRENT SCALE FACTOR AI*		*V1=AV×V2; I1=-AI×I2*
NUM.- DEFINED ELE.	*1-PORT*	*ZFOF*	*FOF WITH 1 OR 2 COLUMNS*		*SEE HOWZFOF IN 100 MARTHAE*
NUM.- DEFINED ELE.	*2-PORT*	*ZFOF*	*FOF WITH 3, 4, 6, OR 8 COLUMNS*		*SEE HOWZFOF IN 100 MARTHAE*
NUM.- DEFINED ELE.	*1-PORT*	*YFOF*	*FOF WITH 1 OR 2 COLUMNS*		*SEE HOWYFOF IN 100 MARTHAE*
NUM.- DEFINED ELE.	*2-PORT*	*YFOF*	*FOF WITH 3, 4, 6, OR 8 COLUMNS*		*SEE HOWYFOF IN 100 MARTHAE*
NUM.- DEFINED ELE.	*2-PORT*	*HFOF*	*FOF WITH 3, 4, 6, OR 8 COLUMNS*		*SEE HOWHFOF IN 100 MARTHAE*
NUM.- DEFINED ELE.	*1-PORT*	*SFOF*	*FOF WITH 1 OR 2 COLUMNS*		*SEE HOWSFOF IN 100 MARTHAE*
NUM.- DEFINED ELE.	*2-PORT*	*SFOF*	*FOF WITH 3, 4, 6, OR 8 COLUMNS*		*SEE HOWSFOF IN 100 MARTHAE*
NUM.- DEFINED ELE.	*2-PORT*	*ABCDFOF*	*FOF WITH 3, 4, 6, OR 8 COLUMNS*		*SEE HOWABCDFOF IN 100 MARTHAE*

MARTHA   71∘*A*   1 *JULY* 1971

WIRING FUNCTIONS IN MARTHA:

```
    -----|¯¯|        -----|¯¯|         -----|¯¯|          -----|¯¯|
    |    |S1|        |    |S1|         |    |S1|          |    |S1|
    |  --|__|        |  --|__|         |  --|__|          |  --|__|
 ∘--|  |          ∘---  |              |  |               |  |
    |  |                |           ∘-----+-------∘     ∘---  --------∘
 ∘--+--|          ∘---  |                 |
    ---+-|¯¯|        |  --|¯¯|      ∘-------------∘     ∘-------------∘
       | |S2|        |    |S2|
       --|__|        -----|__|          WP S1               WS S1

    S1 P S2          S1 S S2


 ∘--|¯¯¯¯|----|¯¯¯¯|--∘   ∘--|¯¯¯¯|--○   ∘--|¯¯¯¯|--   ∘--|¯¯¯¯|---|¯¯|
    | S1 |    | S2 |         | S1 |         | S1 | |      | S1 |   |S2|
 ∘--|____|----|____|--∘   ∘--|____|--○   ∘--|____|--   ∘--|____|---|__|

       S1 WC S2             WTO S1         WTS S1         S1 WT S2


   -----|¯¯¯¯|--            -----|¯¯¯¯|-----          -------------
   |    | S1 | |            |    | S1 |    |          |  ---------+---
   |  --|____|-+---         |  --|____|--  |          |  |        |  |
 ∘-|  |        |--+-∘     ∘-|  |         |  --∘     ∘--  --|¯¯¯¯|--  --∘
   |  |        |  |         |  |         |                 | S1 |
 ∘-+--|        |  |-∘     ∘-+--|         |  --∘     ∘--  --|____|--  --∘
   ---+-|¯¯¯¯|--  |         ---+-|¯¯¯¯|--  |          |  |        |  |
      | | S2 |    |            | | S2 |    |          |  ---------+---
      --|____|-----            --|____|-----          -------------

     S1 WPP S2                S1 WPS S2                 WN S1


   -----|¯¯¯¯|--            -----|¯¯¯¯|-----          ----------------
   |    | S1 | |            |    | S1 |    |          |              |
   |  --|____|-+---         |  --|____|--  |          --|¯¯¯¯|-----  |
 ∘--  |        |--+-∘     ∘--  |         |  --∘         | S1 |    |  |
      |        |  |            |         |            --|____|--  |  |
 ∘--  |        |  |-∘     ∘--  |         |  --∘       |        |  |  |
   |  --|¯¯¯¯|--  |         |  --|¯¯¯¯|--  |        ∘-----------  |  --∘
   |    | S2 |    |         |    | S2 |    |                      |
   -----|____|-----         -----|____|-----        ∘------------------∘

     S1 WSP S2                S1 WSS S2                WROT S1


 ∘--|¯¯¯¯¯¯¯|----|¯¯¯¯|----|¯¯¯¯¯¯¯¯|--∘   ∘--|¯¯¯¯|---|¯¯¯¯¯¯¯¯¯|---|¯¯|
    |IT N★.5|    | S2 |    |IT N★.¯5|         |INIC|   |IT(|N)★.5|   |S1|
 ∘--|_______|----|____|----|________|--∘   ∘--|____|---|_________|---|__|

              N ZSCALE S1                        N ZSCALE S1  (N<0)


 ∘-|¯¯¯¯¯¯¯|--|¯¯|  ∘-|¯¯¯¯|--|¯¯¯¯¯¯¯¯¯|--|¯¯¯¯|--|¯¯¯¯¯¯¯¯¯¯|--|¯¯¯¯|--∘
   |IT N★.5|  |S1|    |INIC|  |IT(|N)★.5|  | S1 |  |IT(|N)★.¯5|  |INIC|
 ∘-|_______|--|__|  ∘-|____|--|_________|--|____|--|__________|--|____|--∘

     N ZSCALE S1                         N ZSCALE S1  (N<0)
```

IF A 2-PORT SECTION IS EXPECTED AND A 1-PORT SECTION APPEARS, WP IS AUTOMATICALLY INVOKED TO CHANGE IT INTO A 2-PORT SECTION. IF A 1-PORT SECTION IS EXPECTED AND A 2-PORT SECTION APPEARS, WTO IS AUTOMATICALLY INVOKED TO CHANGE IT INTO A 1-PORT SECTION.

WIRING FUNCTIONS IN 100 *MARTHAW*:

```
∘--|¯¯¯¯¯¯¯¯¯¯¯|---|¯¯¯¯|---|¯¯¯¯¯¯¯¯¯¯¯¯|--∘  ∘--|¯¯¯¯¯¯¯¯¯¯¯|---|¯¯|
   |GYRATOR RES|   | S1 |   |GYRATOR -RES|        |GYRATOR RES|   |S1|
∘--|___________|---|____|---|____________|--∘  ∘--|___________|---|__|

                 RES WDUAL2 S1                         RES WDUAL1 S1
```

*WAD S*1   *HAS Z, Y, OR S-MATRIX = TRANSPOSE OF Z, Y, OR S OF S*1.
*WCC S*1   *HAS Z, Y, OR S-MATRIX = COMPLEX CONJUGATE OF Z, Y, OR S OF S*1.
*WTM S*1   *IS A* 1*-PORT FOUND BY TERMINATING S*1 *IN A MATCHED LOAD.*

```
MARTHA   71∘A    1 JULY 1971

RESPONSE FUNCTIONS IN MARTHA;  26 COMPLEX, 4 REAL:

NAME  C/R           MEANING                        DEPENDS ON
----  ---  ----------------------------------  --------------------
Z      C   IMPEDANCE OF A 1-PORT NETWORK       NETWORK
            V=Z×I
Y      C   ADMITTANCE OF A 1-PORT NETWORK      NETWORK
            I=Y×V
SC     C   REFLECTION COEFFICIENT OF 1-PORT    NETWORK, ZN
            B=SC×A

Z11    C   IMPEDANCE MATRIX                    NETWORK
Z12    C    V1=(Z11×I1)+(Z12×I2)
Z21    C    V2=(Z21×I1)+(Z22×I2)
Z22    C
Y11    C   ADMITTANCE MATRIX                   NETWORK
Y12    C    I1=(Y11×V1)+(Y12×V2)
Y21    C    I2=(Y21×V1)+(Y22×V2)
Y22    C
H11    C   HYBRID MATRIX                       NETWORK
H12    C    V1=(H11×I1)+(H12×V2)
H21    C    I2=(H21×I1)+(H22×V2)
H22    C
S11    C   SCATTERING MATRIX                   NETWORK, ZNIN, ZNOUT
S12    C    B1=(S11×A1)+(S12×A2)
S21    C    B2=(S21×A1)+(S22×A2)
S22    C

ZIN    C   INPUT IMPEDANCE  V1÷I1              NETWORK, ZL
YIN    C   INPUT ADMITTANCE  I1÷V1             NETWORK, ZL
SIN    C   INPUT REFLECTION COEFFICIENT B1÷A1  NETWORK, ZL, ZNIN
ZOUT   C   OUTPUT IMPEDANCE                    NETWORK, ZG
            V2÷I2 WHEN OUTPUT EXCITED
YOUT   C   OUTPUT ADMITTANCE                   NETWORK, ZG
            I2÷V2 WHEN OUTPUT EXCITED
SOUT   C   OUTPUT REFLECTION COEFFICIENT       NETWORK, ZG, ZNOUT
            B2÷A2 WHEN OUTPUT EXCITED

VG     C   VOLTAGE GAIN  V2÷EG                 NETWORK, ZG, ZL
AG     R   AVAILABLE GAIN  POUT,AV÷PIN,AV      NETWORK, ZG
IG     R   INSERTION GAIN                      NETWORK, ZG, ZL
            POUT(NETWORK)÷POUT(WTHRU)
PG     R   POWER GAIN  POUT÷PIN                NETWORK, ZL
TG     R   TRANSDUCER GAIN  POUT÷PIN,AV        NETWORK, ZG, ZL

WAVE-VARIABLE DEFINITIONS      NETWORK AS TERMINATED WITH GENERATOR AND LOAD
A=(V+ZN×I)÷(ZN★.5)                  - _ +  ZG  + I1→ _______ ←I2 +  ZL
B=(V-ZN×I)÷(ZN★.5)                 --(~)--/\/\-∘----|       |----∘-/\/\-
A1=(V1+ZNIN×I1)÷(ZN★.5)            |  ¯EG      V1   |NETWORK|   V2     |
B1=(V1-ZNIN×I1)÷(ZN★.5)            ------------∘----|_______|----∘------
A2=(V2+ZNOUT×I2)÷(ZNOUT★.5)                    -                 -
B2=(V2-ZNOUT×I2)÷(ZNOUT★.5)
```

RESPONSE FUNCTIONS IN 100 MARTHAR; 34 COMPLEX, 7 REAL:

NAME	C/R	MEANING	DEPENDS ON
G11	C	HYBRID MATRIX	NETWORK
G12	C	I1=(G11×V1)+(G12×I2)	
G21	C	V2=(G21×V1)+(G22×I2)	
G22	C		
$\underline{A}BCD$	C	ABCD TRANSMISSION MATRIX	NETWORK
$A\underline{B}CD$	C	V1=($\underline{A}BCD$×V2)-($A\underline{B}CD$×I2)	
$AB\underline{C}D$	C	I1=($AB\underline{C}D$×V2)-($ABC\underline{D}$×I2)	
$ABC\underline{D}$	C		
RVV	C	REVERSE TRANSMISSION MATRIX	NETWORK
RVI	C	V2=(RVV×V1)-(RVI×I1)	
RIV	C	I2=(RIV×V1)-(RII×I1)	
RII	C		
K11	C	INVERSE SCATTERING MATRIX	NETWORK, ZNIN, ZNOUT
K12	C	A1=(K11×B1)+(K12×B2)	
K21	C	A2=(K21×B1)+(K22×B2)	
K22	C		
U11	C	HYBRID SCATTERING MATRIX	NETWORK, ZNIN, ZNOUT
U12	C	B1=(U11×A1)+(U12×B2)	
U21	C	A2=(U21×A1)+(U22×B2)	
U22	C		
W11	C	HYBRID SCATTERING MATRIX	NETWORK, ZNIN, ZNOUT
W12	C	A1=(W11×B1)+(W12×A2)	
W21	C	B2=(W21×B1)+(W22×A2)	
W22	C		
TAA	C	SCATTERING TRANSMISSION MATRIX	NETWORK, ZNIN, ZNOUT
TAB	C	A1=(TAA×A2)+(TAB×B2)	
TBA	C	B1=(TBA×A2)+(TBB×B2)	
TBB	C		
RAA	C	REVERSE SCATTERING TRANSMISSION	NETWORK, ZNIN, ZNOUT
RAB	C	MATRIX	
RBA	C	A2=(RAA×A1)+(RAB×B1)	
RBB	C	B2=(RBA×A1)+(RBB×B1)	
OCVG	C	OPEN-CIRCUIT VOLTAGE GAIN	NETWORK
		V2÷V1 WHEN I2=0	
SCCG	C	SHORT-CIRCUIT CURRENT GAIN	NETWORK
		-I2÷I1 WHEN V2=0	
AL	R	AVAILABLE LOSS ÷AG	NETWORK, ZG
IL	R	INSERTION LOSS ÷IG	NETWORK, ZG, ZL
PL	R	POWER LOSS ÷PG	NETWORK, ZL
TL	R	TRANSDUCER LOSS ÷TG	NETWORK, ZG, ZL
VSWR	R	VOLTAGE STANDING-WAVE RATIO	NETWORK, ZN
		OF 1-PORT |(1+|SC)÷(1-|SC)	
VSWRIN	R	VOLTAGE STANDING-WAVE RATIO	NETWORK, ZL, ZNIN
		AT INPUT |(1+|SIN)÷1-|SIN	
VSWROUT	R	VOLTAGE STANDING-WAVE RATIO	NETWORK, ZG, ZNOUT
		AT OUTPUT |(1+|SOUT)÷1-|SOUT	
OUTFOF	R,C	ALL COLUMNS OF A FOF, USING	FOF, F
		INTERPOLATION OR EXTRAPOLATION	
		2-COLUMN FOF COMPLEX; OTHERS	
		REAL	

```
MARTHA   71∘A    1 JULY 1971

MODIFIERS IN MARTHA:

NAME      MEANING            COMMENTS
----   --------------    ------------------------------------------
RE     REAL PART         FOR COMPLEX RESPONSES.  WILL BE IGNORED IF
                          APPLIED TO A REAL RESPONSE.

IM     IMAGINARY PART    FOR COMPLEX RESPONSES.  WILL BE IGNORED IF
                          APPLIED TO A REAL RESPONSE.

MAG    MAGNITUDE         FOR COMPLEX RESPONSES, THE MAGNITUDE.  FOR
                          REAL RESPONSES, THE ABSOLUTE VALUE.

RAD    PHASE ANGLE       FOR COMPLEX RESPONSES.  WILL BE IGNORED IF
        IN RADIANS        APPLIED TO A REAL RESPONSE.

DEG    PHASE ANGLE       FOR COMPLEX RESPONSES.  WILL BE IGNORED IF
        IN DEGREES        APPLIED TO A REAL RESPONSE.

DB     MAGNITUDE IN      FOR COMPLEX RESPONSES, 20×10⍟MAGNITUDE.
        DECIBELS          FOR REAL RESPONSES, 10×10⍟ABSOLUTE VALUE.

IF NO MODIFIER IS USED, THE REAL AND IMAGINARY PARTS OF COMPLEX
RESPONSES WILL RESULT.


MODIFIERS IN  100 MARTHAM:

 NAME      MEANING          COMMENTS
------  --------------  ------------------------------------------
PD      PHASE DELAY     FOR COMPLEX RESPONSES.  WILL BE IGNORED IF
         (MOD. PERIOD)   APPLIED TO A REAL RESPONSE.

REC     RECIPROCAL      FOR REAL OR COMPLEX RESPONSES.

MAGRAD  MAGNITUDE AND   FOR COMPLEX RESPONSES.  WILL BE IGNORED IF
         PHASE (RADS.)   APPLIED TO A REAL RESPONSE.

MAGDEG  MAGNITUDE AND   FOR COMPLEX RESPONSES.  WILL BE IGNORED IF
         PHASE (DEGS.)   APPLIED TO A REAL RESPONSE.

DBRAD   MAG. (DB) AND   FOR COMPLEX RESPONSES.  WILL BE IGNORED IF
         PHASE (RADS.)   APPLIED TO A REAL RESPONSE.

DBDEG   MAG. (DB) AND   FOR COMPLEX RESPONSES.  WILL BE IGNORED IF
         PHASE (DEGS.)   APPLIED TO A REAL RESPONSE.
```

```
MARTHA   71∘A    1 JULY 1971

FORMAT FUNCTIONS IN MARTHA:

 NAME                             PURPOSE
------- ---------------------------------------------------------------
PRINT   PRINTS RESPONSE FUNCTIONS.  WITH ONLY ONE FREQUENCY, HEADING IS
         OMITTED.  'PRINT ⍳0' PRODUCES ONLY THE ONE-LINE HEADING.

PLOT    PLOTS RESPONSE FUNCTIONS VS FREQUENCY, AUTOMATICALLY CHOOSING
         SCALES.

PLOG    PLOTS RESPONSE FUNCTIONS VS FREQUENCY ON A LOGARITHMIC SCALE,
         AUTOMATICALLY CHOOSING SCALES.

TITLE   OPTIONAL.  SET THE VARIABLE  TITLE  TO ANYTHING TO BE PRINTED
         AT THE TOP OF THE NEXT OUTPUT.  (NORMALLY BLANK)

SS      OPTIONAL.  PLACE ANYWHERE IN THE OUTPUT LIST.  FORCES ALL
         DEPENDENT VARIABLES TO BE PLOTTED ON THE SAME SCALE.  IGNORED
         IN PRINTED OUTPUT.  (NORMALLY NOT ON SAME SCALE)

SYMBOLS OPTIONAL.  PLACE ANYWHERE IN THE OUTPUT LIST, PRECEEDED BY
         CHARACTERS TO BE USED AS PLOT CHARACTERS IN THE NEXT OUTPUT.
         IGNORED IN PRINTED OUTPUT.  (NORMALLY '○×=∘+≠★∆÷∇~⎕')

WIDE    OPTIONAL.  PLACE ANYWHERE IN THE OUTPUT LIST, PRECEEDED BY
         NUMBER.  NEXT OUTPUT WILL BE THAT NUMBER OF SPACES WIDE.
         IGNORED IN PRINTED OUTPUT.  (NORMALLY 50 WIDE)

HIGH    OPTIONAL.  PLACE ANYWHERE IN THE OUTPUT LIST, PRECEEDED BY A
         NUMBER.  NEXT OUTPUT WILL BE THAT NUMBER OF LINES HIGH.
         IGNORED IN PRINTED OUTPUT.  (NORMALLY 50 HIGH)

VS      OPTIONAL.  PLACE IN OUTPUT LIST BEFORE DESIRED INDEPENDENT
         VARIABLE.  NEXT OUTPUT HAS ALL OTHER RESPONSE FUNCTIONS AS
         DEPENDENT VARIABLES.  IF USED WITH  PLOT,  ALL PLOTTED AGAINST
         INDEPENDENT VARIABLE ON A LINEAR SCALE.  IF USED WITH  PLOG,
         ALL PLOTTED AGAINST INDEPENDENT VARIABLE ON A LOGARITHMIC
         SCALE.  IF USED WITH  PRINT,  INDEPENDENT VARIABLE APPEARS IN
         THE FIRST COLUMN.  (NORMALLY VS F)

OF      REQUIRED.  PLACE AT END OF OUTPUT LIST, BEFORE NETWORK DESCRIP-
         TION.

GENERAL FORM OF AN OUTPUT REQUEST IN MARTHA:

      (PRINT)
      (PLOT )  <OUTPUT LIST> OF <NETWORK>
      (PLOG )
```

```
MARTHA   71∘A    1 JULY 1971

CONTENTS OF  100 MARTHAX.  REFERENCE IS TO SECTION IN  MARTHA USER'S
 MANUAL.

   NAME    LEFT ARG  RIGHT ARG    RESULT          PURPOSE         REF
---------- -------- ----------- ----------- ---------------------- ----
WHATIS              ANYTHING                IDENTIFIES ARGUMENT    3.8

MAKEFOF             SCALAR      FOF         INTERACTIVE            3.9
                     NUMBER                  FOF-CREATER

MAKEFOF             FOF         FOF         EDITS FOF              3.9

COLUMNSOF  VECTOR   FOF         FOF         SELECTS COLUMNS OF FOF 3.9
            OF COL.
            NUMBERS

STORE               FOF                     COMBINES FOF WITH      3.10
                                             STORED

STORE               OUTPUT                  COMBINES OUTPUT        3.10
                     LIST                    WITH STORED

ATATIME    SCALAR   FREQUENCY   FOF         MANY-FREQUENCY         3.11
            NUMBER   VECTOR                  SWEEP

SWEEPPAR            PARAMETER   FOF         PARAMETER SWEEP        3.12
                     VECTOR

WAVESAT    NO. OF   REFERENCE   (SPECIAL    SETS UP CALCULATION    4.5
            WAVELEN  FREQUENCY   CODE)       OF PHYSICAL LENGTH

COAX                TWO RADII   Z0,LENGTH   CALCULATES  Z0  OF     4.5
                    LENGTH                   COAXIAL LINE

MICROSTRIP          STRIP WIDTH Z0,LENGTH   CALCULATES  Z0  OF     4.5
                    SUBSTRATE                MICROSTRIP LINE
                     THICKNESS
                    LENGTH

COAXDISCAP          COMMON RAD  CAPACITANCE CALCULATES CAP. OF     4.5
                    2 DISC RAD               COAXIAL DISCONTINUITY

RECT1               2 DIMENS    FC,ZINF,    CALCULATES FC,ZINF     4.5
                    LENGTH       LENGTH      FC←1.5E8÷A×DIEL★.5
                                             ZINF←753×B÷A×DIEL★.5

RECT2               2 DIMENS    FC,ZINF,    CALCULATES FC,ZINF     4.5
                    LENGTH       LENGTH      FC←1.5E8÷A×DIEL★.5
                                             ZINF←377×B÷DIEL★.5

WR2300 THROUGH WR3 -- SEE  HOWWR  IN  100 MARTHAX
```

```
MARTHA    71∘A    1 JULY 1971

HINTS

1.  LINEAR FREQUENCY SWEEPS
      F←FMIN+DF×0,⍳N
      F←FMIN+((FMAX-FMIN)÷N)×0,⍳N
      F←FMIN+DF×0,⍳⌈(FMAX-FMIN)÷DF
      F←FMAX-DF×N-0,⍳N

2.  LOGARITHMIC FREQUENCY SWEEP,  N  PER DECADE BETWEEN  10★I  AND  10★J
      F←10★I+(÷N)×0,⍳N×J-I

3.  ARRANGE  F  IN NUMERICAL ORDER
      F←(,F)[⍋,F]

4.  TRANSISTOR CONFIGURATIONS
      GROUNDED EMITTER:   Q1
      GROUNDED-BASE:      WROT WN Q1
      GROUNDED-COLLECTOR: WN WROT Q1
      DARLINGTON PAIR:    WN WROT(WN WROT Q1)WC WN WROT Q2
                      OR  WROT WROT(WROT Q2)WC WROT Q1

5.  INDUCTOR WITH INDUCTANCE  IND  AND Q AT FREQUENCY  FQ
      (L IND)S R ○2×FQ×IND÷Q

6.  FORMULAS FOR DE-EMBEDDING.  ASSUME  A  AND  B  ARE KNOWN, AND  X  IS
    UNKNOWN.
```

IF	THEN
A=X S B OR A=B S X	X←A S ¯1 ZSCALE B
A=X P B OR A=B P X	X←A P ¯1 ZSCALE B
A=WP X	X←WTO A
A=WS X	X←WTS A
A=X WC B	X←A WC ¯1 ZSCALE WN B
A=B WC X	X←(¯1 ZSCALE WN B)WC A
A=B WT X	X←(¯1 ZSCALE WN B)WT A
A=B WPP X OR A=X WPP B	X←A WPP ¯1 ZSCALE B
A=B WPS X OR A=X WPS B	X←A WPS WR WC ¯1 ZSCALE B
A=B WSP X OR A=X WSP B	X←A WSP WR WC ¯1 ZSCALE B
A=B WSS X OR A=X WSS B	X←A WSS ¯1 ZSCALE B
A=WN X	X←WN A
A=WROT X	X←WROT WROT A
A=N ZSCALE X	X←(÷N) ZSCALE A
A=WAD X	X←WAD A
A=WCC X	X←WCC A
A=RES WDUAL1 X	X←RES WDUAL1 A
A=RES WDUAL2 X	X←RES WDUAL2 A

```
MARTHA    71∘A    1 JULY 1971

EXAMPLES (ALSO SEE SECTIONS 2.13 AND 3.6 OF MARTHA USER'S MANUAL):

1.  TWIN-TEE FILTER
      F←1E¯3×79,81,50+2×0,⍳25
      PLOT DB VG OF((WS C 1)WC(R 1)WC WS C 1)WPP(WS R 2)WC(C 2)WC WS R 2

2.  ACTIVE MICROWAVE FILTER USING IMPATT DIODES
      F←1E9×9.5,9.53,9.54,9.58,9.59,9.62,9.66,9.69,9.7,9.74,9+.04×0,⍳25
      ZG←ZL←ZNIN←ZNOUT←50
      ∇Z←DIODE
[1]   Z←(R .59)S(L .128E¯9)S(C 3.3E¯12)P(R ¯43.3)P(L .93E¯9)S R 6.53∇
      ∇Z←TANK
[1]   Z←(TEM 10 .63E¯2)WT DIODE∇
      ∇Z←FILTER
[1]   Z←(WS C .1E¯12)WC TANK WC(WS C .03E¯12)WC TANK WC WS C .1E¯12∇
      PLOT DB IG,MAG SIN,MAG S21,VSWR OF FILTER
      PLOT 72 WIDE 43 HIGH IM S11 VS RE S11 OF FILTER

3.  CASCODE TRANSISTOR AMPLIFIER (GRAY AND SEARLE, PAGE 539)
      F←10★4+(0,⍳25)÷5
      ZG←ZL←200
      ∇Z←Q CPI
[1]   Z←HYBRIDPI 20 250,CPI,5E¯12 .4∇
      ∇Z←CASCODE CPI1
[1]   Z←(Q CPI1)WC WROT WN Q 100E¯12∇
      PLOG SS (DB VG OF CASCODE 100E¯12),DB VG OF CASCODE 50E¯12
      PLOG DB VG,MAG VG,DEG VG OF CASCODE 100E¯12

4.  LOW-PASS PROTOTYPE FILTERS (ONE OHM DOUBLY TERMINATED, ONE RAD/SEC)
    BUTTERWORTH AND 3-DB CHEBYSHEV
      F←.148, .156, .164, .008×⍳25
      ZL←ZG←1
      ∇Z←BUT1
[1]   Z←C 2∇
      ∇Z←BUT3
[1]   Z←(C 1)WC(WS L 2)WC C 1∇
      ∇Z←BUT5
[1]   Z←(C .6180)WC(WS L 1.618)WC(C 2)WC(WS L 1.618)WC C .6180∇
      ∇Z←CHEB1
[1]   Z←←C 1.9953∇
      ∇Z←CHEB3
[1]   Z←←(C 3.3487)WC(WS L .7117)WC C 3.3487∇
      ∇Z←CHEB5
[1]   Z←(C 3.4817)WC(WS L .7618)WC(C 4.538)WC(WS L .7618)WC C 3.4817∇
      PRINT (DB IG OF BUT1),(DB IG OF BUT3),DB IG OF BUT5
      PLOT SS (DB IG OF CHEB1),(DB IG OF CHEB3),DB IG OF CHEB5
      PLOT (DEG VG OF BUT5),DEG VG OF CHEB5
```

```
MARTHA    71∘A    1 JULY 1971

ERROR MESSAGES IN MARTHA (BESIDES APL ERROR MESSAGES):

 START OF MESSAGE                 MEANING
------------------  ----------------------------------------------------
ATTEMPT TO DIVIDE   ATTEMPT TO DIVIDE BY COMPLEX ZERO OR TAKE ITS
 BY ZERO             RECIPROCAL

NOT A CAPACITANCE   ARGUMENT FOR FUNCTION  C  HAS LENGTH ≠ 1

NOT A CGS, CGD, GM  ARGUMENT FOR FUNCTION  FET  HAS LENGTH ≠ 3

NOT A NETWORK       ATTEMPT TO USE RESPONSE FUNCTION WITH IMPROPER
                     NETWORK,  EG,  ZG,  ZL,  ZN,  ZNIN,  OR  ZNOUT

NOT AN FC, ZINF,    ARGUMENT FOR FUNCTION  WG  HAS LENGTH ≠ 2 OR 3
 LENGTH

NOT AN INDUCTANCE   ARGUMENT FOR FUNCTION  L  HAS LENGTH ≠ 1 OR 3

NOT AN RX, RPI,     ARGUMENT FOR FUNCTION  HYBRIDPI  HAS LENGTH ≠ 5
 CPI, CMU, GM

NOT A REFERENCE     ATTEMPT TO USE A REFERENCE FREQUENCY BELOW CUTOFF
 FREQUENCY           FREQUENCY OF A WAVEGUIDE

NOT A RESISTANCE    ARGUMENT FOR FUNCTION  R  HAS LENGTH ≠ 1

NOT A SCALE FACTOR  LEFT ARGUMENT FOR FUNCTION  ZSCALE  HAS LENGTH ≠ 1

NOT A SECTION       ATTEMPT TO USE A WIRING FUNCTION ON SOMETHING OTHER
                     THAN A SECTION OR AN ELEMENT

NOT A TURNS RATIO   ARGUMENT FOR FUNCTION  IT  HAS LENGTH ≠ 1

NOT A VOLTAGE GAIN  ARGUMENT FOR FUNCTION  OPAMP  HAS LENGTH ≠ 1, 2 OR 3

NOT A Z0, LENGTH    ARGUMENT FOR FUNCTION  TEM  HAS LENGTH ≠ 1 OR 2


ADDITIONAL ERROR MESSAGES FROM FUNCTIONS IN THE MARTHA LIBRARY:

 START OF MESSAGE                 MEANING
------------------  ----------------------------------------------------
PREVIOUS STORED     ATTEMPT TO APPEND WITH FUNCTION  STORE  WITH WRONG
 RESULTS LOST        NUMBER OF COLUMNS

REPLACE LINE [3]    ATTEMPT TO USE FUNCTION  ATATIME  OR  SWEEPPAR
                     WITHOUT HAVING EDITED THEM

NOT A BETA, IC      ARGUMENT FOR FUNCTION  BETAIC  HAS LENGTH ≠ 2

NOT A CHARGE GAIN   ARGUMENT FOR FUNCTION  QCQS  HAS LENGTH ≠ 1

NOT A CIN, CMID,    ARGUMENT FOR FUNCTION  CPI  OR  CTEE HAS LENGTH ≠ 3
 COUT

NOT A COMMON        ARGUMENT FOR FUNCTION  COAXDISCAP  HAS LENGTH ≠ 3
 RADIUS, TWO         OR COMMON RADIUS LIES BETWEEN DISCONTINUOUS RADII

NOT A CURRENT GAIN  ARGUMENT FOR FUNCTION  CCCS  HAS LENGTH ≠ 1

NOT A CURRENT       ARGUMENT FOR FUNCTION  ISCALOR  HAS LENGTH ≠ 1
 SCALE FACTOR

NOT A FLUX GAIN     ARGUMENT FOR FUNCTION  FCFS  HAS LENGTH ≠ 1

NOT A FOF           ARGUMENT FOR FUNCTION  ABCDFOF,  COLUMNSOF,  HFOF,
                     OUTFOF,  SFOF,  YFOF,  ZFOF,  OR A FUNCTION IN  100
                     MARTHAN  IS NOT A FOF, OR HAS WRONG NO. OF COLUMNS

NOT AN ANGLE,       ARGUMENT FOR FUNCTION  CREFLECTOR  OR  CROTATOR  HAS
 CAPACITANCE         LENGTH ≠ 2

NOT AN ANGLE,       ARGUMENT FOR FUNCTION  LREFLECTOR  OR  LROTATOR  HAS
 INDUCTANCE          LENGTH ≠ 2

NOT AN ANGLE,       ARGUMENT FOR FUNCTION  RREFLECTOR  OR  RROTATOR  HAS
 RESISTANCE          LENGTH ≠ 2

NOT AN INVERSE      ARGUMENT FOR FUNCTION  QCVS  HAS LENGTH ≠ 1
 CAPACITANCE

NOT AN INVERSE      ARGUMENT FOR FUNCTION  FCCS  HAS LENGTH ≠ 1
 INDUCTANCE
```

NOT AN LIN, LMID, LOUT	*ARGUMENT FOR FUNCTION LPI OR LTEE HAS LENGTH ≠ 3*
NOT AN RIN, RMID, ROUT	*ARGUMENT FOR FUNCTION RPI OR RTEE HAS LENGTH ≠ 3*
NOT A RESISTANCE	*ARGUMENT FOR FUNCTION GYRATOR OR LEFT ARGUMENT FOR FUNCTION WDUAL HAS LENGTH ≠ 1*
NOT A STRIP WIDTH, SUBSTRATE	*ARGUMENT FOR FUNCTION MICROSTRIP HAS LENGTH ≠ 2 OR 3*
NOT A TRANSCAPACITANCE	*ARGUMENT FOR FUNCTION VCQS HAS LENGTH ≠ 1*
NOT A TRANSCONDUCTANCE	*ARGUMENT FOR FUNCTION FCQS OR VCCS HAS LENGTH ≠ 1*
NOT A TRANSFER INVERSE TIME	*ARGUMENT FOR FUNCTION FCVS OR QCCS HAS LENGTH ≠ 1*
NOT A TRANSFER TIME	*ARGUMENT FOR FUNCTION CCQS OR VCFS HAS LENGTH ≠ 1*
NOT A TRANSINDUCTANCE	*ARGUMENT FOR FUNCTION CCFS HAS LENGTH ≠ 1*
NOT A TRANSRESISTANCE	*ARGUMENT FOR FUNCTION CCVS OR QCFS HAS LENGTH ≠ 1*
NOT A VOLTAGE GAIN	*ARGUMENT FOR FUNCTION VCVS HAS LENGTH ≠ 1*
NOT A VOLTAGE SCALE FACTOR	*ARGUMENT FOR FUNCTION VSCALOR HAS LENGTH ≠ 1 OR ARGUMENT FOR FUNCTION PSCALOR HAS LENGTH ≠ 2*
NOT A Z0, ATTENUATION	*ARGUMENT FOR FUNCTION ATTENUATOR HAS LENGTH ≠ 2*
NOT A Z0, FWD LOSS, REV LOSS	*ARGUMENT FOR FUNCTION ISOLATOR HAS LENGTH ≠ 1, 2 OR 3*
NOT TWO INSIDE DIMENSIONS,	*ARGUMENT FOR FUNCTION RECT, RECT1, OR RECT2 HAS LENGTH ≠ 2 OR 3*
NOT TWO RADII, LENGTH	*ARGUMENT FOR FUNCTION COAX HAS LENGTH ≠ 2 OR 3*

References

1.
G. L. Matthaei, L. Young, and E. M. T. Jones, Microwave Filters, Impedance-Matching Networks, and Coupling Structures, McGraw-Hill Book Co., New York, N.Y. 1964; Table 4.05-2(a), p. 101.
2.
T. S. Huang and R. R. Parker, Network Theory: An Introductory Course, Addison-Wesley Publishing Co., Reading, Mass; 1971; pp. 550-553.
3.
B. B. Bhattacharyya, "Realization of an All-Pass Transfer Function," Proc. IEEE, vol. 57, no. 11, pp. 2092-2093; November, 1969.
4.
Reference Data for Radio Engineers, Fifth Edition, Howard W. Sams and Co., Inc., Indianapolis, Indiana; 1968; p. 11-7.
5.
C. Toliver, private communication; 1971.
6.
W. Atwood and H. E. Stinehelfer, Sr., "A Multistub Filter for Microstripline," IEEE Trans. on Microwave Theory and Techniques, vol. MTT-16, no. 7, pp. 477-480; July, 1968.
7.
W. A. Davis and P. J. Kahn, "Coaxial Bandpass Filter Design," IEEE Trans. on Microwave Theory and Techniques, vol. MTT-19, no.4, pp. 373-380; April, 1971.
8
G. L. Matthaei, L. Young, and E. M. T. Jones, Microwave Filters, Impedance-Matching Networks, and Coupling Structures, McGraw-Hill Book Co., New York, N.Y.; 1964; pp. 302-304.
9.
A. D. Falkoff and K. E. Iverson, APL\360: User's Manual, International Business Machines Corp., White Plains, New York; 1968.
10.
W. K. Kahn, "Scattering Equivalent Circuits for Common Symmetrical Junctions," IRE Trans. on Circuit Theory, vol. CT-3, no. 2, pp. 121-127; June, 1956.
11.
R. W. P. King and C. W. Harrison, Jr., Antennas and Waves: A Modern Approach, The MIT Press, Cambridge, Mass.; 1969; from eq. (3.7.42) on p. 187.
12.
H. J. Carlin, "Singular Network Elements," IEEE Trans. on Circuit Theory, vol. CT-11, no. 1, pp. 67-72; March, 1964.
13.
L. O. Chua, "The Rotator---A New Network Component," Proc. IEEE, vol. 55, no. 9, pp. 1566-1577; September, 1967.
14.
L. O. Chua, "Synthesis of New Nonlinear Network Elements," Proc. IEEE, vol. 56, no. 8, pp. 1325-1340; August, 1968.
15.
P. I. Somlo, "The Computation of Coaxial Line Step Capacitances, IEEE Trans. on Microwave Theory and Techniques, vol. MTT-15, no. 1, pp.

48-53; January, 1967; equations on top of page 49.
16
P. Daly, "Hybrid-Mode Analysis of Microstrip by Finite-Element Methods," IEEE Trans. on Microwave Theory and Techniques, vol. MTT-19, no. 1, pp. 19-25; January, 1971.
17.
E. J. Denlinger, "A Frequency Dependent Solution for Microstrip Transmission Lines," IEEE Trans. on Microwave Theory and Techniques, vol. MTT-19, no. 1, pp. 30-39; January, 1971.
18.
R. Mittra and T. Itoh, "A New Technique for the Analysis of the Dispersion Characteristics of Microstrip Lines," IEEE Trans. on Microwave Theory and Techniques, vol. MTT-19, no. 1, pp. 47-56; January, 1971.
19
H. A. Wheeler, "Transmission-Line Properties of Parallel Strips Separated by a Dielectric Sheet," IEEE Trans. on Microwave Theory and Techniques, vol. MTT-13, no. 2, pp. 172-185; March, 1965; Eqs. (31) and (48).
20.
T. Moreno, Microwave Transmission Design Data, Dover Publications, Inc, New York, N.Y.; 1958; p. 34.
21.
N. Marcuvitz, Waveguide Handbook, vol. 10 of the M.I.T. Radiation Laboratory Series, McGraw-Hill Book Co., Inc., New York, N.Y.; 1951; Section 5.26.
22.
T. K. Ishii, Microwave Engineering, The Ronald Press Company, New York, N.Y.; 1966; pp. 46-47.
23
W. J. Getsinger and T. E. Maggiacomo, "Automatic Microwave Circuit Analysis with GCP-MOD2," M.I.T. Lincoln Laboratory, Technical Note No. 1969-34; 10 July 1969.
24.
W. J. Getsinger, "The Packaged and Mounted Diode as a Microwave Circuit," IEEE Trans. on Microwave Theory and Techniques, vol. MTT-14, no. 2, pp. 58-69; February, 1966.
25.
N. Marcuvitz (ref. 21), Section 5.24.
26.
H. J. Riblet, "A General Design Procedure for Quarter-Wavelength Inhomogeneous Impedance Transformers Having Approximately Equal-Ripple Performance," IEEE Trans. on Microwave Theory and Techniques, vol. MTT-13, no. 5, pp. 622-629; September, 1965; see especially the Appendix.
27.
T. K. Ishii (ref. 22), p. 53.
28.
P. Berry, APL\360 Primer, International Business Machines Corp., White Plains, N.Y.; 1969.

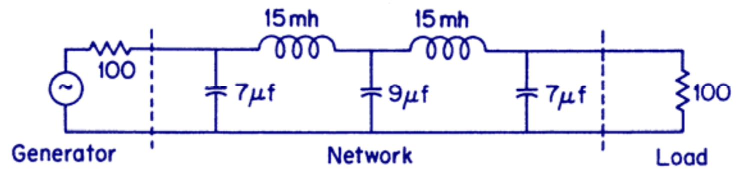
Index

MARTHA is a set of computer programs for general-purpose analysis of electrical networks.

Paul Penfield, Jr., is the author of the programs and of this manual. He is a Professor of Electrical Engineering at M.I.T. His previous books include the MIT Press publications *Frequency-Power Formulas* (1960), and as co-author *Varactor Applications* (1962), *Electrodynamics of Moving Media* (1967), and *Tellegen's Theorem and Electrical Networks* (1970).

The MIT Press
Massachusetts Institute of Technology
Cambridge, Massachusetts 02142

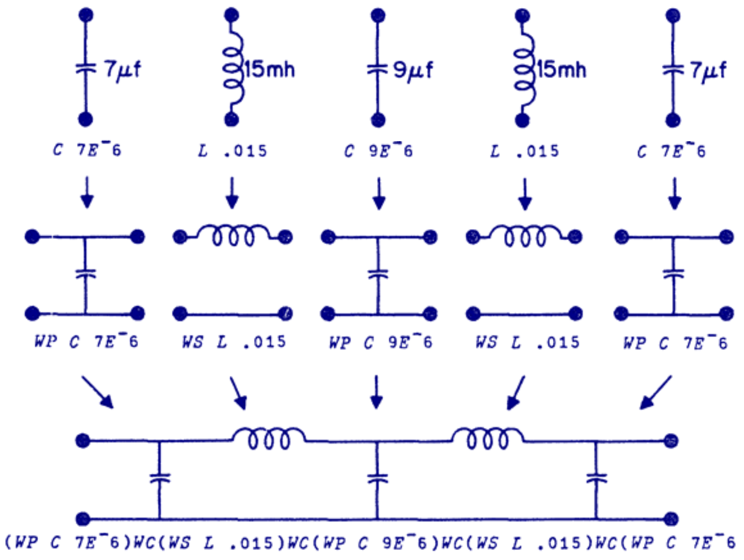
To analyze this network with MARTHA



is as simple as A-B-C .. .

A. Ask for your generator and load resistances ZG and ZL, and frequency sweep F.

B. Build the network up from its elements --



C. Command the output you want--for example the insertion gain expressed in dB, or magnitude and phase of the input impedance.

Here's how it looks:

YOU
DO THIS

```
A          ZG←100
           ZL←100
           F←200,400,600,800,1000,1200,1400,1600,1800,2000

           ∇N←NETWORK
B    [1]   N←(WP C 7E¯6)WC(WS L .015)WC(WP C 9E¯6)WC(WS L .015)WC(WP C 7E¯6)
     [2]   ∇

C          PRINT ZIN, MAG ZIN, DEG ZIN, DB IG OF NETWORK
```

MARTHA
DOES THIS

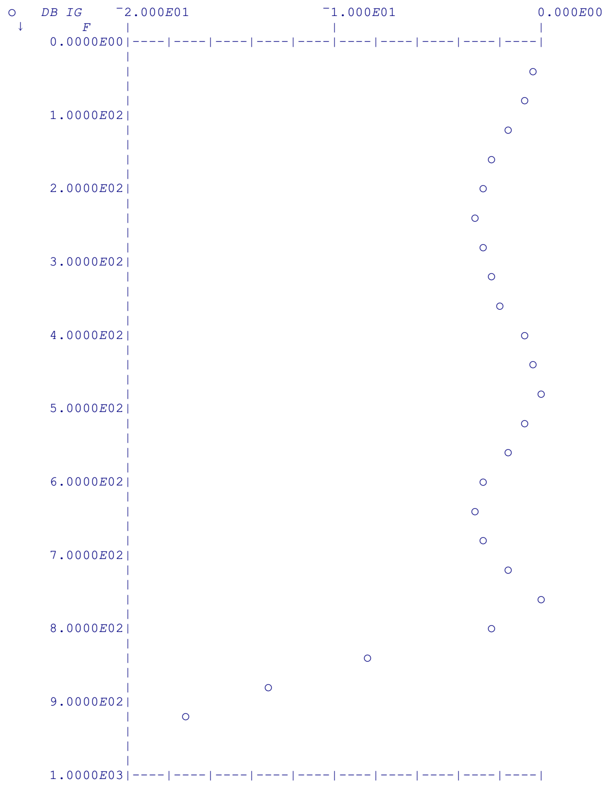
```
CIRCUIT ANALYSIS BY MARTHA.   71∘A   6/25/71   10:54

      F         RE ZIN      IM ZIN      MAG ZIN     DEG ZIN      DB IG
 ----------- ----------- ----------- ----------- ----------- -----------
  2.0000E02   1.8783E01  ¯1.7671E01   2.5789E01  ¯4.3252E01  ¯2.8318E00
  4.0000E02   4.0744E01   2.5226E01   4.7921E01   3.1763E01  ¯9.8468E¯01
  6.0000E02   2.3609E01  ¯4.4737E01   5.0584E01  ¯6.2178E01  ¯2.6242E00
  8.0000E02   4.3381E01  ¯1.0357E02   1.1229E02  ¯6.7273E01  ¯2.5598E00
  1.0000E03   1.1247E¯01 ¯3.3464E01   3.3465E01  ¯8.9807E01  ¯2.3939E01

  1.2000E03   7.1485E¯03 ¯2.3627E01   2.3627E01  ¯8.9983E01  ¯3.5674E01
  1.4000E03   9.7871E¯04 ¯1.8838E01   1.8838E01  ¯8.9997E01  ¯4.4224E01
  1.6000E03   1.9874E¯04 ¯1.5831E01   1.5831E01  ¯8.9999E01  ¯5.1104E01
  1.8000E03   5.1841E¯05 ¯1.3720E01   1.3720E01  ¯9.0000E01  ¯5.6914E01
  2.0000E03   1.6133E¯05 ¯1.2138E01   1.2138E01  ¯9.0000E01  ¯6.1966E01
```

YOU
DO THIS

```
           F←40×⍳23
           PLOT DB IG OF NETWORK
```

MARTHA
DOES THIS

```
CIRCUIT ANALYSIS BY MARTHA.   71∘A   6/25/71  10:55
```


DB IG
F
¯2.000E01
¯1.000E01
0.000E00
0.0000E00
1.0000E02
2.0000E02
3.0000E02
4.0000E02
5.0000E02
6.0000E02
7.0000E02
8.0000E02
9.0000E02
1.0000E03