https://ntrs.nasa.gov/search.jsp?R=19720022498 2020-03-11T19:35:26+00:00Z

CR-128516

(NASA-CR-128516) ADVANCED COMMUNICATION SYSTEM TIME DOMAIN MODELING TECHNIQUES ASYSTD SOFTWARE DESCRIPTION. VOLUME 1: PROGRAM USERS GUIDE (Systems Associates, Inc.) Aug. 1972 168 p CSCL 17B G3/07

N72-30148

Unclas 39624

SYSTEMS ASSOCIATES

SEP 1972
RECEIVED
NASA STI FACILITY
INPUT BRANCH

SYSTEMS ASSOCIATES, INC.
444 West Ocean Boulevard
Long Beach, California 90802

Reproduced by
NATIONAL TECHNICAL
INFORMATION SERVICE
U S Department of Commerce
Springfield VA 22151

168

12/1/72

ATTENTION REPRO:

BEFORE PRINTING, CONTACT INPUT FOR PAGINATION

PROCESSOR VH

NTIS-185(3-72)

R72-001
Contract No. NAS9-11743

ADVANCED COMMUNICATION SYSTEM TIME DOMAIN MODELING TECHNIQUES

ASYSTD SOFTWARE DESCRIPTION

VOLUME I

PROGRAM USER'S GUIDE

August 1972

Prepared for:

Space Electronics System Division
National Aeronautics Space Administration
Manned Spacecraft Center
Houston, Texas 77058

SYSTEMS ASSOCIATES

444 West Ocean Boulevard
Long Beach, California 90802
Telephone 213/435-8282

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

PREFACE

This document describes the computer program ASYSTD, which is primarily designed to simulate the transient behavior of communications systems.

The ASYSTD program is based in part on earlier work performed for NASA MSC under Contract No. NAS9-10831, the development of the SYSTID program. The program is written in FORTRAN V for the UNIVAC 1108 computer operating under EXEC II.

The document is presented in two volumes – Volume I, Program User's Guide, and Volume II, Program Support Documentation. Program listings and detailed flow charts are under separate cover.

The analyses performed during the ASYSTD development contract are published in a series of progress reports addressing the individual tasks, which include:

- Parametric Analysis
- Optimization and Statistical Analysis
- Propagation Model Development
- Frequency to Time Domain Filter Transformations
- Orthogonal Transforms
- Bit Error Rate Measurements (EVT)
- SNR Measurements
- Distortion Measurements

SECTION 1.0
PROGRAM DESCRIPTION

ASYSTD is a system of computer routines which provides the analyst with a powerful tool for the transient simulation and analysis of complex systems — in particular, telecommunications systems, although other continuous and discrete systems can be simulated.

The unique characteristic of telecommunications systems is the large ratio between the RF carrier and baseband frequencies. Simulation of such systems would normally require prohibitive run times. However, with the techniques utilized in ASYSTD, this problem is alleviated. In particular, the RF system elements can be translated to baseband with complete retention of their RF characteristics.

The program accepts as input a topological "black box" description of a system, automatically generates the appropriate algorithms, and then proceeds to execute the simulation program. Thus the user is not necessarily required to write the algorithms in a computer language nor possess a great facility in computer programming. The system description, including both topology and element information, is supplied to the program in a free-form, user controlled engineering language which is easily learned.

ASYSTD offers the user enormous flexibility in the representation of system elements, i.e., "black boxes". An element may be defined as:

1) An ASYSTD library model.

2) A user written, temporary ASYSTD model.

3) A FORTRAN arithmetic expression involving any intrinsic ASYSTD parameter, constants, variables, FORTRAN library functions, ASYSTD library functions, model output nodes (TAP's), and user supplied FORTRAN functions.

The ASYSTD model library consists of a set of computer routines, either written in FORTRAN or ASYSTD, which have been stored on a library file and cataloged in the ASYSTD directory. The user, at any time, can modify or replace the library and directory as he may choose — thus every user can easily create his own library. One unique characteristic of ASYSTD is the capability of nesting models to a level of 100 — that is, any model (or system) can reference up to 100 models, excluding itself. The nesting feature provides the user with the tools necessary to build a model library to suit his needs based upon a canonic set of models. An example might be a receiver that is used in several systems — the receiver would be a model consisting of a connection of other models.

The basic, or canonic, ASYSTD library consists mainly of a group of routines which aid in the simulation of continuous functions, that is G(s). The technique applied is that of the bi-linear z-transform representation of G(s). The transfer function may be defined in several ways — in terms of its poles and zeros or as one of the classical functions such as BESSEL, ELLIPTIC, etc. The sample data routines accomplish all the necessary transformations in addition to the numerical processing such as integration and differentiation. In addition, all of the FORTRAN arithmetic features are an intrinsic part of the ASYSTD library — although they do not appear in the directory.

The bi-linear transform rather than the standard z-transform is used in the representation of continuous functions because it eliminates aliasing errors, making possible the realization of commonly encountered functions whose response does not approach zero at high frequencies. Note that aliasing of the signals, however, is possible.

Another aspect of the ASYSTD model library is that it contains FORTRAN subroutines — that is, when a model (or system) is processed by

ASYSTD, the result is a FORTRAN subroutine (or main program) which is available to the user for any purpose, whether for ASYSTD or not. Thus, ASYSTD can be viewed as a FORTRAN program generator which converts a topological, non-procedural input into a procedural language-FORTRAN. Although not unique to ASYSTD, this aspect allows one to evaluate mathematical problems via ASYSTD with no concern for the Input/Output coding necessary in FORTRAN programs. That is, ASYSTD may be used as a shorthand FORTRAN system.

ASYSTD's flexibility is in part attained by designing the program to execute as a multipass processor in a batch mode of operation. The first phase reads the user input description of all models and/or a system and proceeds to formulate the corresponding FORTRAN algorithms. In this phase, the program checks for input errors such as erroneous model references, dangling nodes, etc., in which case appropriate error messages are issued. If the first phase terminates without fatal errors, the FORTRAN routines are automatically compiled and collected with the ASYSTD library to from the second phase, that of executing the simulation.

Output from the program includes plots as well as tabulated data. Conventional output is any system node or "TAP" which may be individually selected, or any variable whether intrinsic or user defined. Plots can be produced on the printer as well as a digital plotter. Printed data can be formatted, under user control, for either 8-1/2 by 11 inch pages or the full 11 by 14 inch page. Digital plotters are handled by the subroutine TMPLT, which is installation dependent.

The additional flexibility of linking to a user defined post processing routine is intrinsic to ASYSTD when utilizing the POST system identifier. This feature allows the user to access the time histories of any node, tap, or variable much the same way as the plot routines. As a matter of fact, the plot routines are indeed intrinsically named post-processors. Utility routines are available to perform any necessary input/output for the user.

The user, because of the two phase aspect, has available to him several techniques for controlling his computer runs and ensuring that the most effective use is made of the machine time. The primary means is that of saving the results of the first phase, that is, the collected simulation package for subsequent reruns with alternate input data. Rerun would then simply entail a load-go operation. The alternate input data can be provided at execution time by use of the "DATA" identifier in the first phase.

SECTION 2.0

DATA PREPARATION

Given a description of a system or model in an engineering oriented language, ASYSTD will automatically generate the FORTRAN code required to simulate the system or model. The user, however, must reduce the system or model to an equivalent block diagram form consisting of model references, math expressions, etc., which can be interpreted by ASYSTD. The transcription of the equivalent block diagram into the input language is straightforward and easily mastered.

The initial step in using ASYSTD is to prepare the equivalent block diagram utilizing the ASYSTD library directory and FORTRAN expressions available. The model library directory currently available is given in Figure 2-1. The usage of these internal models is explained in Appendix B. The user is not in any way restricted or limited to this library — any particular function in the library can be replaced with one's own model. Thus the user's model repertoire consists of:

1) The invoked ASYSTD library directory.

2) FORTRAN math functions, including user variables, constants, etc.

3) Any ASYSTD model defined in the same run stream.

```
@ ELT LIBARY,1,720202, 53988

.
1
2 THIS ELEMENT IS THE MODEL LIBARY DIRECTORY FOR THE SYSTID PROCESSOR.
3 ADDITIONS AND DELETIONS CAN BE MADE SIMPLY BY REMOVING OR ENTERING THE
4 DESCRIPTOR CARD.  DO NOT EMBED BLANK CARDS IN THIS ELEMENT
.
5         FORMAT
.
6    A    CC  1-36       MODEL NAME, ALPHANUMERIC, LEFT ADJUST AND NO EMBEDDED
7                        BLANKS
8    B    CC 41-46       THE ENTRY POINT NAME CORRESPONDING TO (A)
9    C    CC    50       =1 TO INDICATE SUBROUTINE, 0 FOR A FUNCTION.
10   D    CC    53       = 0-9 IS THE NUMBER OF ARGUMENTS REQUIRED FOR (B)
11   E    CC    56       = 0-9 IS THE NUMBER OF TAPS ON THE MODEL
12   F    CC 60-71       OCTAL WORD REPRESENTING TYPE OF EACH TAP,  I.E. THE
13                       OCTAL WORD MADE UP FROM A 36 BIT WORD CONTAINING
14                       A 1 FOR INPUT, 0 FOR OUTPUT IN THE BIT POSITION
15                       OF THE TAP NUMBER STARTING FROM THE LEFTMOST BIT.
15   G    CC    73       PHASE INDICATOR = 1 FOR SETUP  = 2 FOR RUN  =3 FOR POST
.
.
**   THIS CARD STARTS THE LIBRARY. DO NOT REMOVE.  COMMENTS ABOVE IT ONLY.
MODEL MODEL     DO NOT REMOVE THIS IS THE TLEMENT NAME COUNTER
FILTER                                 FILTER  1  9  0          2
GENRAL                                 GENRAL  1  9  0          2
GENERALFILTER                          GENRAL  1  9  0          2
BUTWTH                                 BUTWTH  1  6  0          2
BUTTERWORTH                            BUTWTH  1  6  0          2
BUFUNCTION                             BUTWTH  1  6  0          2
BESSEL                                 BESSEL  1  6  0          2
BEFUNCTION                             BESSEL  1  6  0          2
BUTOM                                  BUTOM   1  7  0          2
BUTTERWORTHTHOMPSON                    BUTOM   1  7  0          2
BTFUNCTION                             BUTOM   1  7  0          2
ELIPTC                                 ELIPTC  1  8  0          2
ELLIPTIC                               ELIPTC  1  8  0          2
ELFUNCTION                             ELIPTC  1  8  0          2
QFACT                                  QFACT   1  7  0          2
QFACTOR                                QFACT   1  7  0          2
QUADRATICFACTOR                        QFACT   1  7  0          2
LEADLAG                                LEDLAG  1  5  0          2
LOOPFILTER                             LEDLAG  1  5  0          2
LEADFUNCTION                           LEADIT  1  4  0          2
CHEBY                                  CHEBY   1  7  0          2
CHEBYCHEV                              CHEBY   1  7  0          2
TCHEBYCHEV                             CHEBY   1  7  0          2
AGATE                                  AGATE   1  2  0          2
AMMODULATOR                            AMMOD   1  2  0          2
AMMOD                                  AMMOD   1  2  0          2
AMPLITUDEDEMODULATOR                   AMDEM   1  0  0          2
```

Figure 2-1. Model Library

000051	AMDEM	AMDEM	1	0	0	2
000052	AMPLITUDEDEMODULATORSQUARELAW	AMDESQ	1	2	0	2
000053	AMDESQ	AMDESQ	1	2	0	2
000054	CALINB	CALINB	1	1	0	2
000055	CADD	CADD	1	0	0	2
000056	COMPLEXADDER	CADD	1	2	0	2
000057	COMPLEXMULTIPLIER	CMULT	1	2	0	2
000058	CMULT	CMULT	1	2	0	2
000059	COSINE	COSINE	0	1	0	2
000060	DELAY	DELAY	1	1	0	2
000061	DELTAMODULATOR	DELMOD	1	2	0	2
000062	DELMOD	DELMOD	1	2	0	2
000063	DIFFERENTIATOR	DIFFER	1	0	0	2
000064	DIF	DIFFER	1	0	0	2
000065	DIFFER	DIFFER	1	0	0	2
000066	FMMODULATOR	FMMOD	1	2	0	2
000067	FMMOD	FMMOD	1	2	0	2
000068	FREQUENCYDEMODULATOR	FMDEMM	1	2	0	2
000069	FMDEMM	FMDEMM	1	2	0	2
000070	FMDEM	FMDEMM	1	2	0	2
000071	FMDEMOD	FMDEMM	1	2	0	2
000072	FMDCTD	FMDEMM	1	2	0	2
000073	GNOISE	GNOISE	0	3	0	2
000074	GNOIS2	GNOIS2	0	2	0	2
000075	HARDLIMITER	HARD	1	0	0	2
000076	HARD	HARD	1	0	0	2
000077	INTEGRATOR	INTGRT	1	0	0	2
000078	INTGRT	INTGRT	1	0	0	2
000079	INTEGRALWITHINITIALCONDITIONS	INTGIC	1	1	0	2
000080	INTGIC	INTGIC	1	1	0	2
000081	MATCHEDFILTER	MFLTER	1	1	0	2
000082	MFLTER	MFLTER	1	1	0	2
000083	MONOSTABLE	MONO	1	1	0	2
000084	MONO	MONO	1	1	0	2
000085	MULTILEVELPCM	MLTPCM	1	2	0	2
000086	MLTPCM	MLTPCM	1	2	0	2
000087	NRZLBITSTREAM	NRZL	1	3	0	2
000088	NRZL	NRZL	1	3	0	2
000089	PHASEMODULATOR	PMMODD	1	2	0	2
000090	PMMOD	PMMODD	1	2	0	2
000091	PMMODD	PMMODD	1	2	0	2
000092	PHASEDEMODULATOR	PMDEMM	1	2	0	2
000093	PMDEMM	PMDEMM	1	2	0	2
000094	PHASESHIFTER	PHSHFT	1	2	0	2
000095	PHSHFT	PHSHFT	1	2	0	2
000096	PERIODICTABLEFUNCTION	PTABLE	0	10	0	2
000097	PTABLE	PTABLE	0	10	0	2
000098	PULSE	PULSE	0	5	0	2
000099	RANDOMBITGENERATOR	RBGEN	1	1	0	2
000100	RBGEN	RBGEN	1	1	0	2
000101	RFAMPLITUDEMODULATOR	RAMMOD	1	3	0	2
000102	RAMMOD	RAMMOD	1	3	0	2
000103	RFAMDEMODULATOR	RAMDEM	1	1	0	2
000104	RAMDEM	RAMDEM	1	1	0	2
000105	RFENVELOPE	RFENVE	1	1	0	2
000106	RFENVE	RFENVE	1	1	0	2

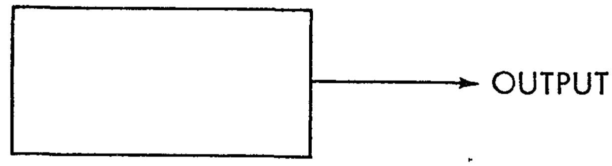
Figure 2-1. Model Library (Cont)

000107	RFFMMODULATOR	RFMMOD	1	1	0	2	
000108	RFMMOD	RFMMOD	1	1	0	2	
000109	RFFREQUENCYDEMODULATOR	RFMDEM	1	1	0	2	
000110	RFMDEM	RFMDEM	1	1	0	2	
000111	RFREQUENCY	RFFREQ	1	0	0	2	
000112	RFFREQ	RFFREQ	1	0	0	2	
000113	RFPHASEMODULATOR	RPMMOD	1	1	0	2	
000114	RPMMOD	RPMMOD	1	1	0	2	
000115	RFPHASEDEMODULATOR	RFPDEM	1	1	0	2	
000116	RFPDEM	RFPDEM	1	1	0	2	
000117	RFPHASE	RFPHAS	1	0	0	2	
000118	RFPHAS	RFPHAS	1	0	0	2	
000119	RFPHASESHIFTER	RFSHFT	1	2	0	2	
000120	RFSHFT	RFSHFT	1	2	0	2	
000121	RFLIMITER	RFLIMT	1	0	0	2	
000122	RFLIMT	RFLIMT	1	0	0	2	
000123	RFNOISE	RFNOIS	1	2	0	2	
000124	RFNOIS	RFNOIS	1	2	0	2	
000125	RFSOFTLIMITER	RFSOFT	1	2	0	2	
000126	RFSOFT	RFSOFT	1	2	0	2	
000127	SINE	SINE	0	1	0	2	
000128	SOFTLIMITER	SOFTY	1	2	0	2	
000129	SOFTY	SOFTY	1	2	0	2	
000130	SPLIT	SPLIT	1	0	0	2	
000131	SQ	SQ	1	1	0	2	
000132	SQUAREWAVE	SQ	1	1	0	2	
000133	SQUAREWAVEFREQUENCYMODULATOR	SQFMOD	1	2	0	2	
000134	SQFMMOD	SQFMOD	1	2	0	2	
000135	SQUAREWAVEPHASEMODULATOR	SQPMOD	1	2	0	2	
000136	SQPMOD	SQPMOD	1	2	0	2	
000137	TABLE	TABLE	0	11	0	2	
000138	TABL2	TABL2	0	11	0	2	
000139	ZDEMOD	ZDEMOD	1	0	0	2	
000140	ZDEMODULATOR	ZDEMOD	1	0	0	2	
000141	ZEROCROSSINGDETECTOR	ZRODET	1	0	0	2	
000142	ZRODET	ZRODET	1	0	0	2	
000143	ATOD	ATOD	1	4	0	2	*NEW
000144	DTOA	DTOA	1	4	0	2	*NEW
000145	SHDTOA	SHDTOA	1	4	0	2	*NEW
000146	INTLEV	INTLEV	1	2	0	2	*NEW
000147	DELEAV	DELEAV	1	2	0	2	*NEW
000148	FOURT	FOURT	1	2	0	2	*NEW
000149	IFOURT	IFOURT	1	2	0	2	*NEW
000150	HAAR	HAAR	1	3	0	2	*NEW
000151	IHAAR	IHAAR	1	3	0	2	*NEW
000152	HDMRD	HDMRD	1	3	0	2	*NEW
000153	IHDMRD	IHDMRD	1	3	0	2	*NEW
000154	OHMRD	OHMRD	1	3	0	2	*NEW
000155	IOHMRD	IOHMRD	1	3	0	2	*NEW
000156							
000157							
000158							
000159							
000160	$$$$$$	DO NOT REMOVE MARKS END OF LIB.					
000161							

Figure 2-1. Model Library (Cont)

There are three basic types of ASYSTD elements (black boxes) which make up a model or system:

- Mono-node elements

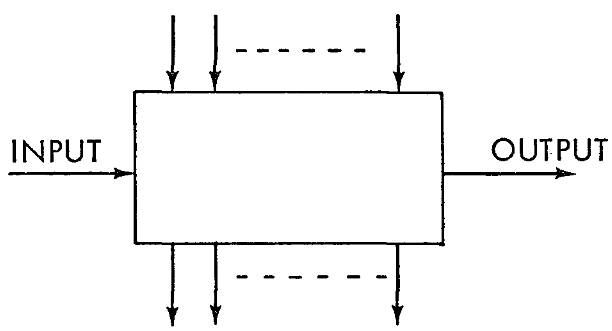


for example, a source

- Bi-node elements

for example, a filter

- Multi-node elements



for example, a mixer

The mono-node and bi-node elements are certainly the most common and most readily implemented. The multi-node element, however, may pose some problems to the inexerpeinced user. All elements or models have, by definition, one INPUT node and one OUTPUT node. Any other connection, be it an input or an output, is considered a "TAP". A "TAP" may be thought of in the conventional way – that is, as a point in the system or model which can be externally accessed either for observation or for insertion of a signal. There is no explicit distinction between the different types of elements since all ASYSTD topological descriptions have one input and one output. In the case of the mono-node device, the input node serves only to satisfy the input syntax. The bi-node and multi-node distinction is even less obvious since the only requirement is that all input TAP connections to a multi-node device must be specified.

Several rules and assumptions make up the formulation of the algorithms for simulating a system or model, and impact on the user in the following ways:

1) Every user defined model must have an input node named "INPUT", and an output node named "OUTPUT".

2) At least one path must be defined which relates OUTPUT to INPUT.

3) All paths must terminate at OUTPUT or at a TAP.

4) The value of the signal at a node is the instantaneous sum of the OUTPUT of all devices connected to the node.

5) The value of a TAP is the instantaneous OUTPUT of the element to which it is attached (not the output node of the element).

6) A TAP may be referenced external to the model in which it is defined.

7) Model nodes are not externally available.

8) TAP's maintain the status of a unique variable name and thus may be used in FORTRAN expressions, subroutine calls, output requests, etc.

9) Node names must be legal FORTRAN variables, i.e., no more than six alphanumeric characters, the first of which must be a letter.

10) TAP names are defined as TAPXXX, where XXX is a unique number less than 1000. Sequential numbering is recommended.

11) All variable names and expressions must be FORTRAN compatible, including integer and floating point conventions. Names beginning with the letters I through N or the letter Z are typed as integers.

12) Intrinsic ASYSTD variables are defined in Table 2-1, and are available to the user.

13) The user is cautioned against using variables which begin with the letters V or Z, as a conflict with system variables is possible.

14) Maximum number of continuation cards is four.

15) A node may not connect directly to itself.

16) A model name may be up to 36 characters long.

17) The distinctions between a model and a system are the declaration and I/O identifiers. A system is executable; a model is callable.

18) A TAP is externally referenced from the element field by stating the model name and the tap number. The tap number is positive for an input, negative for an output. The RNF of the statement is the LNF of the model reference.

In addition to the above, certain procedures and requirements exist as to setting up the appropriate parameter values necessary to ensure an efficient, yet cost effective simulation. These include the following:

1) Sampling rate should be at least 15 to 20 times that of the highest frequency of interest — which is normally the widest bandwidth, although relaxation of this approximation is certainly desirable when possible. Accuracies of less than 1 percent are easily achieved for transfer function representations with lower sampling rates. One obvious problem, particularly for low order functions, is aliasing of the input signals. Thus care must

Table 2-1. Intrinsic ASYSTD Parameters

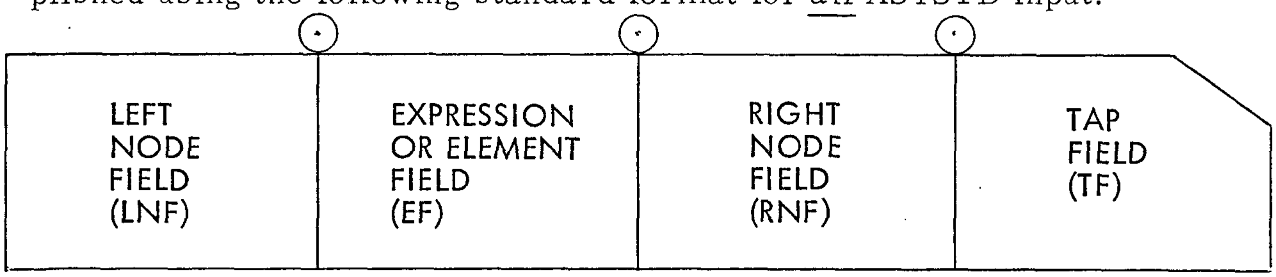
Variable	Usage
TIME (or T)	The time as kept by the Simulation Clock (unrelated to actual computer run time)
TSTART	The simulation start time (i. e., a time bias for output labeling)
TSTOP	Simulation stop time
SETTLE	Setting time before outputting
DT	Sample time
$	Used to denote the current signal at the input node
Z + 1	Absolute address of the first data cell available to the model (V(Z+1))
ZZ	Absolute address of the last data cell used by the model (V(ZZ))
V ( ) or VV ( )	Dynamic storage array
VIN	Address of the current real input
VIN + 1	Address of the current imaginary input
VOUT	Address of the current output
VOUT + 1	Address of the current imaginary output
PI	3. 14159
TWO PI	6. 28318
NPRINT	Output print interval

be taken, and in some cases, the SWAG technique is the only recourse.

2) All simulations require the specification of sampling time (DT) and stop time (TSTOP): Unless parameters are defined in a DATA or DEFAUL statement, ASYSTD will abort the simulation phase.

3) Turn on transients can be eliminated from the output by use of the intrinsic parameter SETTLE. A rule of thumb is that turn on transients should die out at approximately 5/BW, where BW is the approximate system bandwidth.

Once the user has applied the above rules in specifying the block diagram equivalent of his system, he may assign node names and tap names where appropriate, and proceed to encode the system accordingly. The input language is fully explained in the next section; its most important aspects are introduced here. The basic problem is to represent the two-dimensional block-diagram using a series of statements. This is accomplished using the following standard format for <u>all</u> ASYSTD input:



where ⊙ denotes <u>any</u> non-alphanumeric character delimiter (other than (7-8) and (0-8-2) punches) which serve as field separators. The Tap Field is not required.

A general element definition is exemplified by:

NODE1 < BLACK BOX > RIGHT 'TAP69

This statement says that model BLACK BOX processes the signal at NODE1 and its output is at node RIGHT. The tap TAP69 also contains the signal output by model BLACK BOX which may not be the signal at node RIGHT.

The standard card format is used for inputting the various ASYSTD identifiers and is given in Table 2-2. These identifiers are required to define whether the descriptions are for a system or model. All but DEFINE, SET, and END are illegal for a model description. The ASYSTD identifiers occupy the LNF with the appropriate data in the EF. Recall that the distinction between a model and a system is the input/output and declaration information.

An example of a model definition:

```
MODEL  =  EXAMPLE WITH TAPS, A, B
    INPUT  =  SIN ($)/TAP1  =  NODE1
    NODE1  =  DUMY MODEL (TAP2)  =  NODE2 'TAP1
    NODE2  =  $*$*A/B  =  OUTPUT
END
```

Although this example is physically meaningless, it serves to illustrate one use of taps and is explained thusly: The signal at NODE1 equals the sine of the input signal (denoted by $) divided by the signal output from model DUMY MODEL. (This follows from the definition of TAP1.) The signal at NODE2 is then equal to the output of DUMY MODEL, when NODE1 and TAP2 are its inputs. Note that TAP2 is an input to EXAMPLE WITH TAPS, since it was not defined in the model description. The output of EXAMPLE WITH TAPS, OUTPUT, is the signal at NODE2 squared, times A, divided by B. Here A and B are variables which must be specified by whatever model references the EXAMPLE WITH TAPS model. Also note that TAP1 is externally available for any other model reference.

Table 2-2. ASYSTD Identifiers

Identifier	Use
SYSTEM	Indicates that the deck following defines an ASYSTD system
MODEL – Name - arg 1, arg 2	Indicates that the deck which follows defines an ASYSTD model
END	Used to indicate the end of a model or system deck
DATA – Variable list	Indicates that the following values are to be read in at execution time by namelist (used only in a system) name ASYSTD
DEFAUL(T) – Variable list	Indicates the default values for DATA parameters not input through namelist
PRINT – Node or tap names	Indicates a list follows specifying output to the printer
PLOT – Node or tap names	Indicates a list follows specifying output to be plotted on the CalComp plotter
PPLOT – Node or tap names	Indicates a list follows specifying output to be plotted on the printer
POST – Routine name, node or tap names	Indicates that the following post-processing routine is to be called
PAGE	When present causes 8-1/2" x 11" compatible output
DEFINE – Variable - expression	Generates a FORTRAN define statement, which produces code to evaluate the expression
SET – Variable - expression	Generates a FORTRAN assignment statement, which gives the variable the value of the expression
VARY – Variable - parameters	Indicates the value of the variable is to be varied over a specified range

The external connection of taps between models is best illustrated in the following example:

```
MODEL - EXTERNAL TAP CONNECT, FC
INPUT < SINE (TWOPI*FC*TIME) > NODE1
    NODE1 < EXAMPLE WITH TAPS (1., 2.) > NODE2
    NODE2 < $*$ > NODE3
    NODE3 < EXAMPLE WITH TAPS +2 > NODE1
    NODE3 < 2.*$ > OUTPUT
    NODE4 < EXAMPLE WITH TAPS -1 > NODE1
    NODE4 < $ >DUMY 'TAP09
```

In this example the signal at NODE3 is connected to input tap number two of the model named EXAMPLE WITH TAPS (as denoted in the EF and RNF of the statement). In addition, tap one of the model was externally connected to NODE4 and given the external name of TAP09, which in turn makes it available to yet another level of models.

The instructions necessary to define a system are identical to those for a model, except for the addition of the declarations and Input/Output statements. These are fully described in the next section. The entire input for similar problems is described in Section 6.0.

Several advanced techniques are available to the user in modeling any system. The principal technique is modeling directly in FORTRAN (or other compatible language) and interfacing with ASYSTD generated models. The most likely necessity for such a requirement is in RF element modeling, in which case the signals become complex. ASYSTD maintains any complex quantities but does not directly perform complex arithmetic operations. All complex RF models are, or use, models written in FORTRAN. Such an undertaking is not recommended to the novice user.

As noted above, the user may modify or replace the ASYSTD library and directory at will. This task is accomplished within the computer system executive, utilizing the file manipulation and updating features. The library directory given earlier in Figure 2-1 contains the necessary information for modifying the directory. Since each model is a subroutine, any computer system file may be designated as a library to be searched, or the EXEC temporary program file can be loaded with the desired subroutines at allocation time.

In any event, the directory serves to:

1) Cross reference model names and subroutine entry points.

2) Flag the model as a subroutine or function.

3) Defines the number of arguments to the routine.

4) Defines the number and type of each tap.

5) Defines the use (presently ignored) of the model.

Whenever a model is processed by ASYSTD, an entry point name is assigned to it, and any subsequent models in the same run. The assignment is made by alphabetically incrementing the least significant character of the directory entry at Line 23, i.e., if Line 23 is MODEL MODEL, then the first temporary model would have entry MODELA, the second MODELB, etc. If Line 23 was MODELX, then the first model would be MODELY, the second MODELZ, the third MODEMA, etc. In addition, the appropriate entries are temporarily made in the directory for defining the model for the particular run only. An example is given in Figure 2-2 where the directory entry is the first line of output. If it were desirable to add the model to the permanent library, one of two paths is available:

1) Rename the entry point by recompiling and updating the FORTRAN Subroutine and entering the appropriate data into the directory.

2) Adding the ASYSTD generated directory card into the directory and changing Line 23 to reflect the "highest" model name in the library.

Certainly, the latter is much easier, but may lead to future confusion.

```
NRZ                                                        MODELA   1  1  0    000000000000 2
ASYSTD PROCESSOR LEVEL II
VERSION DATED 01 NOV 71 FOR THE MSC U1108 SYSTEM
THIS DECK PROCESSED ON 28 APR 72 AT 19:17:05


SYSTID MODELS REFERENCED                                  ENTRY POINT

      SQ                                                     SQ

THIS MODEL ASSIGNED THE ENTRY POINT NAME MODELA



000001                MODEL=NRZ,BR
000002                      INPUT < SQ(BR/2.) > N1
000003                      N1 < $*.5+.5 > OUTPUT
000004                END
```

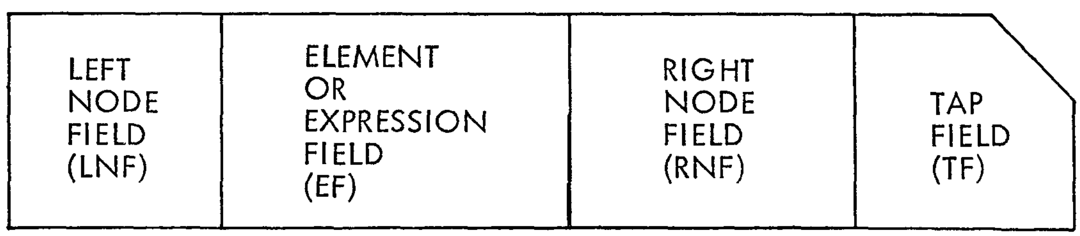
Figure 2-2. ASYSTD Phase 1 Output for Model NRZ

SECTION 3.0

INPUT LANGUAGE

The ASYSTD input language consists of descriptive statements constructed largely from user defined names and specifications. Several key variables are used, as defined earlier in Table 2-2, in addition to names which reference defined library models. The statements for the most part define some input and output node, along with an element specification or expression relating the two. The statements can be punched in a completely free-field data card (Columns 1 through 80). Since all statements are scanned from both the left and right, field delimiters can be <u>any</u> non-alphanumeric character other than (7-8) and (0-2-8) punches. A statement may have up to four continuation cards, which are defined by a non-alphanumeric in Column 1. With the exception of the title on a SYSTEM card, all blanks are ignored by the processor.

As discussed in Section 2.0, the standard card format is utilized for all ASYSTD input, including the identifiers given in Table 2-2. These identifiers primarily concern themselves with initializing and controlling the execution of a system simulation. The identifier occupies the LNF and any data occupies the EF of the standard card format defined as follows:



The various identifiers are discussed below:

DATA: var_1, var_2, ..., var_n

This identifier names a list of variables which can be read in at simulation time under namelist SYSTID. Whenever this statement appears, a namelist data card must appear at execution time. The number of variables is limited only by the number one can squeeze on 5 cards. The variables must conform to FORTRAN requirements, and may be any intrinsic ASYSTD variable, e.g.,

DATA = TSTOP, DT, NPRINT, MEDIA, SAM, A

The Phase II input data for this example would be, for example:

$SYSTID DT = 1.5 E-6, TSTOP = 10 ... $

which is standard FORTRAN namelist input.

DEFAUL: var_1 = const, var_2 = const, ..., var_n = const

This identifier serves to load default values for any variable in the simulation, including any intrinsic ASYSTD variable. The constant values must conform to the FORTRAN rules of integer and Floating point variables or errors may occur. The number of entries is limited by the number one can squeeze on 5 cards, e.g.,

DEFAUL: DT = 1.5 E-6, TSTOP = 2.0, A = 10.,

: IOU = 3

DEFINE: variable = FORTRAN expression

This identifier generates a FORTRAN 'DEFINE' statement. 'Variable' must be a legal FORTRAN name. Whenever this name appears in the generated FORTRAN program, the FORTRAN expression is calculated using current values of any variables which may appear in the expression, and this result is used for the value of 'variable'.

END: comment

Signifies the end of a model or system description.

MODEL: model external name, arg_1, arg_2, ..., arg_n

This identifier instructs ASYSTD to expect a model definition only (i.e., topological information) and to generate a subroutine.

The external name must be no more than 36 characters, with no more than 99 arguments, e.g.,

MODEL: FM MODULATOR, BETA, FC

or

MODEL: UHF RECEIVER

PAGE: comment

This identifier, merely by being present, causes all printed output to be 8-1/2 by 11 inch compatible.

PLOT: $name_1$, $name_2$, ..., $name_n$

This identifier is similar to PPLOT, the exception being that here the CalComp (or SC4060) plotter is utilized, e.g.,

PLOT: TAP69, X, NODE4, OUTPUT

POST: subroutine name, $name_1$, $name_2$, ..., $name_n$

This identifier causes a post-processing routine named "subroutine name" to be called following the simulation. This routine is called for each occurrence of $name_i$, which may be nodes, taps, or variables. The call sequence generated is similar to that of the plot request, the only difference being the entry point. Utility routines are available for interfacing a user post-processor with the data time histories which are stored on drum.

PPLOT: $name_1$, $name_2$, ..., $name_n$

This identifier defines the data to be plotted vs. TIME following the simulation. The quantities may be nodes, taps, and variables, e.g.,

PPLOT: NODE1, TAP69, B

PRINT: $name_1$, $name_2$, $name_3$, ..., $name_n$

This identifier defines the data to be printed during the simulation. The quantities which may be printed consist of nodes, taps, and variables. Note that TIME is automatically printed – it need not be requested, e.g.,

PRINT: NODE1, TAP69, AVAR

SET: variable = FORTRAN expression

This identifier generates a FORTRAN assignment statement which sets 'variable' equal to the value of 'FORTRAN expression'.

SYSTEM: title

This identifier instructs ASYSTD to expect other identifiers dealing with input/output, etc., and to generate a main program. The title serves to label all output (36 characters), e.g.,

SYSTEM: TRY ME SOMETIME

VARY: variable = min, max, delta

This statement generates a FORTRAN DO-LOOP which has within its range any VARY or SET statements which follow it in the ASYSTD deck as well as the program simulation. If the 'variable' name is type integer (first letter is I through N or Z), the parameters are expected to be of integer type, otherwise the variable name and parameters can be real or integer. The parameters (min, max, delta) must be constants, DEFINE variable names, or variable names which have appeared in a DATA or DEFAULT statement.

Any card which has an empty LNF (a non-alphanumeric in Column 2 or later as the first non-blank character) is treated as a comment. A comment may appear anywhere except the middle of a continued statement. Comments may be continued.

Topological input data fully utilizes the standard card format, that is:

> Input node $\odot$ expression or element $\odot$ output node $\odot$ Tap assignment

where $\odot$ is any valid delimiter as defined above. For example,

```
NODE1 = RF PHASE MODULATOR (BETA, FC) = NODE2
      = TAP8
```

represents a normal statement utilizing the four fields of a standard input form.

Thus, with one format, ASYSTD input data is extremely straightforward and easily mastered.

Table 3-1 shows which statements are allowed and what order they must be placed in Systems and Models respectively.

Table 3-1. ASYSTD Identifier Hierarchy

SYSTEM	MODEL
DATA	DEFINE
DEFAUL	SET
PAGE	{TOPOLOGICAL
PLOT	INFORMATION}
POST	END
PPLOT	
PRINT	
DEFINE	
VARY	
SET	
{TOPOLOGICAL	
INFORMATION}	
END	

The first card of a System is the SYSTEM card. This is followed by DATA, DEFAUL, ..., PRINT statements as appropriate in any order. Any DEFINE statements come next followed by any VARY or SET statements which are required. Topological information cards describing the System come next in any order, followed by the END card.

A MODEL card defines a Model. Any DEFINES followed by any SET statements appear next. The topological information is again followed by an END card.

SECTION 4.0

PROBLEM SUBMISSION

4.1 PROGRAM RUN PREPARATIONS

When executing the first phase of ASYSTD, the procedure PROCS/SYSTID and the element LIBARY are required along with the absolute element for the first phase (SYSTID). The LIBARY element is the model library dictionary.

The second phase requires only the library file containing all the ASYSTD library relocatable elements and the elements output to the PCF by the first phase (MAIN𝑏̸𝑏̸/SYSTID).

4.1.1 Deck Setup

The technique used by ASYSTD is to output the processed models to the PCF as elements named MODELA/SYSTID, MODELB/SYSTID, etc., in the same order as processed. When a system is processed, its element name is MAIN𝑏̸𝑏̸/SYSTID. Therefore, the user at MSC must provide the FORTRAN control card for each of the elements. The models generated in the following example are "temporary" models for this run only. That is, any system description can reference model EXAMPLE 1 during this run only. Models are permanent when the relocatable is available and an entry is made in the LIBARY element used by the first pass. Figure 4-1 is the run deck setup for the MSC UNIVAC 1108 system.

```
@ XQT CUR

---- LOAD THE PCF WITH SYSTID ABSOLUTE, PROCS/SYSTID, AND LIBARY

@ XQT SYSTID                                    )
MODEL: EXAMPLE ONE                              |
       .                                        |  MODELA/SYSTID
       .                                        |  is generated in PCF
       .                                        |
       .                                        |
END                                             )
MODEL: EXAMPLE TWO                              )
       .                                        |
       .                                        |  MODELB/SYSTID
       .                                        |
       .                                        |
END                                             )
MODEL: EXAMPLE THREE                            )
       .                                        |
       .                                        |  MODELC/SYSTID
       .                                        |
       .                                        |
END                                             )
SYSTEM. USE EXAMPLES ONE,TWO,THREE              )
       .                                        |
       .                                        |  MAINbb/SYSTID
       .                                        |
       .                                        |
END                                             )
@I FOR,* MODELA/SYSTID                          )
@I FOR,* MODELB/SYSTID                          |  User supplied
@I FOR,* MODELC/SYSTID                          |  compiler cards
@I FOR,* MAIN  /SYSTID                          )
@ XQT MAIN  /SYSTID                             }  Execute the second pass
 $SYSTID   [NAMELIST I/O, IF SPECIFIED IN SIMULATION]
```

Figure 4-1. Run Deck Setup for MSC 1108 Exec II System

4.1.2 Required I/O Devices

The ASYSTD program contains two phases. The first phase requires the Logical Unit assignments as follows:

Logical Unit	Device
5	Card Reader
6	Line Printer
1	Scratch Drum

Phase two requires the following I/O devices:

Logical Unit	Device
5	Card Reader
6	Line Printer
13	Drum
14	Drum

4.2 OUTPUT DESCRIPTION

4.2.1 Data Output

ASYSTD output consists of the first phase processing and the simulation or second phase output. The first phase provides output similar to the FORTRAN V compiler. Figure 4-2 is an example of the first phase output for a particular model. Annotations on the output fully explain their significance.

```
FIRSTIFAMP                                   MODELA   1  3  2   400000000000 2
ASYSTD PROCESSOR LEVEL II
VERSION DATED 01 NOV 71 FOR THE MSC U1108 SYSTEM
THIS DECK PROCESSED ON 07 AUG 72 AT 14:25:41
```

} LIBARY card for permanent entry

```
SYSTID MODELS REFERENCED                         ENTRY POINT

      COMPLEXMULTIPLIER                            CMULT
      FILTER                                       FILTER

THIS MODEL ASSIGNED THE ENTRY POINT NAME MODELA

TAP ORDER:   1 TAP001   2 TAP002
```

```
            MODEL=FIRST IF AMP, GAIN,SLOPE,IFUNC
                  INPUT < COMPLEX MULTIPLIER(*,TAP2) > N1
                  N1 < FILTER(2,IFUNC,3,0.,6.E6,60.E6,1.,0.,0.) > N2
                  N2 < FILTER(2,IFUNC,3,0.,6.E6,60.E6,1.,0.,0.) > N3
                  N3 < FILTER(2,IFUNC,3,0.,6.E6,60.E6,1.,0.,0.) > OUTPUT
                  INPUT < 10.**((GAIN+TAP1*SLOPE)/20.) > D1 ' TAP2
            END
```

Figure 4-2. ASYSTD First Phase Output

4.2.2 Optional Output

The second phase output is completely optional under user control. The two forms of output are printed and graphical presentations, whose size is selectable as 8-1/2 by 11 inches or 11 by 14 inches (see Section 3.1). The current version provides printer plot (PTPLT), with the entry point TMPLT reserved for CalComp or SC4020 graphical routines. Examples of the output are contained in Section 6.0.

SECTION 5.0

FORTRAN INTERFACING

There are two instances in which the user may wish to write his own FORTRAN subroutines to interface with the main ASYSTD simulation. These are Models which, for some reason, are not easily or efficiently written in the ASYSTD language and post-processing routines.

5.1 FORTRAN LIBRARY MODEL PROCEDURES

An important feature of ASYSTD is its ability to reference any FORTRAN subprogram (model). However, in order for a user's FORTRAN subprogram to be re-entrant and interface correctly with any other model, it must conform to certain procedures. The characteristics of such a subprogram and then usage should be:

- Input arguments are never altered.
- Every reference to the subprogram is unique (independent).
- All local storage is considered scratch; permanent storage is only available in a common storage pool named "V".
- If a subroutine-primary input is at V(VIN) and V(VIN + 1), primary output goes to V(VOUT) and V(VOUT + 1).
- Numeric values of VIN and VOUT should be restored prior to exit.

To properly interface with any routine processed by ASYSTD, the following procedures should be followed, if applicable:

- A FORTRAN procedure named HEDFOR is available to provide compatible definition of intrinsic ASYSTD storage and variables. Use of the procedure is the FORTRAN V statement:

  INCLUDE HEDFOR, LIST

  When writing FORTRAN models, a current listing of this procedure is sometimes useful.

- Defining <u>local</u> variable Z as the last cell in the "V" pool used by the calling subroutine, and <u>common</u> variable ZZ the last cell to be used by this routine, set:

  Z = ZZ

  and

  ZZ = ZZ + n

  where n is the number of locations required for permanent storage in the "V" pool.

  A reservation of storage space for a particular reference to this model is made, reserving cells (V(Z + 1) through V(Z + n) for storage.

- Local variable Z is used to address the V array in the model. Common variable ZZ should not be utilized in any statement other than above, unless no other models are referenced.

- When referencing other models, VIN and VOUT should be assigned to one of the reserved cells in the "V" pool.

If a user written model is to be referenced, the user must update the ASYSTD library (element LIBRARY). This task is performed under the system executive, external to ASYSTD. See Pages 2-2 and 2-13 for instructions on making a library entry.

5.2 FORTRAN POST-PROCESSING ROUTINES

Linkage to any user post-processing routine is available through use of the ASYSTD identifier "POST". When a reference to post is made in any system description, a subroutine call is made to the user's program for every specified variable in the form:

```
CALL NAME (LABEL, NPAGE)
```

where

LABEL is the hollerith name of the variable.

NPAGE conveys the output sizing, i.e., NPAGE = 4 is 8.5 x 11 compatible output, NPAGE = 7 is standard computer size output.

For example:

The ASYSTD statements

```
POST Ø SPECTM Ø NODE1, OUTPUT
```

would generate the following two lines in the main simulation program:

```
CALL SPECTM ('NODE1', 7)

CALL SPECTM ('OUTPUT', 7)
```

The first line of the user written routine SPECTM would generally be as follows:

```
SUBROUTINE SPECTM (LABEL, NPAGE)
```

Two input parameters are required, even if the user is not concerned with output sizing.

Whenever a POST statement appears in an ASYSTD program, MAIN/SYSTID contains the following common block (all variables which begin with Z are integers):

```
COMMON/DRMHED/ZDATE(2), ZTOD(2), TSTART,
TSTOP, VEQDT, SETTLE, ZEDIT, ZNPU, ZRESRV,
ZUSED, ZNAMES, ZNAME (< ZNAMES >)
```

where

ZDATE	=	BCD of DATE, left adjusted, blank filled
ZTOD	=	BCD of TIME OF DAY, left adjusted, blank filled
TSTOP, TSTART, VEQDT, SETTLE		Self explanatory, VEQDT = DT (which appears in /COGENT/)
ZEDIT	=	Edit interval (if all values won't fit on drum)
ZNPU	=	Number of variables saved on each logical unit
ZRESRV	=	Number of words of storage reserved for each variable to be saved

ZUSED = Number of values for each variable stored on drum (in some routines named NVAL)

ZNAMES = Number of different variables stored on drum

ZNAME(I), I = 1, < ZNAMES > = names of variables stored on drum

Inclusion of this common block in a post-processing routine allows the user access to several important variables, particularly NVAL (ZUSED), the number of values available for any particular variable stored on drum.

All retrieval of information stored on drum should be accomplished with the two subroutines DRMGET and DRMEDT.

<u>DRMGET:</u> Calling sequence:

```
CALL DRMGET (NAME, ARRAY, LENGTH, MSKIP)
```

where

NAME contains name of variable referenced (BCD).

ARRAY is array into which values are to be read (dimensioned at least by LENGTH).

LENGTH is number of values to be read.

MSKIP is number of words to skip before reading.

Continuing the example, the following lines of code retrieve all values stored on drum for a particular variable, 1000 values at a time.

Example 1:

```
SUBROUTINE SPECTM (LABEL, NPAGE)
COMMON/DRMHED/DUMMY(11), NVAL
DIMENSION ARRAY (1000)
    .
    .
    .
DO 100 MSKIP = 0, NVAL, 1000
CALL DRMGET (LABEL, ARRAY, 1000, MSKIP)
    .
    .
    .
          Analysis of the 1000 values
          of the variable named in LABEL
    .
    .
    .
100 CONTINUE
```

DRMEDT: Calling sequence:

```
CALL DRMEDT (NAME, NREQ, ARRAY)
```

where

NAME contains name of variable referenced (BCD).

NREQ equals number of values to be read off drum.

ARRAY is array into which values are placed (dimensioned at least by NREQ)

Let NVAL be the twelfth word of common block /DRMHED/. NVAL equals the number of values stored on drum for each variable. If NREQ ≥ NVAL, then NVAL words (all that are on the drum) will be read into ARRAY. If NREQ < NVAL, the data on drum is edited and NREQ equally spaced values are read into ARRAY.

Example 2:

```
SUBROUTINE SPECTM (LABEL, NPAGE)
DIMENSION ARRAY (1000)
    .
    .
    .
CALL DRMEDT (LABEL, 1000, ARRAY)
```

ARRAY now contains 1000 equally spaced values of the variable named in LABEL.

Values of TIME are not stored on drum but are computed by DRMGET and DRMEDT when needed, saving drum I/O time.

In Example 1, assume the array TIME is dimensioned by 1000. Then the statement:

```
CALL DRMGET ('TIME', TIME, 1000, MSKIP)
```

placed inside the DO LOOP will compute the 1000 values of time corresponding to the values in ARRAY.

Similarly, in Example 2, the statement:

```
CALL DRMEDT ('TIME', 1000, TIME)
```

would produce 1000 values of time in the array TIME such that ARRAY(I) was the value of the variable named in LABEL at time TIME(I).

SECTION 6.0

EXAMPLES

Four example sets are presented in this section, namely:

1) ASYSTD simulation of an Apollo PCM/PM/PM communications link whose characteristics are given in Figure 6-1.

2) ASYSTD squaring loop model also defined in Figure 6-1.

3) ASYSTD model of sonar propagation through sea water. This example serves to illustrate evaluation of a mathematical function with ASYSTD.

4) ASYSTD model of FILTER response. This example serves to illustrate the DEFINE, VARY, and SET commands.

The first example consists of a temporary definition of model NRZ and use of several library models, along with some math expressions. Figure 6-2 illustrates the ASYSTD output for the run. The sequence numbers to the left of the input statements correspond to the data card number. Thus, model NRZ was defined with the first four cards of the input data deck. Following the two pages of Phase I output are the FORTRAN compilations of the resultant subroutine and main program. These two routines, along with the ASYSTD library make up the Phase II simulation program. The results of the simulation are evident following the FORTRAN routines.

A few remarks concerning this simulation are in order. Notice that following the square wave phase modulator is an RF phase modulator. At this point, we are representing the RF portion of the link at baseband.

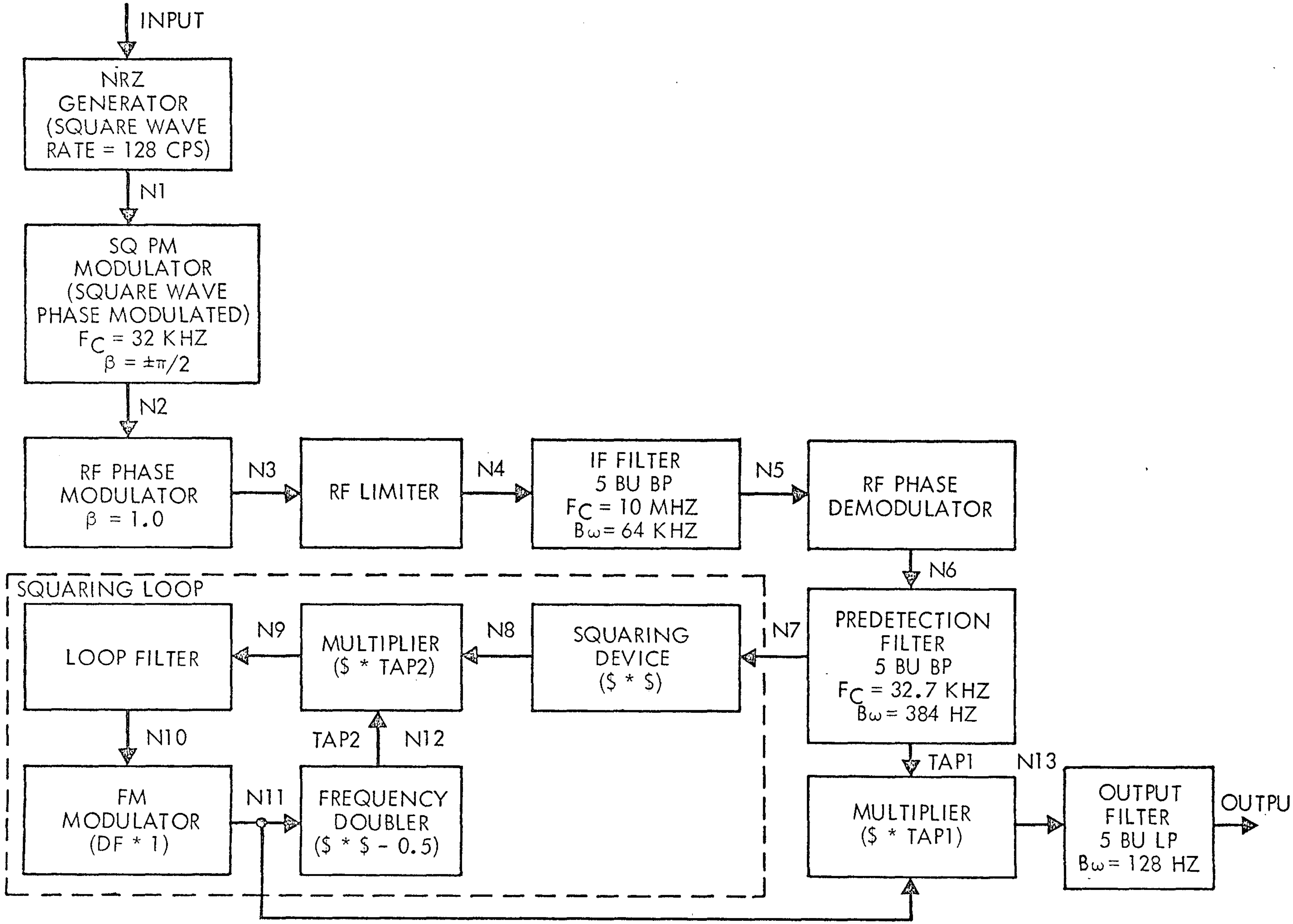


Figure 6-1. Apollo PCM/PM/PM Link Block Diagram

```
NRZ                                                          MODELA    1  1  0    000000000000 2
ASYSTD PROCESSOR LEVEL II
VERSION DATED 01 NOV 71 FOR THE MSC U1108 SYSTEM
THIS DECK PROCESSED ON 01 MAY 72 AT 16:10:25


SYSTID MODELS REFERENCED                                          ENTRY POINT

      SQ                                                             SQ


THIS MODEL ASSIGNED THE ENTRY POINT NAME MODELA

000001              MODEL=NRZ,BR
000002                    INPUT < SQ(BR/2.) > N1
000003                    N1 < $*.5+.5 > OUTPUT
000004              END

APOLLO PCM/PM/PM LINK
ASYSTD PROCESSOR LEVEL II
VERSION DATED 01 NOV 71 FOR THE MSC U1108 SYSTEM
THIS DECK PROCESSED ON 01 MAY 72 AT 16:10:26

SYSTID MODELS REFERENCED                                          ENTRY POINT

      NRZ                                                            MODELA
      SQPMOD                                                         SQPMOD
      RFPHASEMODULATOR                                               RPMMOD
      RFLIMITER                                                      RFLIMT
      BUTTERWORTH                                                    BUTWTH
      RFPHASEDEMODULATOR                                             RFPDEM
      LOOPFILTER                                                     LEDLAG
      FMMODULATOR                                                    FMMOD

TAP ORDER:    1 TAP001    2 TAP002

000005              SYSTEM .  APOLLO PCM/PM/PM LINK
000006              PAGE,  SMALL
000007              DEFAUL. TSTART=0.,TSTOP=.03,DT=1.5E-6,NPRINT=200
000008              DATA = DT,TSTOP,SETTLE
000009              PRINT.N1,N10,N13,OUTPUT
000010              PPLOT.N1,N10,N13,OUTPUT
```

Figure 6-2. ASYSTD Example, Output

```
                  INPUT < NRZ(128.) > N1
                  N1 < SQPMOD(PI,32.768E3) > N2
                  N2 < RF PHASE MODULATOR (1.0) > N3
                  N3 < RF LIMITER > N4
                  N4 < BUTTERWORTH (5,3,10.E6,64.E3,10.E6,1.) > N5
                  N5 < RF PHASE DEMODULATOR (1.0) > N6
                  N6 < BUTTERWORTH (5,3,32.768E3,384.,0.,1.) > N7 'TAP1
                  N7 < $*$ > N8
                  N8 < $*TAP2 > N9
                  N9 < LOOP FILTER (200.,1.8775994,0.,.19751719,0.) > N10
                  N10 < FM MODULATOR (2.*PI,32.768E3) > N11
                  N11 < $*$-0.5 > N12 'TAP2
                  N11 < $*TAP1 > N13
                  N13 < BUTTERWORTH (2,1,0.,128.,0.,1.) > OUTPUT
             END

@ ADD ADDFIL

@I FOR,* MODELA/SYSTID,MODELA/SYSTID
FORTRAN V:  ISD VERSION 2.2
THIS COMPILATION WAS DONE ON 01 MAY 72 AT 16:10:27


   SUBROUTINE MODELA     ENTRY POINT 000065

   STORAGE USED (BLOCK, NAME, LENGTH)

         0001    *CODE     000077
         0000    *DATA     000025
         0002    *BLANK    000000
         0003    COGENT    000010
         0004    VSPACE    000003
         0005    FLR       000014


   EXTERNAL REFERENCES (BLOCK, NAME)

         0006    SQ
         0007    NERR3$
```

Figure 6-2. ASYSTD Example, Output (Cont)

```
STORAGE ASSIGNMENT FOR VARIABLES (BLOCK, TYPE, RELATIVE LOCATION, NAME)

 0001     000044 150L        0001     000003 20L         0003 R 000002 DT        0003 R 000005 PI        0003     000003 T
 0003 R 000003 TIME          0003 R 000007 TWOPI         0004 R 000001 V         0003 I 000006 VCIN      0005 R 000002 VF
 0003 I 000000 VIN           0000 I 000001 VNSAVE        0003 I 000001 VOUT      0000 I 000000 VSAVE     0004 R 000000 VSUB0
 0000 I 000002 Z             0005 I 000001 ZNUMF         0003 I 000004 ZZ        0005 I 000000 Z1STF


00101      1*           SUBROUTINE  MODELA(BR)
00101      2*     CMODEL        NRZ
00103      3*           INCLUDE MODEL1,LIST
00104      3*           IMPLICIT INTEGER (Z)
00105      3*           INTEGER VIN,VOUT,VCIN,VSAVE,VNSAVE
00106      3*           COMMON /COGENT/VIN,VOUT,DT,TIME,ZZ,PI,VCIN,TWOPI
00107      3*           COMMON /VSPACE/ VSUB0,V(2)
00110      3*           COMMON /FLR/ Z1STF,ZNUMF,VF(5,2)
00111      3*           EQUIVALENCE (TIME,T)
00112      3*           END
00113      4*           INCLUDE MODEL2,LIST
00114      4*           Z=ZZ
00115      4*           GO TO 150
00116      4*        20 VNSAVE=VIN
00117      4*           VSAVE=VOUT
00120      4*           END
00121      5*           V(Z+1)=V(VIN)
00121      6*     C     SQ
00122      7*           VIN=Z+1
00123      8*           VOUT=Z+3
00124      9*           CALL  SQ(BR/2.)
00125     10*           V(Z+5)=V(Z+3)*.5+.5
00126     11*           VIN=VNSAVE
00127     12*           VOUT=VSAVE
00130     13*           V(VSAVE)=V(Z+5)


00131     14*           RETURN
00132     15*       150 ZZ=ZZ+      6
00133     16*           GO TO 20
00134     17*           END
```

Figure 6-2. ASYSTD Example, Output (Cont)

```
@I FOR,* MAIN  /SYSTID,MAIN  /SYSTID                                                    01 MAY 72 * 16:10:28.756
FORTRAN V:  ISD VERSION 2.2
THIS COMPILATION WAS DONE ON 01 MAY 72 AT 16:10:28

   MAIN PROGRAM

   STORAGE USED (BLOCK, NAME, LENGTH)

         0001    *CODE    000550
         0000    *DATA    002131
         0002    *BLANK   000000
         0003    COGENT   000010
         0004    VSPACE   023421
         0005    FLR      030326
         0006    NAME     000006
         0007    INT      000001
         0010    DIF      000001
         0011    DRMHED   000021


   EXTERNAL REFERENCES (BLOCK, NAME)

         0012    DATIN
         0013    PAGER
         0014    MODELA
         0015    SQPMOD
         0016    RPMMOD
         0017    RFLIMT
         0020    BUTWTH
         0021    RFPDEM
         0022    LEDLAG
         0023    FMMOD
         0024    DRUMIT
         0025    PTPLT
         0026    NRNL$
         0027    NWNL$
         0030    NWDU$
         0031    NI02$
         0032    NSTOP$
         0033    NI01$


   STORAGE ASSIGNMENT FOR VARIABLES (BLOCK, TYPE, RELATIVE LOCATION, NAME)

    0001     000113 10L        0001     000461 100L       0000     002057 120F       0001     000513 130L       0001     000541 150L
    0001     000125 20L        0001     000544 200L       0001     000066 221G       0001     000072 226G       0001     000072 231G
    0001     000367 30L        0001     000445 363G       0000     002030 4F         0001     000404 40L        0001     000476 401G
    0001     000503 405G       0001     000033 5L         0000     002052 50F        0000     002056 60F        0003 R 000002 DT
    0000 I 000001 I$           0000 I 000000 NPRINT      0003 R 000005 PI          0011 R 000007 SETTLE      0003     000003 T
    0003 R 000003 TIME         0006 R 000000 TITLE       0011 R 000004 TSTART      0011 R 000005 TSTOP       0003 R 000007 TWOPI
    0004 R 000001 V            0003 I 000006 VCIN        0000 R 000033 VDRUM       0007 R 000000 VDT2        0011 R 000006 VEQDT
    0005 R 000002 VF           0003 I 000000 VIN         0003 I 000001 VOUT        0000 R 000002 VPRINT      0000 R 002011 VSETTL
```

Figure 6-2. ASYSTD Example, Output (Cont)

```
0004 R 000000 VSUB0     0010 R 000000 VTDT      0000 I 002012 Z        0000 I 002005 ZCOUNT    0011 I 000000 ZDATE
0000 I 002004 ZDRUM     0000 I 002003 ZEDCNT    0011 I 000010 ZEDIT    0000 I 002013 ZIOT1     0000 I 002014 ZIOT2
0011 I 000015 ZNAME     0011 I 000014 ZNAMES    0011 I 000011 ZNPU     0005 I 000001 ZNUMF     0000 I 002006 ZPRINT
0011 I 000012 ZRESRV    0011 I 000002 ZTOD      0011 I 000013 ZUSED    0003 I 000004 ZZ        0000 I 002007 Z1
0005 I 000000 Z1STF     0000 I 002010 Z2
```

```
00101     1*        INCLUDE MAIN1,LIST
00103     1*        IMPLICIT INTEGER (Z)
00104     1*        INTEGER VIN,VOUT,VCIN
00105     1*        PARAMETER VSIZE=10000,VFSIZE=2500
00106     1*        COMMON /COGENT/VIN,VOUT,DT,TIME,ZZ,PI,VCIN,TWOPI
00107     1*        COMMON /VSPACE/VSUB0,V(VSIZE)
00110     1*        COMMON /FLR/Z1STF,ZNUMF,VF(5,VFSIZE)
00111     1*        COMMON /NAME/TITLE(6)
00112     1*        COMMON /INT/VDT2   /DIF/VTDT
00113     1*        EQUIVALENCE (T,TIME)
00114     1*        DATA PI/3.1415927/, TWOPI/6.2831853/
00117     1*        DATA TSTART,SETTLE /0.,0./, NPRINT/1/
00123     1*        END
00124     2*        DATA TITLE/'APOLLO PCM/PM/PM LINK                 '/
00126     3*        DATA  TSTART/0./,TSTOP/.03/,DT/1.5E-6/,NPRINT/200/
00133     4*        NAMELIST/SYSTID/DT,TSTOP,SETTLE
00134     5*        PARAMETER ZPSIZE= 5
00135     6*        DIMENSION VPRINT(5,ZPSIZE)
00136     7*        DATA VPRINT(1, 1)/'TIME  '/
00140     8*        DATA VPRINT(1, 2)/'N1    '/
00142     9*        DATA VPRINT(1, 3)/'N10   '/
00144    10*        DATA VPRINT(1, 4)/'N13   '/
00146    11*        DATA VPRINT(1, 5)/'OUTPUT'/
00150    12*        PARAMETER ZDRMSZ= 250
00151    13*        DIMENSION VDRUM(ZDRMSZ, 4)
00152    14*        COMMON /DRMHED/ZDATE(2),ZTOD(2),TSTART,TSTOP,VEQDT,SETTLE,
00152    15*       .ZEDIT,ZNPU,ZRESRV,ZUSED,ZNAMES,ZNAME( 4)
00153    16*        DATA ZNPU/ 2/,  ZRESRV/131063/,  ZUSED/ 0/,  ZNAMES/ 4/
00160    17*        DATA ZNAME( 1)/'N1    '/
00162    18*        DATA ZNAME( 2)/'N10   '/
00164    19*        DATA ZNAME( 3)/'N13   '/
00166    20*        DATA ZNAME( 4)/'OUTPUT'/
00170    21*        READ(5,SYSTID)
00173    22*        WRITE(6,SYSTID)
00176    23*        CALL DATIN(ZDATE)
00177    24*        INCLUDE MAIN2,LIST
00200    24*        IF((TSTOP-TSTART)/DT.GT.0) GO TO 5
00202    24*        WRITE(6,4)TSTART,TSTOP,DT
00207    24*      4 FORMAT(1X,'** ERROR ** THE VALUES TSTART=',E12.6,' TSTOP=',E12.6,
00207    24*       .  ' DT=',E12.6,' ARE UNREASONABLE.')
00210    24*        STOP
00211    24*      5 ZEDIT=(TSTOP-(TSTART+SETTLE))/(DT*ZRESRV)+1
00212    24*        ZEDCNT=ZEDIT-1
00213    24*        ZDRUM=0
```

Figure 6-2. ASYSTD Example, Output (Cont)

```
00214    24*         ZUSED=0
00215    24*         ZCOUNT=NPRINT-1
00216    24*         ZPRINT=1
00217    24*         VSUB0=0.0
00220    24*         DO 6 Z1=1,VSIZE
00223    24*       6 V(Z1)=0.0
00225    24*         DO 7 Z2=1,VFSIZE
00230    24*         DO 7 Z1=1,5
00233    24*       7 VF(Z1,Z2)=0.0
00236    24*         VEQDT=DT
00237    24*         VDT2=DT/2.0
00240    24*         VTDT=2.0/DT
00241    24*         TIME=TSTART-DT
00242    24*         VSETTL=ISTART+SETTLE
00243    24*         CALL PAGER
00244    24*         @.......................................................................
00244    24*         @  THE SIMULATION LOOP BEGINS HERE.  GOOD LUCK.
00244    24*         @.......................................................................
00244    24*      10 TIME=TIME+DT
00245    24*         IF(TIME.GT.TSTOP) GO TO 100
00247    24*         Z=2
00250    24*         GO TO 150
00251    24*      20 Z1STF=1
00252    24*         VIN=1
00253    24*         VOUT=2
00254    24*         END
00255    25*         ZIOT1=Z+1
00256    26*         ZIOT2=Z+2
00256    27*   C     NRZ
00257    28*         VIN=Z+4
00260    29*         VOUT=Z+6
00261    30*         CALL  MODELA(128.)
00261    31*   C     SQPMOD
00262    32*         VIN=Z+6
00263    33*         VOUT=Z+8
00264    34*         CALL  SQPMOD(PI,32.768E3)
00264    35*   C     RFPHASEMODULATOR
00265    36*         VIN=Z+8
00266    37*         VOUT=Z+10
00267    38*         CALL  RPMMOD(1.0)
00267    39*   C     RFLIMITER
00270    40*         VIN=Z+10
00271    41*         VOUT=Z+12
00272    42*         CALL  RFLIMT
00272    43*   C     BUTTERWORTH
00273    44*         VIN=Z+12
00274    45*         VOUT=Z+14
00275    46*         CALL  BUTWTH(5,3,10.E6,64.E3,10.E6,1.)
00275    47*   C     RFPHASEDEMODULATOR
00276    48*         VIN=Z+14
00277    49*         VOUT=Z+16
```

Figure 6-2. ASYSTD Example, Output (Cont)

```
00300    50*          CALL  RFPDEM(1.0)
00300    51*    C     BUTTERWORTH
00301    52*          VIN=Z+16
00302    53*          VOUT=Z+18
00303    54*          CALL  BUTWTH(5,3,32.768E3,384.,0.,1.)
00304    55*          V(ZIOT1)=V(Z+18)
00305    56*          V(Z+20)=V(Z+18)*V(Z+18)
00306    57*          V(Z+22)=V(Z+20)*V(ZIOT2)
00306    58*    C     LOOPFILTER
00307    59*          VIN=Z+22
00310    60*          VOUT=Z+24
00311    61*          CALL  LEDLAG(200.,1.8775994,0.,.19751719,0.)
00311    62*    C     FMMODULATOR
00312    63*          VIN=Z+24
00313    64*          VOUT=Z+26
00314    65*          CALL  FMMOD(2.*PI,32.768E3)
00315    66*          V(Z+28)=V(Z+26)*V(ZIOT1)
00316    67*          V(Z+30)=V(Z+26)*V(Z+26)-0.5
00317    68*          V(ZIOT2)=V(Z+30)
00317    69*    C     BUTTERWORTH
00320    70*          VIN=Z+28
00321    71*          VOUT=Z+32
00322    72*          CALL  BUTWTH(2,1,0.,128.,0.,1.)
00323    73*          IF(TIME.LT.VSETTL) GO TO 10
00325    74*          INCLUDE MAIN3,LIST
00326    74*          ZEDCNT=ZEDCNT+1
00327    74*          IF(ZEDCNT.NE.ZEDIT) GO TO 40
00331    74*          ZEDCNT=0
00332    74*          IF(ZDRUM.NE.ZDRMSZ) GO TO 30
00334    74*          CALL DRUMIT(VDRUM,ZDRMSZ,ZDRMSZ)
00335    74*          ZDRUM=0
00336    74*       30 ZDRUM=ZDRUM+1
00337    74*          END
00340    75*          VDRUM(ZDRUM,1)=V(Z+6)
00341    76*          VDRUM(ZDRUM,2)=V(Z+24)
00342    77*          VDRUM(ZDRUM,3)=V(Z+28)
00343    78*          VDRUM(ZDRUM,4)=V(Z+32)
00344    79*       40 CONTINUE
00345    80*          ZCOUNT=ZCOUNT+1
00346    81*          IF(ZCOUNT.NE.NPRINT) GO TO 10
00350    82*          ZCOUNT=0
00351    83*          ZPRINT=ZPRINT+1
00352    84*          VPRINT(ZPRINT,1)=TIME
00353    85*          VPRINT(ZPRINT,2)=V(Z+6)
00354    86*          VPRINT(ZPRINT,3)=V(Z+24)
00355    87*          VPRINT(ZPRINT,4)=V(Z+28)
00356    88*          VPRINT(ZPRINT,5)=V(Z+32)
00357    89*          IF(ZPRINT.NE.5) GO TO 10
00361    90*          WRITE(6,50)VPRINT
00367    91*       50 FORMAT( 5(6X,A6,4(4X,E12.6),/))
```

Figure 6-2. ASYSTD Example, Output (Cont)

```
00370     92*         INCLUDE MAIN5,LIST
00371     92*         WRITE(6,60)
00373     92*      60 FORMAT(//)
00374     92*         ZPRINT=1
00375     92*         GO TO 10
00376     92*         @-------------------------------------------------------------
00376     92*         @ END OF SIMULATION AND OUTPUT LOOP
00376     92*         @-------------------------------------------------------------
00376     92*     100 IF(ZPRINT.LT.2) GO TO 130
00400     92*         DO 110 Z2=1,ZPSIZE
00403     92*     110 WRITE(6,120)(VPRINT(Z1,Z2),Z1=1,ZPRINT)
00412     92*     120 FORMAT(6X,A6,8(4X,E12.6))
00413     92*     130 CONTINUE
00414     92*         END
00415     93*         CALL DRUMIT(VDRUM,ZDRMSZ,ZDRUM)
00416     94*         CALL PTPLT('N1',4)
00417     95*         CALL PTPLT('N10',4)
00420     96*         CALL PTPLT('N13',4)
00421     97*         CALL PTPLT('OUTPUT',4)
00422     98*         GO TO 200
00423     99*     150 ZZ=   35
00424    100*         GO TO 20
00425    101*     200 CONTINUE
00426    102*         STOP
00427    103*         END
```

```
$SYSTID
DT       =     .15000000E-05,
TSTOP    =     .30000000E-01,
SETTLE   =     .00000000E+00,
$END
```

APOLLO PCM/PM/PM LINK

TIME	.00000	.300000-03	.599999-03	.899998-03
N1	.500000+00	.100000+01	.100000+01	.100000+01
N10	.000000	.656525-08	-.746476-07	-.152746-03
N13	.000000	.227434-04	.175139-03	.245243-03
OUTPUT	.000000	.156697-07	.168748-05	.243908-04

TIME	.120000-02	.150000-02	.180000-02	.209999-02
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	.35/258-02	-.559070-04	-.526487-01	.262237+00
N13	.169327-01	.374304-02	.961061-02	.146836+00
OUTPUT	.152122-03	.597630-03	.174886-02	.417225-02

Figure 6-2. ASYSTD Example, Output (Cont)

TIME	.239999-02	.269999-02	.299998-02	.329998-02
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	.235829-03	-.732689+00	.207953+01	-.479202-02
N13	.213897-02	.571740-01	.419319+00	-.285553-01
OUTPUT	.856548-02	.156568-01	.260872-01	.402898-01

TIME	.359998-02	.389997-02	.419996-02	.449995-02
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	-.264826+01	.551619+01	-.715591-01	-.448266+01
N13	.147583+00	.691304+00	-.865632-01	.235775+00
OUTPUT	.583719-01	.800794-01	.104793+00	.131583+00

TIME	.479995-02	.509994-02	.539993-02	.569992-02
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	.753886+01	-.237658+00	-.462787+01	.683724+01
N13	.816727+00	-.143214+00	.287642+00	.786661+00
OUTPUT	.159302+00	.186727+00	.212670+00	.236107+00

TIME	.599991-02	.629990-02	.659989-02	.689988-02
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	-.431207+00	-.347788+01	.512809+01	-.616192+00
N13	-.176832+00	.310868+00	.691535+00	-.191894+00
OUTPUT	.256240+00	.272514+00	.284704+00	.292891+00

TIME	.719987-02	.749986-02	.779985-02	.809983-02
N1	.100000+01	.100000+01	.100000+01	.000000
N10	-.225543+01	.387623+01	-.871415+00	-.128855+01
N13	.330832+00	.612792+00	-.205552+00	.369509+00
OUTPUT	.297365+00	.298637+00	.297303+00	.294000+00

TIME	.839981-02	.869979-02	.899977-02	.929975-02
N1	.000000	.000000	.000000	.000000
N10	.318014+01	-.128211+01	-.251971+00	.215747+01
N13	.569352+00	-.220764+00	.411885+00	.483117+00
OUTPUT	.289434+00	.284182+00	.278610+00	.272790+00

Figure 6-2. ASYSTD Example, Output (Cont)

TIME	.959973-02	.989971-02	.101997-01	.104997-01
N1	.000000	.000000	.000000	.000000
N10	-.121934+01	.411117+00	.434535+00	-.990002-01
N13	-.190814+00	.317573+00	.217844+00	-.547896-01
OUTPUT	.266355+00	.258530+00	.248216+00	.234191+00

TIME	.107996-01	.110996-01	.113996-01	.116996-01
N1	.000000	.000000	.000000	.000000
N10	.420782-01	.190610+00	-.956791+00	.220183+01
N13	-.229144-01	-.142686+00	.126381+00	-.471236+00
OUTPUT	.215278+00	.190604+00	.159713+00	.122710+00

TIME	.119996-01	.122995-01	.125995-01	.128995-01
N1	.000000	.000000	.000000	.000000
N10	.980072+00	-.490457+01	.648092+01	.955244+00
N13	-.376862+00	.243143+00	-.764494+00	-.396353+00
OUTPUT	.803356-01	.339293-01	-.147052-01	-.635299-01

TIME	.131995-01	.134995-01	.137994-01	.140994-01
N1	.000000	.000000	.000000	.000000
N10	-.743435+01	.788194+01	.445719+00	-.684126+01
N13	.256648+00	-.812628+00	-.286450+00	.202163+00
OUTPUT	-.110463+00	-.153620+00	-.191486+00	-.223013+00

TIME	.143994-01	.146994-01	.149993-01	.152993-01
N1	.000000	.000000	.000000	.000000
N10	.662391+01	.151791+00	-.558136+01	.554226+01
N13	-.726501+00	-.167211+00	.139113+00	-.655248+00
OUTPUT	-.247648+00	-.265355+00	-.276509+00	-.281881+00

TIME	.155993-01	.158993-01	.161993-01	.164992-01
N1	.000000	.100000+01	.100000+01	.100000+01
N10	.668839-01	-.534075+01	.568418+01	.496007-01
N13	-.885654-01	.911298-01	-.660766+00	-.349722-01
OUTPUT	-.262513+00	-.279539+00	-.274107+00	-.267287+00

Figure 6-2. ASYSTD Example, Output (Cont)

TIME	.167992-01	.170992-01	.173992-01	.176992-01
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	-.552587+01	.529758+01	.488094-01	-.279278+01
N13	.451700-01	-.639167+00	.133613-01	-.399623-02
OUTPUT	-.259906+00	-.252391+00	-.244711+00	-.236339+00

TIME	.179991-01	.182991-01	.185991-01	.188991-01
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	.154222+01	.483110-01	.480901-01	.558543+00
N13	-.343496+00	.173536-01	.874546-03	.203078+00
OUTPUT	-.226334+00	-.213517+00	-.196680+00	-.174882+00

TIME	.191991-01	.194990-01	.197990-01	.200990-01
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	.296511-01	-.333966+01	.556423+01	-.197780+00
N13	-.608075-01	.991263-01	.678739+00	-.175912+00
OUTPUT	-.147563+00	-.114669+00	-.767387-01	-.348599-01

TIME	.203990-01	.206989-01	.209989-01	.212989-01
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	-.723729+01	.830803+01	-.658058+00	-.617685+01
N13	.229905+00	.846334+00	-.251546+00	.314120+00
OUTPUT	.942134-02	.543021-01	.978994-01	.138434+00

TIME	.215989-01	.218989-01	.221988-01	.224988-01
N1	.100000+01	.100000+01	.100000+01	.100000+01
N10	.626401+01	-.101773+01	-.324278+01	.360482+01
N13	.751280+00	-.264644+00	.340925+00	.584649+00
OUTPUT	.174419+00	.204787+00	.228915+00	.246658+00

TIME	.227988-01	.230988-01	.233988-01	.236987-01
N1	.100000+01	.100000+01	.100000+01	.000000
N10	-.124278+01	-.130019+01	.227149+01	-.164206+01
N13	-.250826+00	.358504+00	.478412+00	-.247866+00
OUTPUT	.258292+00	.264429+00	.265969+00	.263979+00

Figure 6-2. ASYSTD Example, Output (Cont)

TIME	.239987-01	.242987-01	.245987-01	.248987-01
N1	.000000	.000000	.000000	.000000
N10	-.747024-01	.174500+01	-.223840+01	.101721+01
N13	.410315+00	.435458+00	-.248915+00	.445291+00
OUTPUT	.259541+00	.253673+00	.247193+00	.240563+00

TIME	.251986-01	.254986-01	.257986-01	.260986-01
N1	.000000	.000000	.000000	.000000
N10	.954893+00	-.146940+01	.646723+00	.110304+00
N13	.334148+00	-.174473+00	.269222+00	.922981-01
OUTPUT	.233784+00	.226352+00	.217359+00	.205662+00

TIME	.263986-01	.266985-01	.269985-01	.272985-01
N1	.000000	.000000	.000000	.000000
N10	.487551-01	.321839+00	.202539+00	-.280043+01
N13	.610647-03	-.162654+00	-.157077+00	.169523+00
OUTPUT	.190084+00	.169686+00	.143897+00	.112620+00

TIME	.275985-01	.278984-01	.281984-01	.284984-01
N1	.000000	.000000	.000000	.000000
N10	.447980+01	.389647+00	-.808026+01	.898008+01
N13	-.617993+00	-.257488+00	.231121+00	-.849890+00
OUTPUT	.763427-01	.360805-01	-.671068-02	-.502888-01

TIME	.287984-01	.290984-01	.293983-01	.296983-01
N1	.000000	.000000	.000000	.000000
N10	.214964+00	-.948450+01	.889698+01	.735933-01
N13	-.205760+00	.190190+00	-.832553+00	-.100174+00
OUTPUT	-.928312-01	-.132600+00	-.168103+00	-.198254+00

TIME	.299983-01
N1	.000000
N10	-.756394+01
N13	.113971+00
OUTPUT	-.222388+00

Figure 6-2. ASYSTD Example, Output (Cont)

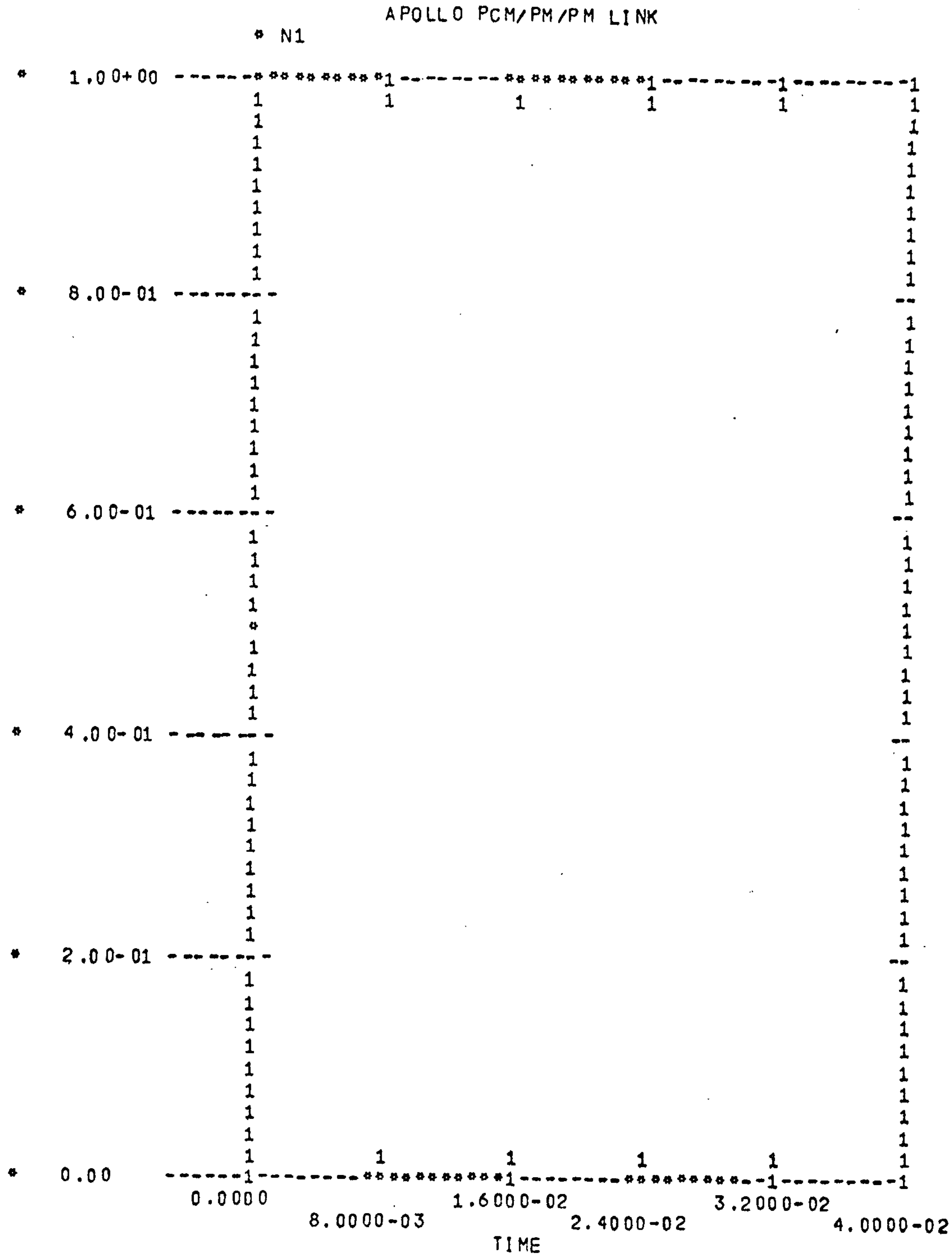


Figure 6-2. ASYSTD Example, Output (Cont)

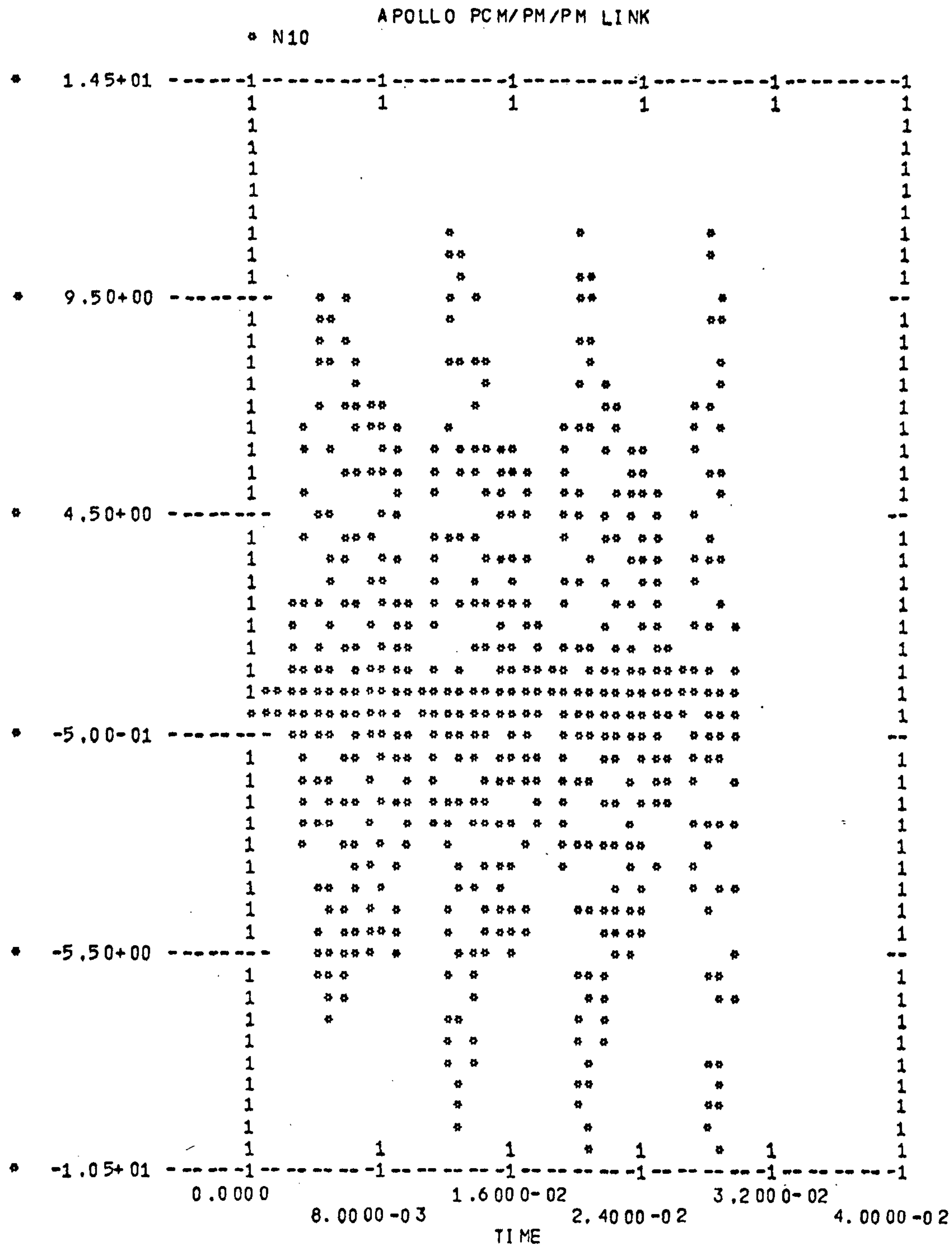


Figure 6-2. ASYSTD Example, Output (Cont)

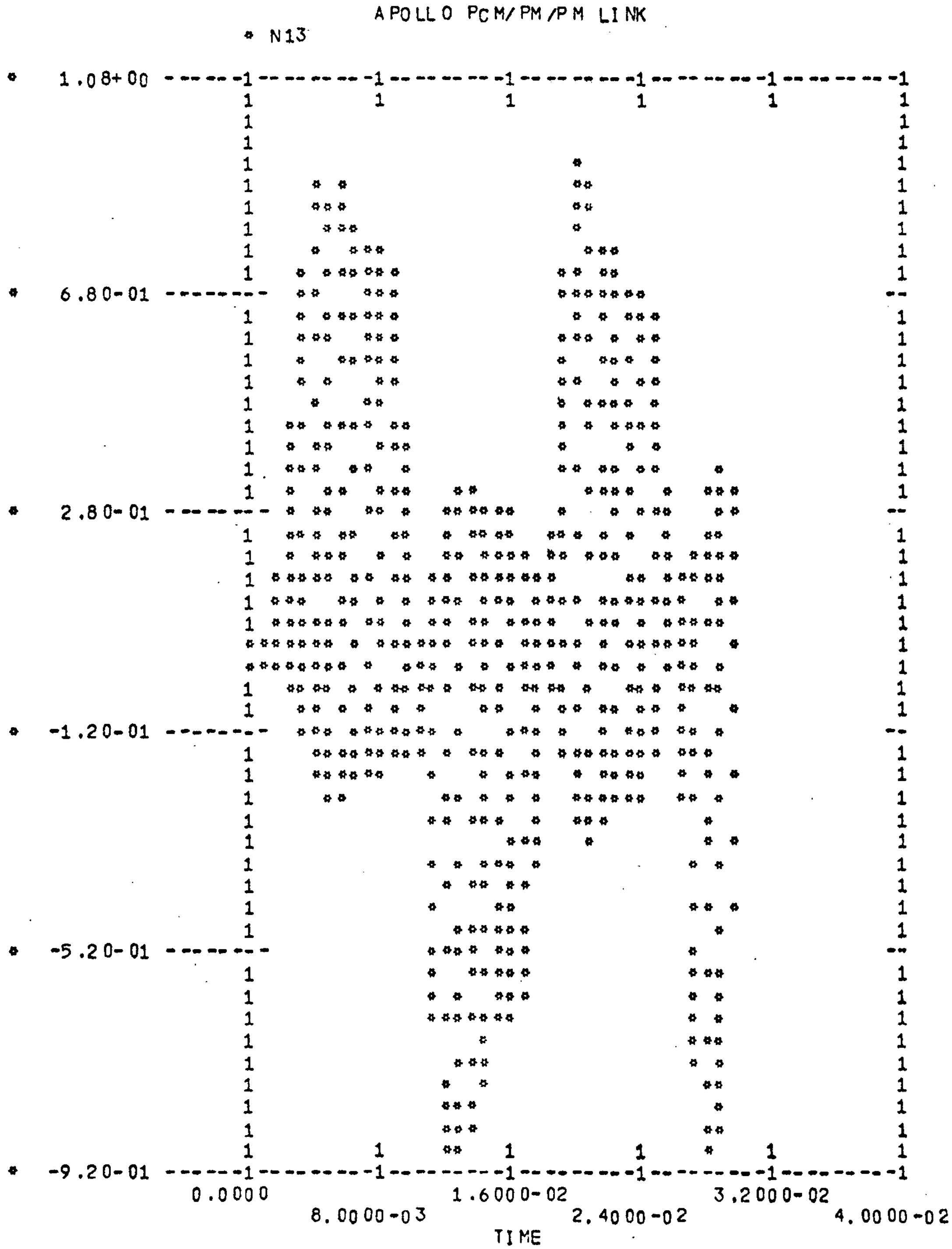


Figure 6-2. ASYSTD Example, Output (Cont)

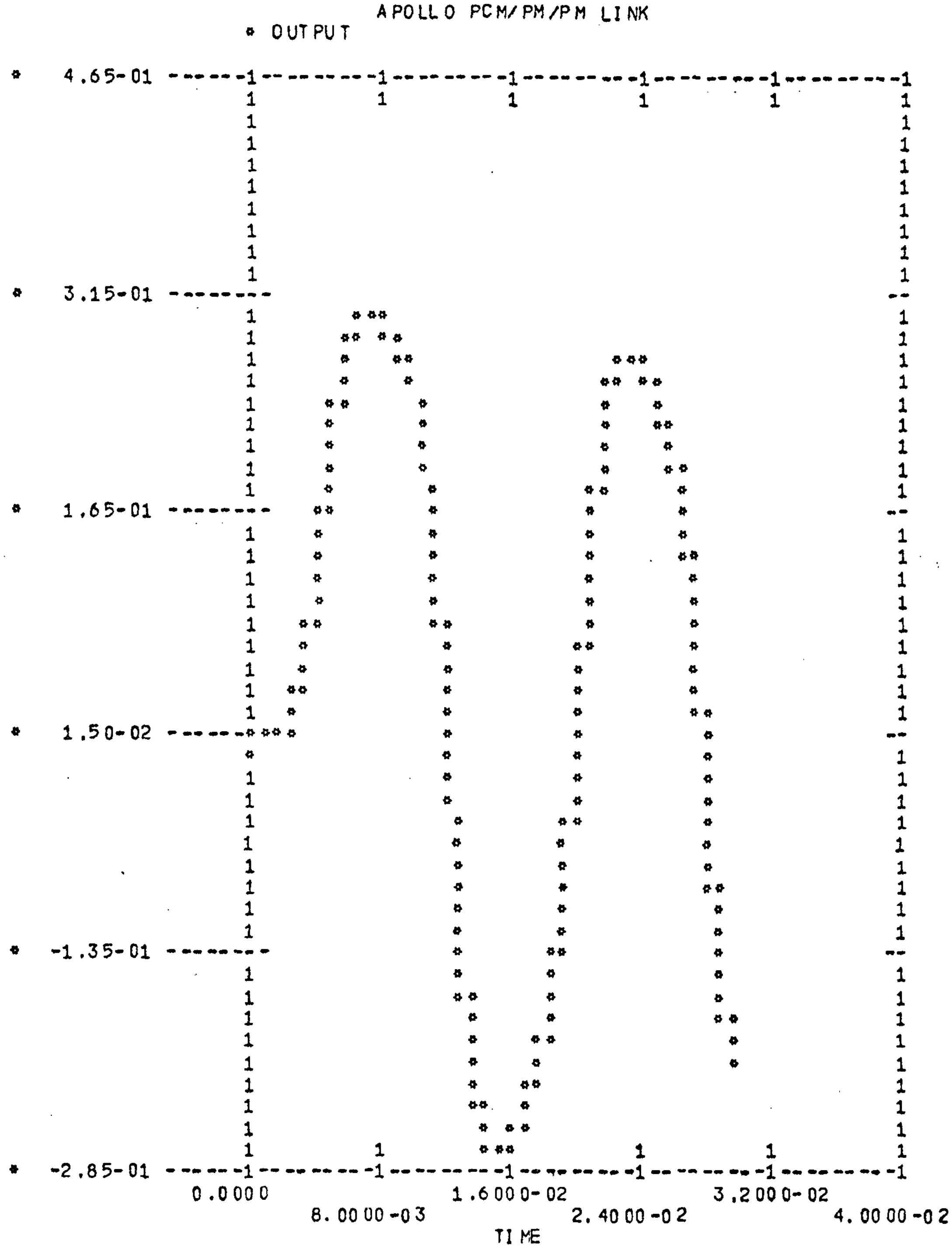


Figure 6-2. ASYSTD Example, Output (Cont)

This process is actually implemented by assuming that the input signals are analytic. This condition is met if the baseband signal spectrum is essentially zero at the carrier frequency. If this assumption is not true, a ripple in the simulation output at frequencies of approximately twice the carrier is introduced — the case if the input is a step function, for instance. In any event, the ripple is normally negligible due to the large ratio between baseband and carrier.

The second example is to illustrate the definition of a model, namely a squaring loop. Squaring loops have been utilized for deriving subcarrier phase references for detection of phase modulated signals. The topology is shown in Figure 6-1 since the elements making up the proposed model were used in the Apollo link simulation. The model utilizes a single tap for use by the multiplier. Figure 6-3 presents the Phase I ASYSTD output and resulting FORTRAN subroutine.

The third example is a model representing a sonar system propagation function. The mathematical form of the model is given by:

$$a = \frac{A f_m f^2}{f_m^2 + f^2} + B f^2 \text{ dB/meter}$$

where

f_m = Relaxation frequency (kHz) given in table below

f = Operating frequency (kHz)

A, B are curve fit coefficients for salt water given by the following

Temperature	f_m	A	B
5°C	60 kHz	6×10^{-4}	3.2×10^{-7}
15°C	100 kHz	6×10^{-4}	2.4×10^{-7}

```
SQUARINGLOOP                                     MODELA    1   0   ▮     000000000000 2
ASYSTD PROCESSOR LEVEL II
VERSION DATED 01 NOV 71 FOR THE MSC U1108 SYSTEM
THIS DECK PROCESSED ON 01 MAY 72 AT 14:52:49


SYSTID MODELS REFERENCED                              ENTRY POINT

      LOOPFILTER                                        LEDLAG
      FMMODULATOR                                       FMMOD


THIS MODEL ASSIGNED THE ENTRY POINT NAME MODELA


TAP ORDER:    1 TAP002



                  MODEL=SQUARING LOOP
                        INPUT < $*$ > N8
                        N8 < $*TAP2 > N9
                        N9 < LOOP FILTER (200.,1.8775994,0.,.19751719,0.) > N10
                        N10 < FM MODULATOR (2.*PI,32.768E3) > OUTPUT
                        OUTPUT < $*$-0.5 > N12 ' TAP2
                  END
@ ADD ADDFIL
@I FOR,* MODELA/SYSTID,MODELA/SYSTID                                                 01 MAY 72 * 14:52:49.756
FORTRAN V:   ISD VERSION 2.2
THIS COMPILATION WAS DONE ON 01 MAY 72 AT 14:52:49

   SUBROUTINE MODELA      ENTRY POINT 000116

   STORAGE USED (BLOCK, NAME, LENGTH)

         0001    *CODE     000127
         0000    *DATA     000034
         0002    *BLANK    000000
         0003    COGENI    000010
         0004    VSPACE    000003
         0005    FLR       000014


   EXTERNAL REFERENCES (BLOCK, NAME)

         0006    LEDLAG
         0007    FMMOD
         0010    NERR3$
```

Figure 6-3. ASYSTD Example 2

```
STORAGE ASSIGNMENT FOR VARIABLES (BLOCK, TYPE, RELATIVE LOCATION, NAME)

 0001     000074 150L      0001     000003 20L       0003 R 000002 DT      0003 R 000005 PI      0003   000003 T
 0003 R 000003 TIME        0003 R 000007 TWOPI       0004 R 000001 V       0003 I 000006 VCIN    0005 R 000002 VF
 0003 I 000000 VIN         0000 I 000001 VNSAVE      0003 I 000001 VOUT    0000 I 000000 VSAVE   0004 R 000000 VSUB0
 0000 I 000002 Z           0000 I 000003 ZIOT1       0005 I 000001 ZNUMF   0003 I 000004 ZZ      0005 I 000000 Z1STF


00101      1*        SUBROUTINE MODELA
00101      2*  CMODEL         SQUARINGLOOP
00103      3*        INCLUDE MODEL1,LIST
00104      3*        IMPLICIT INTEGER (Z)
00105      3*        INTEGER VIN,VOUT,VCIN,VSAVE,VNSAVE
00106      3*        COMMON /COGENT/VIN,VOUT,DT,TIME,ZZ,PI,VCIN,TWOPI
00107      3*        COMMON /VSPACE/ VSUB0,V(2)
00110      3*        COMMON /FLR/ Z1STF,ZNUMF,VF(5,2)
00111      3*        EQUIVALENCE (TIME,T)
00112      3*        END
00113      4*        INCLUDE MODEL2,LIST
00114      4*        Z=ZZ
00115      4*        GO TO 150
00116      4*     20 VNSAVE=VIN
00117      4*        VSAVE=VOUT
00120      4*        END
00121      5*        ZIOT1=Z+1
00122      6*        V(Z+2)=V(VIN)
00123      7*        V(Z+4)=V(Z+2)*V(Z+2)
00124      8*        V(Z+6)=V(Z+4)*V(ZIOT1)
00124      9*  C     LOOPFILTER
00125     10*        VIN=Z+6
00126     11*        VOUT=Z+8
00127     12*        CALL  LEDLAG(200.,1.8775994,0.,.19751719,0.)

00127     13*  C     FMMODULATOR
00130     14*        VIN=Z+8
00131     15*        VOUT=Z+10
00132     16*        CALL  FMMOD(2.*PI,32.768E3)
00133     17*        V(Z+12)=V(Z+10)*V(Z+10)-0.5
00134     18*        V(ZIOT1)=V(Z+12)
00135     19*        VIN=VNSAVE
00136     20*        VOUT=VSAVE
00137     21*        V(VSAVE)=V(Z+10)
00140     22*        RETURN
00141     23*    150 ZZ=ZZ+    13
00142     24*        GO TO 20
00143     25*        END
```

Figure 6-3. ASYSTD Example 2 (Cont)

Figure 6-4 is the ASYSTD output for both the model and test run which evaluates the function over a given frequency range. Note that the intrinsic variable TIME is used to represent frequency.

In the fourth example, a filter is simulated, and its response to an input signal measured for various bandwidths. The system variables DT and TSTOP are adjusted appropriately for each bandwidth (BW) considered.

The output generated by the first and last passes of this simulation are included following the FORTRAN program as shown in Figure 6-5.

```
SONARABSORPTION                                   MODELA    1   5   0    000000000000 2
ASYSTD PROCESSOR LEVEL II
VERSION DATED 01 NOV 71 FOR THE MSC U1108 SYSTEM
THIS DECK PROCESSED ON 01 MAY 72 AT 14:57:50


THIS MODEL ASSIGNED THE ENTRY POINT NAME MODELA


TAP ORDER:    1 TAP001    2 TAP002



                 MODEL = SONAR ABSORPTION,A,B,FM,FR,R
                       INPUT < $/10.**(R*TAP2/10.) > OUTPUT
                       INPUT < A*TAP1*FM/(TAP1+FM*FM)+B*TAP1 > N2 ' TAP2
                       INPUT < FR*FR > N3 ' TAP1
                 END


SIMULATE ABSORPTION VS. FREQ AT 1 ME
ASYSTD PROCESSOR LEVEL II
VERSION DATED 01 NOV 71 FOR THE MSC U1108 SYSTEM
THIS DECK PROCESSED ON 01 MAY 72 AT 14:57:50


SYSTID MODELS REFERENCED                               ENTRY POINT

     SONARABSORPTION                                     MODELA
```

Figure 6-4. ASYSTD Example 3 Output

```
SYSTEM. SIMULATE ABSORPTION VS. FREQ AT 1 METER
DEFAUL. TSTART=2., TSTOP=1000., DT=2., NPRINT=25
PAGE  . SMALL FOR PUBS
DATA  . A,B,FM
PRINT. RATIO,OUTPUT
PPLOT. OUTPUT
       INPUT < 1.0 > DRIVE
       DRIVE < SONAR ABSORPTION(A,B,FM,TIME,1.0) > RATIO
       RATIO < 10.*ALOG10($) > OUTPUT
END
```

```
@ ADD ADDFIL
```

```
$SYSTID
A         =     .60000000E-03,
B         =     .32000000E-06,
FM        =     .60000000E+02,
$END
```

Figure 6-4. ASYSTD Example 3 Output (Cont)

SIMULATE ABSORPTION VS, FREQ AT 1 ME

TIME	.200000+01	.520000+02	.102000+03	.152000+03
RATIO	.999991+00	.996252+00	.993099+00	.991165+00
OUTPUT	-.412234-04	-.163069-01	-.300748-01	-.385401-01

TIME	.202000+03	.252000+03	.302000+03	.352000+03
RATIO	.989432+00	.987554+00	.985413+00	.982962+00
OUTPUT	-.461386-01	-.543899-01	-.638183-01	-.746328-01

TIME	.402000+03	.452000+03	.502000+03	.552000+03
RATIO	.980183+00	.977068+00	.973613+00	.969821+00
OUTPUT	-.869288-01	-.100754+00	-.116134+00	-.133085+00

TIME	.602000+03	.652000+03	.702000+03	.752000+03
RATIO	.965692+00	.961229+00	.956437+00	.951320+00
OUTPUT	-.151615+00	-.171731+00	-.193436+00	-.216734+00

TIME	.802000+03	.852000+03	.902000+03	.952000+03
RATIO	.945883+00	.940132+00	.934072+00	.927711+00
OUTPUT	-.241625+00	-.268112+00	-.296195+00	-.325875+00

Figure 6-4. ASYSTD Example 3 Output (Cont)

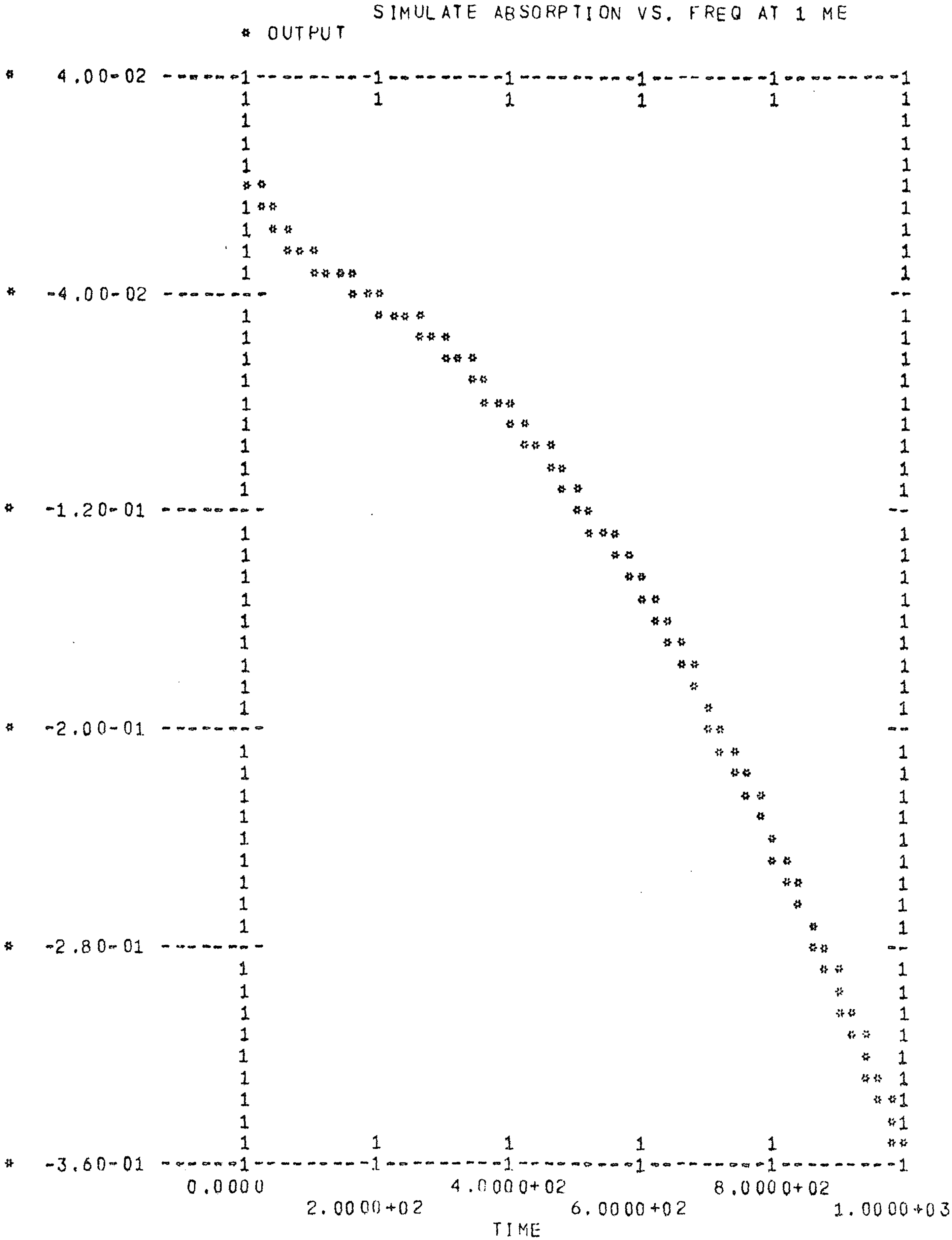


Figure 6-4. ASYSTD Example 3 Output (Cont)

```
TEST THE VARY/DEFINE FEATURE
ASYSTD PROCESSOR LEVEL II
VERSION DATED 01 NOV 71 FOR THE MSC U1108 SYSTEM
THIS DECK PROCESSED ON 21 DEC 71 AT 15:51:34


SYSTID MODELS REFERENCED                              ENTRY POINT

     BUTTERWORTH                                        BUTWTH


000001 ** WARNING ** AN OUTPUT LIST CONTAINS AN ITEM WHICH IS NOT A NODE, A TAP, OR TIME,
000001              SYSTEM,  TEST THE VARY/DEFINE FEATURE
000002              PRINT,  OUTPUT,TESTER
000003              PAGE,     SMALL
000004              PPLOT,  OUTPUT
000005              DEFINE,   TESTER=1.+(BW-10.)/20
000006              SET:  NPRINT=10
000007              VARY,   BW:10,:110,:20,
000008              SET:  DT=.05/BW
000009              SET:  TSTOP=5/BW
000010                    INPUT< 1,0 >N1
000011                    N1< BUTTERWORTH(5,1,0.,BW,0.,1.) >OUTPUT
000012              END
 @ ADD ADDFIL


@I FOR,* MAIN  /SYSTID,MAIN  /SYSTID                                       21 DEC 71 * 15:51:35,497
UNIVAC 1108 FORTRAN V LEVEL   2206 0018 F5018S
THIS COMPILATION WAS DONE ON 21 DEC 71 AT 15:51:35


  MAIN PROGRAM

  STORAGE USED (BLOCK, NAME, LENGTH)

        0001    *CODE     000353
        0000    *DATA     002072
        0002    *BLANK    000000
        0003    COGENT    000010
        0004    VSPACE    023421
        0005    FLR       030326
        0006    NAME      000006
        0007    INT       000001
        0010    DIF       000001
        0011    DRMHED    000016
```

Figure 6-5. Vary/Define Feature Example

```
EXTERNAL REFERENCES (BLOCK, NAME)

        0012    DATIN
        0013    PAGER
        0014    BUTWTH
        0015    DRUMIT
        0016    PTPLT
        0017    NWDU$
        0020    NI02$
        0021    NSTOP$
        0022    NI01$


STORAGE ASSIGNMENT FOR VARIABLES (BLOCK, TYPE, RELATIVE LOCATION, NAME)

 0001   000125 10L      0001   000276 100L     0000   002033 120F     0001   000330 130L     0001   000342 150L
 0001   000011 155G     0001   000137 20L      0001   000345 200L     0001   000100 204G     0001   000104 211G
 0001   000104 214G     0001   000262 277G     0001   000213 30L      0001   000313 315G     0001   000320 321G
 0000   002004 4F       0001   000222 40L      0001   000045 5L       0000   002026 50F      0000   002032 60F
 0000 R 001771 BW       0003 R 000002 DT       0000 I 000001 I$       0000 I 000000 NPRINT   0003 R 000005 PI
 0011 R 000007 SETTLE   0003   000003 T        0003 R 000003 TIME     0006 R 000000 TITLE    0011 R 000004 TSTART
 0011 R 000005 TSTOP    0003 R 000007 TWOPI    0004 R 000001 V        0003 I 000006 VCIN     0000 R 000021 VDRUM
 0007 R 000000 VDT2     0011 R 000006 VEQDT    0005 R 000002 VF       0003 I 000000 VIN      0003 I 000001 VOUT
 0000 R 000002 VPRINT   0000 R 002002 VSETTL   0004 R 000000 VSUB0    0010 R 000000 VTDT     0000 I 002003 Z
 0000 I 001776 ZCOUNT   0011 I 000000 ZDATE    0000 I 001773 ZDL1     0000 I 001772 ZDL2     0000 I 001775 ZDRUM
 0000 I 001774 ZEDCNT   0011 I 000010 ZEDIT    0011 I 000015 ZNAME    0011 I 000014 ZNAMES   0011 I 000011 ZNPU
 0005 I 000001 ZNUMF    0000 I 001777 ZPRINT   0011 I 000012 ZRESRV   0011 I 000002 ZTOD     0011 I 000013 ZUSED
 0003 I 000004 ZZ       0000 I 002000 Z1       0005 I 000000 Z1STF    0000 I 002001 Z2


00101      1*          INCLUDE MAIN1,LIST
00103      1*          IMPLICIT INTEGER (Z)
00104      1*          INTEGER VIN,VOUT,VCIN
00105      1*          PARAMETER VSIZE=10000,VFSIZE=2500
00106      1*          COMMON /COGENT/VIN,VOUT,DT,TIME,ZZ,PI,VCIN,TWOPI

00107      1*          COMMON /VSPACE/VSUB0,V(VSIZE)
00110      1*          COMMON /FLR/Z1STF,ZNUMF,VF(5,VFSIZE)
00111      1*          COMMON /NAME/TITLE(6)
00112      1*          COMMON /INT/VDT2    /DIF/VTDT
00113      1*          EQUIVALENCE (T,TIME)
00114      1*          DATA PI/3.1415927/, TWOPI/6.2831853/
00117      1*          DATA TSTART,SETTLE /0.,0./, NPRINT/1/
00123      1*          END
00124      2*          DATA TITLE/'TEST THE VARY/DEFINE FEATURE
00126      3*          PARAMETER ZPSIZE= 3
00127      4*          DIMENSION VPRINT(5,ZPSIZE)
00130      5*          DATA VPRINT(1, 1)/'TIME   '/
```

Figure 6-5. Vary/Define Feature Example (Cont)

```
00132     6*        DATA VPRINT(1, 2)/'OUTPUT'/
00134     7*        DATA VPRINT(1, 3)/'TESTER'/
00136     8*        PARAMETER ZDRMSZ=1000
00137     9*        DIMENSION VDRUM(ZDRMSZ, 1)
00140    10*        COMMON /DRMHED/ZDATE(2),ZTOD(2),TSTART,TSTOP,VEQDT,SETTLE,
00140    11*       ,ZEDIT,ZNPU,ZRESRV,ZUSED,ZNAMES,ZNAME( 1)
00141    12*        DATA ZNPU/ 1/,  ZRESRV/262129/,  ZUSED/ 0/,  ZNAMES/ 1/
00146    13*        DATA ZNAME( 1)/'OUTPUT'/
00150    14*        DEFINE TESTER=1,+(BW-10,)/20
00151    15*        CALL DATIN(ZDATE)
00152    16*        NPRINT=10
00153    17*        ZDL2=(110,-10.)/20.
00154    18*        DO 200 ZDL1=0,ZDL2,1
00157    19*        BW=10,+20,*ZDL1
00160    20*        DT=,05/BW
00161    21*        TSTOP=5/BW
00162    22*        INCLUDE MAIN2,LIST
00163    22*        IF((TSTOP-TSTART)/DT,GT,0) GO TO 5
00165    22*        WRITE(6,4)TSTART,TSTOP,DT
00172    22*      4 FORMAT(1X,'** ERROR ** THE VALUES TSTART=',E12.6,' TSTOP=',E12.6,
00172    22*       ,  ' DT=',E12,6,' ARE UNREASONABLE,')
00173    22*        STOP
00174    22*      5 ZEDIT=(TSTOP-(TSTART+SETTLE))/(DT*ZRESRV)+1
00175    22*        ZEDCNT=ZEDIT-1
00176    22*        ZDRUM=0
00177    22*        ZUSED=0
00200    22*        ZCOUNT=NPRINT-1
00201    22*        ZPRINT=1
00202    22*        VSUBO=0,0
00203    22*        DO 6 Z1=1,VSIZE
00206    22*      6 V(Z1)=0,0
00210    22*        DO 7 Z2=1,VFSIZE
00213    22*        DO 7 Z1=1,5
00216    22*      7 VF(Z1,Z2)=0.0
00221    22*        VEQDT=DT
00222    22*        VDT2=DT/2.0
00223    22*        VTDT=2,0/DT
00224    22*        TIME=TSTART-DT
00225    22*        VSETTL=TSTART+SETTLE
00226    22*        CALL PAGER
00227    22*        @.................................................................
00227    22*        @  THE SIMULATION LOOP BEGINS HERE,  GOOD LUCK,
00227    22*        @.................................................................
00227    22*     10 TIME=TIME+DT
00230    22*        IF(TIME,GT,TSTOP) GO TO 100
00232    22*        Z=2
00233    22*        GO TO 150
00234    22*     20 Z1STF=1
```

Figure 6-5. Vary/Define Feature Example (Cont)

```
00235     22*          VIN=1
00236     22*          VOUT=2
00237     22*          END
00240     23*          V(Z+4)=1.0
00240     24*    C     BUTTERWORTH
00241     25*          VIN=Z+4
00242     26*          VOUT=Z+6
00243     27*          CALL  BUTWTH(5,1,0.,BW,0.,1.)
00244     28*          IF(TIME.LT.VSETTL) GO TO 10
00246     29*          INCLUDE MAIN3,LIST
00247     29*          ZEDCNT=ZEDCNT+1
00250     29*          IF(ZEDCNT.NE.ZEDIT) GO TO 40
00252     29*          ZEDCNT=0
00253     29*          IF(ZDRUM.NE.ZDRMSZ) GO TO 30
00255     29*          CALL DRUMIT(VDRUM,ZDRMSZ,ZDRMSZ)
00256     29*          ZDRUM=0
00257     29*       30 ZDRUM=ZDRUM+1
00260     29*          END
00261     30*          VDRUM(ZDRUM,1)=V(Z+6)
00262     31*       40 CONTINUE
00263     32*          ZCOUNT=ZCOUNT+1
00264     33*          IF(ZCOUNT.NE.NPRINT) GO TO 10
00266     34*          ZCOUNT=0
00267     35*          ZPRINT=ZPRINT+1
00270     36*          VPRINT(ZPRINT,1)=TIME
00271     37*          VPRINT(ZPRINT,2)=V(Z+6)
00272     38*          VPRINT(ZPRINT,3)=TESTER
00273     39*          IF(ZPRINT.NE.5) GO TO 10
00275     40*          WRITE(6,50) VPRINT
00303     41*       50 FORMAT( 3(6X,A6,4(4X,E12.6),/))
00304     42*          INCLUDE MAIN5,LIST
00305     42*          WRITE(6,60)
00307     42*       60 FORMAT(//)
00310     42*          ZPRINT=1
00311     42*          GO TO 10
00312     42*          @-------------------------------------------------------------
00312     42*          @ END OF SIMULATION AND OUTPUT LOOP
00312     42*          @-------------------------------------------------------------
00312     42*      100 IF(ZPRINT.LT.2) GO TO 130
00314     42*          DO 110 Z2=1,ZPSIZE
00317     42*      110 WRITE(6,120)(VPRINT(Z1,Z2),Z1=1,ZPRINT)
00326     42*      120 FORMAT(6X,A6,8(4X,E12.6))
00327     42*      130 CONTINUE
00330     42*          END
00331     43*          CALL DRUMIT(VDRUM,ZDRMSZ,ZDRUM)
00332     44*          CALL PTPLT('OUTPUT',4)
00333     45*          GO TO 200
00334     46*      150 ZZ=     9
00335     47*          GO TO 20
00336     48*      200 CONTINUE
00340     49*          STOP
00341     50*          END

      END OF UNIVAC 1108 FORTRAN V COMPILATION,      0 *DIAGNOSTIC* MESSAGE(S)
```

Figure 6-5. Vary/Define Feature Example (Cont)

TEST THE VARY/DEFINE FEATURE

TIME	.000000	.500000-01	.100000+00	.150000+00
OUTPUT	.597958-04	.434828+00	.112984+01	.954790+00
TESTER	.100000+01	.100000+01	.100000+01	.100000+01

TIME	.200000+00	.250000+00	.300000+00	.350000+00
OUTPUT	.101650+01	.994127+00	.100202+01	.999333+00
TESTER	.100000+01	.100000+01	.100000+01	.100000+01

TIME	.400000+00	.450000+00	.500000+00
OUTPUT	.100021+01	.999940+00	.100002+01
TESTER	.100000+01	.100000+01	.100000+01

Figure 6-5. Vary/Define Feature Example (Cont)

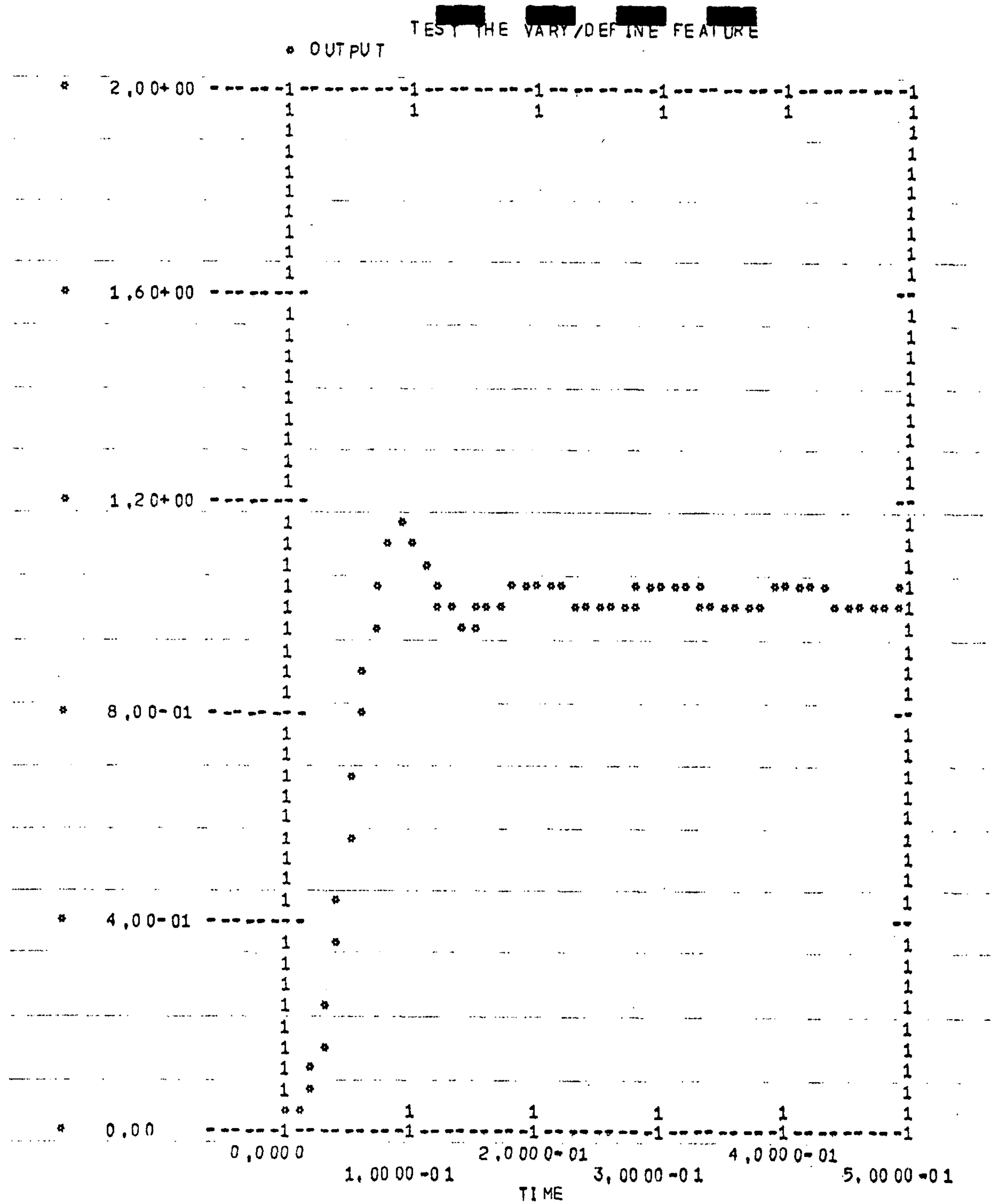


Figure 6-5. Vary/Define Feature Example (Cont)

TEST THE VARY/DEFINE FEATURE

TIME	,000000	,454545-02	,909091-02	,136364-01
OUTPUT	,597958-04	,434828+00	,112984+01	,954790+00
TESTER	,600000+01	,600000+01	,600000+01	,600000+01

TIME	.181818-01	,227273-01	,272727-01	,318182-01
OUTPUT	,101650+01	,994127+00	,100202+01	,999333+00
TESTER	,600000+01	,600000+01	,600000+01	,600000+01

TIME	,363636-01	,409091-01	,454545-01
OUTPUT	,100021+01	,999939+00	,100002+01
TESTER	,600000+01	,600000+01	,600000+01

Figure 6-5. Vary/Define Feature Example (Cont)

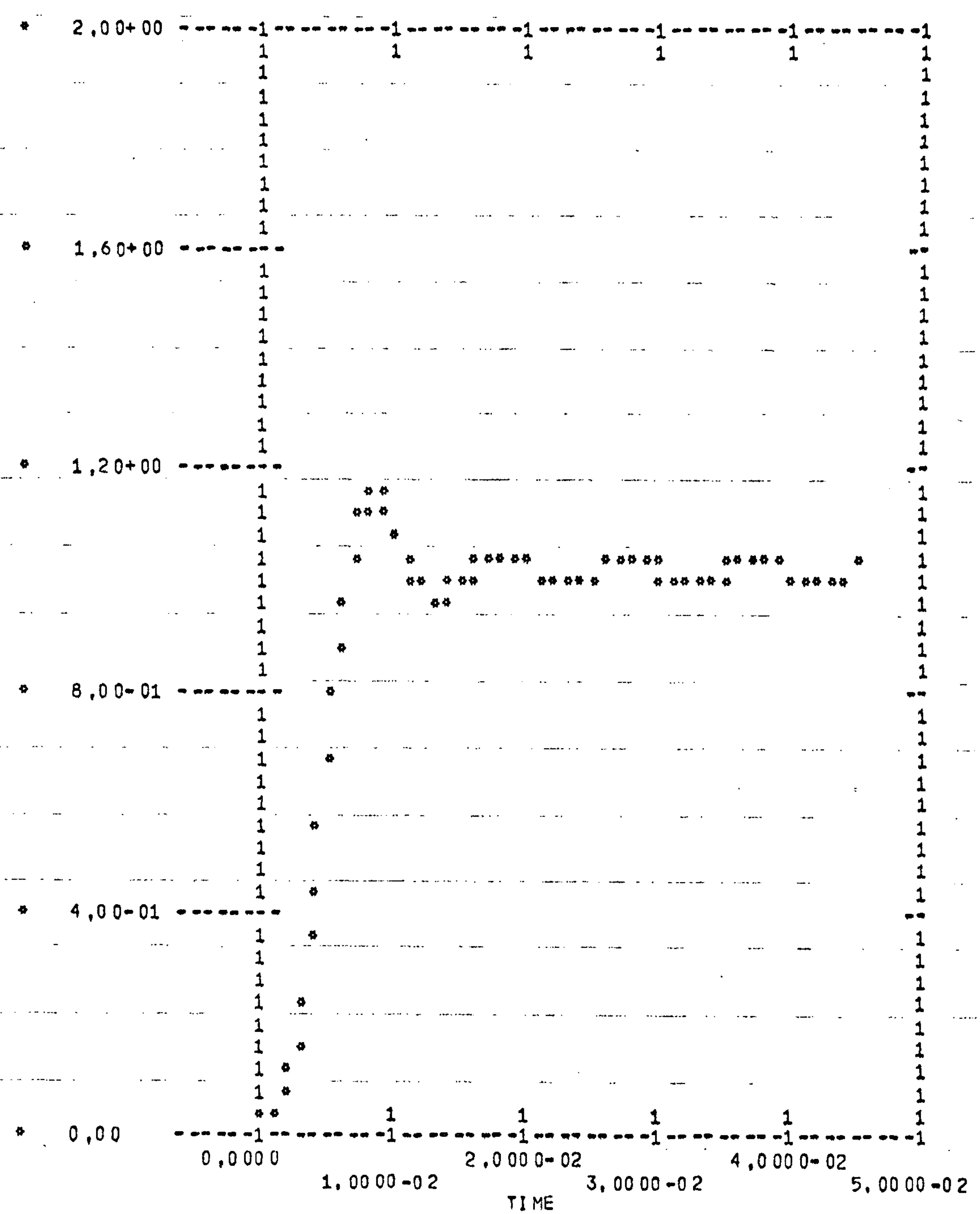


Figure 6-5. Vary/Define Feature Example (Cont)

APPENDIX A

BIBLIOGRAPHY AND REFERENCES

(1) Golden, R. M. and Kaiser, J. F., "Design of Wideband Sampled Data Filters", B.S.T.J., pp. 1533-1545, vol. 43, part 2, July 1964.

(2) Kaiser, J. F., "Design Methods for Sampled-Data Filters", Proc. First Allerton Conference on Circuit and System Theory, Nov., 1963, Monticello, Illinois, pp. 221-236.

(3) Golden, R. M., "Digital Computer Simulation of a Sampled-Data Voice-Excited Vocoder", J. Acoust. Soc. Am., 35, Sept., 1963, pp. 1358-1366.

(4) Wilts, C. H., Principles of Feedback Control, Addison-Wesley, 1960, pp. 197-207.

(5) Hamming, R.W., Numerical Methods for Scientists and Engineers, McGraw-Hill, New York, 1962, pp. 277-280.

(6) Kelly, J. L., Jr., Lochbaum, C. and Vyssotsky, V. A. "a Block Diagram Compiler", B.S.T.J., 40, May, 1961, pp. 669-676.

(7) Storer, J. E., Passive Network Synthesis, McGraw-Hill, New York, 1957, pp. 293-296.

(8) Kuo, F. F. and Kaiser, J. F., System Analysis By Digital Computer, J. Wiley & Sons, Inc., New York, 1966.

(9) D. C. Baxter, Digital Simulation Using Approximate Methods, National Research Council, Ottawa, Canada, Report MK-15, Division of Mechanical Engineering, July 1965.

(10) J. E. Gibson, Nonlinear Automatic Control, McGraw Hill, New York, 1963, pp. 147-159.

(11) K. Steiglitz, The General Theory of Digital Filters with Applications to Spectral Analysis, AFOSR Report No. 64-1664, New York University, New York, May 1963.

(12) K. Steglitz, "The Equivalence of a Digital and Analog Signal Processing", Information and Control, Vol. 8, No. 5, October 1965, pp. 455-467.

(13) J. F. Kaiser, "Some Practical Considerations in the Realization of Linear Digital Filters", Proceedings Third Allerton Conference on Circuit and System Theory, Monticello, Illinois, October 1965, pp. 621-633.

(14) G. E. Heyliger, The Scanning Function Approach to the Design of Numerical Filters, Report R-63-2, Martin Co., Denver, Colorado, April 1963.

(15) R. J. Graham, Determination and Analysis of Numerical Smoothing Weights, NASA Technical Report No. TR-R-179, December 1963.

(16) E. B. Anders et al., Digital Filters, NASA Contractor Report CR-136, December 1964.

(17) D. G. Watts, Optimal Windows for Power Spectra Estimation, Mathematics Research Center, University of Wisconsin, MRC-TSR-506, September 1964.

(18) E. Parzen, Notes on Fourier Analysis and Spectral Windows, Applied Mathematics and Statistical Laboratories, Technical Report No. 48, May 15, 1963, Stanford University, California.

(19) G. A. Campbell and R. M. Foster, Fourier Integrals for Practical Applications, D. Van Nostrand, 1948, p. 113 pair 872.1.

(20) J. F. Kaiser, "A family of window functions having nearly ideal properties", November 1964, unpublished memorandum.

(21) D. Slepian and H. O. Pollak, "Prolate spheroidal wave functions, Fourier analysis and uncertainty - I and II", B. S. T. J. Vol. 40, No. 1, January 1961, pp. 43-84.

(22) P. E. Fleischer, "Digital realization of complex transfer functions", Simulation, Vol. 6, No. 3, March 1966, pp. 171-180.

(23) M. A. Martin, Digital Filters for Data Processing, General Electric Co., Missile and Space Division, Tech. Info. Series Report No. 62-SD484, 1962.

(24) S. A. Schelkunoff, "A mathematical theory of linear arrays", B.S.T.J. Vol. 22, January 1943, pp. 80-107. Mark Tsu-Han Ma, A New Mathematical Approach for Linear Array Analysis and Synthesis, Ph.D. thesis, Syracuse University, 1961, University Microfilms No. 62-3050.

(25) G. M. Jenkins, "A survey of spectral analysis", Applied Statistics, Vol. XIV, No. 1, 1965, pp. 2-32.

(26) H. H. Robertson, "Approximate design of digital filters," Technometrics, Vol. 7, No. 3, August 1965, pp. 387-403.

(27) W. K. Linvill, "Sampled-data control systems studied through comparison of sampling with amplitude modulation, "Trans. AIEE, Vol. 70, Part II, 1951, pp. 1779-1788.

(28) A. Susskind, Notes on Analog-Digital Conversion Techniques, John Wiley, New York, 1957. See especially Chapter II, Sampling and Quantization by D. Ross.

(29) D. T. Ross, Improved Computational Techniques for Fourier Transformation, Servomechanisms Laboratory Report, No. 7138-R-5, Massachusetts Institute of Technology, Cambridge, Massachusetts, June 25, 1954.

(30) C. Jordan, Calculus of Finite Differences, Chelsea, 1960, (Reprint of 1939 Edition).

(31) H. M. James, N. B. Nichols, R. S. Phillips, Theory of Servomechanisms, McGraw-Hill, New York, 1947, pp. 231-261.

(32) J.R. Ragazzini and G. F. Franklin, Sampled-Data Control Systems, McGraw Hill, 1958.

(33) E. I. Jury, Sampled-Data Control Systems, John Wiley and Sons, Inc. 1958.

(34) J. T. Tou, Digital and Sampled-Data Control Systems, McGraw-Hill, 1959.

(35) E. Mishkin and L. Braun, Jr., Adaptive Control Systems, McGraw Hill, New York, 1961, pp. 119-183.

(36) E. I. Jury, Theory and Application of the z-Transform Method, John Wiley, New York, 1964.

(37) H. Freeman, Discrete-Time Systems, John Wiley, 1965.

(38) P. M. DeRusso, R. J. Roy, C. M. Close, State Variables for Engineers, John Wiley, 1965, pp. 158-186.

(39) R. B. Blackman, Linear Data-Smoothing and Prediction in Theory and Practice, Addison-Wesley, Reading, Massachusetts, 1965.

(40) C. Lanczos, Applied Analysis, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1956.

(41) R. B. Blackman, J. W. Tukey, The Measurement of Power Spectra from the Point of View of Communication Engineering, Dover, 1959.

(42) M. A. Martin, "Frequency Domain Applications to date Processing," IRE Transactions on Space Electronics and Telemetry, Vol SET-5, No. 1, March 1959, pp. 33-41.

(43) J. F. A. Ormsby, "Design of Numerical Filters with Applications to Missile Data Processing," Jour. ACM, Vol. 8, No. 3, July 1961, pp. 440-466.

(44) A. J. Monroe, Digital Processes for Sampled Data Systems, John Wiley and Sons, Inc., New York, 1962.

(45) R. M. Golden, "Digital Computer Simulation of Communication Systems Using the Block Diagram Compiler; BLODIB, "Third Annual Allerton Conference on Circuit and System Theory, Monticello, Illinois, October 1965, pp. 690-707.

(46) A. Tustin, "A Method of Analyzing the Behavior of Linear Systems In Terms of Time Series," Jour. IEE, Vol. 94, Part II A, May 1947, pp. 130-142.

(47) J. M. Salzer, "Frequency Analysis of Digital Computers Operating In Real Time," Proc, IRE, Vol. 42, February 1954, pp. 457-466.

(48) R. Boxer, S. Thaler, "A Simplified Method of Solving Linear and Non-Linear Systems," Proc. IRE, Vol. 44, January 1956, pp. 89-101.

(49) R. Boxer, "A Note on Numerical Transform Calculus," Proc. IRE, Vol. 45, No. 10, October 1957, pp. 1401-1406.

(50) D. C. Baxter, The Digital Simulation of Transfer Functions, National Research Laboratories, Ottawa, Canada, DME Report No. MK-13, April 1964.

(51) C. J. Drane, Directivity and Beamwidth Approximations for Large Scanning Dolph-Chebyshev Arrays, AFCRL Physical Science Research Papers No. 117, AFCRL-65-472, June 1965.

(52) H. L. Garabedian (Ed.), Approximation of Functions, Elsevier Publishing Co., 1965, see especially Walsh pp. 1-16 and Cheney pp. 101-110.

(53) E. W. Cheney and H. L. Loeb, "Generalized rational approximation, "J. SIAM, Numerical Analysis, B, Vol. 1, 1964, pp. 11-25.

(54) Josef Stoer, "A direct method for Chebyshev approximation by rational functions," JACM, January 1964, Vol. 11, No. 1, pp. 59-69.

(55) R. M. Golden and J. F. Kaiser, "A computer program for the design of continuous and sampled-data filters", to be published.

(56) C. M. Rader and B. Gold, Digital Filter Design Techniques, Lincoln Laboratory Report, M. I. T., Preprint JA2612, September 1965.

(57) H. Holtz and C. T. Leondes, "The synthesis of recursive filters, "Jour. ACM, Vol. 13, No. 2, April 1966, pp. 262-280.

(58) L. Weinberg, Network Analysis and Synthesis, McGraw Hill, 1962.

(59) D. A. Calahan, Modern Network Synthesis, Hayden, New York, 1964.

(60) J. E. Storer, Passive Network Synthesis, McGraw Hill, New York, 1957, pp. 287-302.

(61) P. Broome, "A frequency transformation for numerical filters," Proc. IEEE, Vol. 52, No. 2, February 1966, pp. 326-7.

(62) P. E. Mantey, Convergent Automatic-Synthesis Procedures for Sampled-Data Networks with Feedback, Stanford Electronics Laboratories, Technical Report No. 6773-1, SU-SEL-112, Stanford University, October 1964.

(63) J. R. B. Whittlesey, "A rapid method for digital filtering, Comm. "ACM, Vol. 7, No. 9, September 1964, pp. 552-556.

(64) T. Y. Young, Representation and Analysis of Signals, Part X. Signal Theory and Electrocardiography, Department of Electrical Engineering, Johns Hopkins University, May 1962.

(65) P. W. Broome, "Discrete orthonormal sequences, "Jour. ACM, Vol. 12, No. 2, April 1965, pp. 151-168. Archambeau et al., "Data processing techniques for the detection and interpretation of teleseismic signals, "Proc. IEEE, Vol 53, No. 12, December 1965, pp. 1860-1994, see especially p. 1878.

(66) H. W. Bode, Network Analysis and Feedback Amplifier Design, Van Nostrand, 1945, pp. 47-49.

(67) M. Mansour, "Instability criteria of linear discrete systems, "Automatica, Vol. 2, No. 3, January 1965. pp. 167-178.

(68) C. E. Maley, "The effect of parameters on the roots of an equation system, "Computer Journal, Vol. 4, 1961-2. pp. 62-63.

(69) J. G. Truxal, Automatic Feedback Control System Synthesis, McGraw Hill Book Co., Inc., New York, 1955, pp. 223-250.

(70) F. F. Kuo, Network Analysis and Synthesis, John Wiley, New York, First Edition, 1962, pp. 136-137, (Second Edition, pp. 148-155).

(71) C. Pottle, "On the partial-fraction expansion of a rational function with multiple poles by a digital computer," IEEE Trans. Circuit Theory, Vol. CT-11, March 1964, pp. 161-162.

(72) W. R. Bennett, "Spectra of quantized signals", B. S. T. J. Vol. 27, July 1948, pp. 446-472.

(73) B. Widrow, "A study of rough amplitude quantization by means of Nyquist sampling theory," Trans, IRE on Circuit Theory, Vol. CT-3, No. 4, December 1956, pp. 266-276.

(74) B. Widrow, "Statistical analysis of amplitude quantized sampled-data systems," Trans. AIEE Applications and Industry, No. 52, January 1961, pp. 555-568.

(75) F. B. Hills, A Study of Incremental Computation by Difference Equations, Servomechanisms Laboratory Report No. 7849-R-1, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1958.

(76) J. B. Knowles and R. Edwards, "Effect of a finite-word-length computer in a sampled-data feedback system," Proc. IEE Vol. 112, No. 6, June 1965, pp. 1197-1207.

(77) J. B. Knowles and R. Edwards, "Simplified analysis of computational errors in a feedback system incorporating a digital computer," S. I. T. Symposium on Direct Digital Control, April 22, 1965, London.

(78) J. B. Knowles and R. Edwards, "Complex cascade programming and associated computational errors," Electronics Letters, Vol. 1, No. 6, August 1965, pp. 160-161.

(79) J. B. Knowles and R. Edwards, "Finite word-length effects in multirate direct digital control systems," Proc. IEE, Vol. 112, No. 12, December 1965, pp. 2376-2384.

(80) B. Gold and C. Rader, "Effects of quantization noise in digital filters," Proceedings Spring Joint Computer Conference, 1966. Vol. 28, pp. 213-219.

(81) J. H. Wilkinson, Rounding Errors in Algebraic Processes, Prentice Hall, Englewood Cliffs, New Jersey, 1963.

(82) J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Mathematics of Computation, Vol. 19, No. 90, April 1965, pp. 297-301.

(83) R. A. Gaskill, "A versatile problem-oriented language for engineers, "IEEE Transactions on Electronic Computers, Vol. EC-13, No. 4 (August, 1964), pp. 415-421.

(84) R. D. Brennan and R. N. Linebarger, "A survey of digital simulation: digital analog simulator programs, "Simulation Vol. 3, No. 6 (December, 1964), pp. 22-36.

(85) J. J. Clancy and M. S. Dineberg, "Digital simulation languages: a critique and a guide, "AFIPS Conference Proceedings, Vol. 27, Part I (1965), Spartan Books, Washingron, D.C., pp. 23-36.

(86) J. C. Strauss and W. L. Gilbert, SCADS: a programming system for the simulation of combined analog digital systems, Carnegie Institute of Technology, March, 1964.

(87) W. M. Syn and D.-G. Wyman, DSL/90 Digital Simulation Language User's Guide, IBM Corporation, San Jose, California, July, 1965.

(88) R. W. Burt and A. P. Sage, "Optimum design and error analysis of digital integrators for discrete system simulation, "AFIPS Conference Proceedings, Vol. 27, Part I (1965), Spartan Books, Washington, C.D., pp. 903-914.

(89) M. E. Fowler, "A new numerical method for simulation," Simulation, Vol. 4, No. 5 (May, 1965), pp. 324-330.

(90) C. C. Cutler, "Transmission Systems Employing Quantization, "U.S. Patent No. 2, 927, 962, March 8, 1960 (filed April 26, 1954).

(91) T. G. Stockham, Jr., "High speed convolution and correlation, "Proceedings Spring Joint Computer Conference, 1966, Vol. 28, pp. 229-233.

(92) H. D. Helms, "Fast Fourier transforms methods of computing difference equations arising from z-transforms and autoregressions," (to appear).

(93) J. L. Kelly, Jr., C. Lochbaum, and V. A. Vyssotsky, "A block diagram compiler", B.S.T.J., Vol. 40, No. 3 (May, 1961) pp. 669-676.

(94) M. R. Schroeder et al., New methods for speech analysis-synthesis and bandwidth compression, Congress Report of the Fourth International Congress on Acoustics, Copenhagen, 1962.

(95) L. S. Frishkopf and L. D. Harmon, "Machine recognition of cursive script," Symposium on Information Theory, London (1960), C. Cherry, Editor, Butterworth and Co. Ltd, London (1961), pp. 300-316.

(96) B. J. Karafin, "The new block diagram compiler for simulation of sampled-data systems," AFIPS Conference Proceedings, Vol. 27, Part I (1965), Spartan Books, Washington, D. C. pp. 53-62.

(97) A study of Math Model Input/Output Parameters for the Computer Aided Analysis Program HASD 6420-821429, Lockheed Electronics Company.

(98) G. Franklin, "Linear Filtering of Sampled Data", IRE Conv. Rec., Vol. 3, Pt IV, pp 119-128, 1955.

(99) R. P. Brennan and R. N. Linebarger, "An Evaluation of Digital Analog Simulator Languages", I. F. I. P. 1965 Proceedings, Vol. 2.

(100) J. R. Hurley and J. J. Skiles, "DYSAC", 1963 SJCC, Vol. 23, Spartan Books, Washington, D. C.

(101) V. C. Rideout and L. Tavernini, "MAD BLOC", Simulation, Vol. 4, No. 1, January 1965.

(102) M. L. Stein, J. Rose and D. B. Parker, "A Compiler with An Analog Oriented Input Language (ASTRAC)", Proc. 1959 WJCC.

(103) F. J. Sansom and H. E. Peterson, "MIMIC - Digital Simulator Program", SESCA Internal Memo 65-12, WPAFB, May 1965.

(104) R. T. Harnett and F. T. Sansom, "MIDAS Programming Guide", Report No. 5EG-TDR-64-1, WPAFB, Ohio, January 1964.

(105) M. Palevsky and J. V. Howell, "DES-1", Fall JCC, Vol. 24, Spartan Books, Inc., Washington, D. C., 1963.

(106) R. D. Brennan and H. Sano, "PACROLUS", Fall JCC, Vol. 26, Spartan Books, Inc., Washington, D. C., 1964.

(107) R. N. Linebarger, "DSL/90", Paper at Joint Meeting Midwestern and Central States Simulation Councils, May 1965.

(108) R. A. Gaskill, J. W. Harris, and A. L. McKnight, "DAS-A Digital Analog Simulator", Proc. 1963 Spring JCC, AFIPS Conference Proc., Vol. 23, p. 83.

(109) G. E. Blechman, "An Enlarged Version of MIDAS", S&I Div. NAR, June 1964; Simulation, Vol. 3, No. 4, October 1964.

(110) M. E. Fowler, "A New Numerical Method for Simulation", Simulation, pp. 324-330, May 1965.

(111) M. E. Fowler, "An Example Showing Use of Root Locus Techniques to Study Nonlinear Systems", TR, August 12, 1964, IBM R&D Ctr., Palo Alto, California.

(112) SAI Proposal No. 69-045, dated October 1969, "Time Domain Simulation of Apollo Telecommunications Lines".

(113) W. E. Thompson, "Network with Maximally Flat Delay," Wireless Engineer, Vol. 29, pg 255, October 1952.

(114) "On the Design of Filters by Synthesis," R. Saul and E. Ulbrich, IRE Transactions on Circuit Theory, December 1958.

(115) "The Design of Filters Using the Catalog of Normalized Low-Pass Filters," R. Saal, Telefunken, 1963.

(116) "Passive Network Synthesis," James E. Storer, McGraw-Hill, 1957.

APPENDIX B

ASYSTD LIBRARY DESCRIPTIONS

The ASYSTD library consists of a collection of models developed by both Systems Associates, Inc. and NASA MSC, Space Electronics Division. The following material describes the use of the models. The descriptions are grouped under various identifiers, an index of which follows. Several blank Library forms have been included for future use.

Filters

- General Filter Model
- Butterworth
- Chebychev
- Bessel
- Butterworth-Thompson
- Elliptic
- Quadratic
- Lead Lag Function
- Lead Function
- Matched Filter

Limiters

- Soft Limiters
- Hard Limiters
- RF Soft Limiter
- RF Hard Limiter

Transforms

- Fourier Transform (FFT) (and Inverse)
- Haar Transform (and Inverse)
- Hadamard Transform (and Inverse)
- Ordered Hadamard Transform (and Inverse)

Coders

- Analog-to-Digital
- Digital-to-Analog
- Sample Hold Digital-to-Analog
- Multi-Level PCM

Math

- Complex Adder
- Complex Multiply
- Differentiator
- Integral with Initial Conditions
- Integral

Miscellaneous

- Interleaver
- De-Interleaver
- Time Delay
- Phase Shifter
- Signal Split
- Time Latch
- Zero Crossing Detector

MODEL	GROUP ID	PAGE	DATE
GAUSSIAN NOISE GENERATOR	Sig. Gen.	1	April, 1972

LIBRARY MODEL NAMES

GNOISE

DESCRIPTION

This function provides noise modeling capability, providing the SNR and ENB of the generator are defined.

USAGE

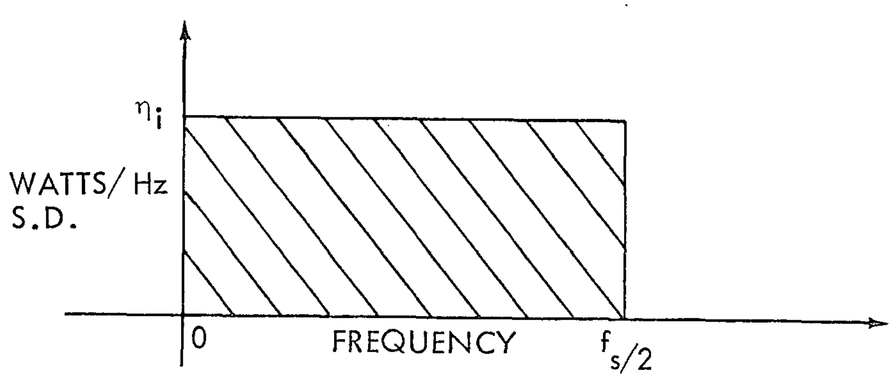
N1 < GNØISE (SNR, ENB, ISTART) > N2

where: SNR is the signal-to-noise ratio desired in ENB (equivalent noise bandwidth) assuming a 1 watt signal level.

ISTART is a positive integer (>0) for initializing the random number generator.

DETAILED DESCRIPTION

WHITE GAUSSION:



MODEL	GROUP ID	PAGE	DATE
GAUSSIAN NOISE GENERATOR	Sig. Gen.	2	April, 1972

where

$$f_s = \frac{1}{DT} = \frac{1}{\text{sampling rate}}$$

$$\sigma_i^2 = \frac{\eta_i f_s}{2} = \frac{\eta_i}{2DT}$$

or

$$\boxed{\eta_i = 2\sigma_i^2 DT} \quad \text{(Watts/Hz)}$$

$$N_o = \eta_i * ENB = 2\sigma^2 DT * ENB \quad \text{(watts)}$$

where ENB = equivalent noise bandwidth under consideration.

For a given SNR in bandwidth, BW:

$$\frac{S}{N_o} = 10^{SNR/10}$$

where

$$S = \text{signal power in BW} \quad \text{(watts)}$$

or

$$N_o = S * 10^{-SNR/10} = 2\sigma_i^2 DT * ENB$$

or

$$\boxed{\sigma_i = \sqrt{S} / \sqrt{10^{SNR/10} * 2DT * ENB}}$$

APPLICATION

This model is a function and may be used in expressions.

MODEL	GROUP ID	PAGE	DATE
GAUSSIAN NOISE TWO	Sig. Gen.	1	April, 1972

LIBRARY MODEL NAMES

GNOIS2

DESCRIPTION

This model provides noise modeling capability, providing the spectral density desired.

USAGE

N1 <GNØIS2 (ETA, ISTART)> N2

where: ETA is the desired spectral density (watts/Hz)

ISTART is a positive integer (>0) for initializing the random number generator

DETAILED DESCRIPTION

See "Gaussian Noise Generator"

$$ETA = \frac{1}{10^{SNR/10} * ENB}$$

APPLICATION

This model is a function and may be used in expressions.

MODEL	GROUP ID	PAGE	DATE
PULSE GENERATOR	Sig. Gen.	1	April, 1972

LIBRARY MODEL NAMES

PULSE

DESCRIPTION

This model produces a periodic output of pulses of the shape described below.

USAGE

N1 < PULSE(RATE, TD, TR, TL, TF) > N2

where: RATE = frequency of output

TD = delay time

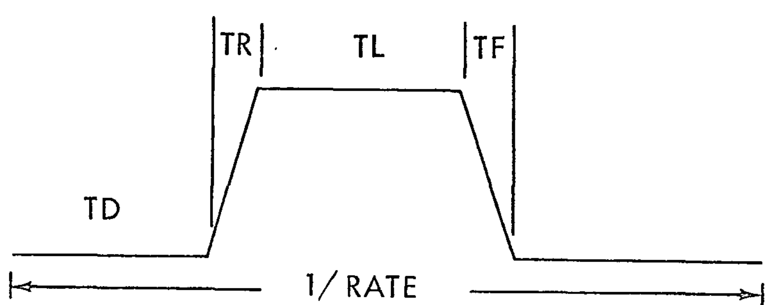
TR = rise time

TL = level time

TF = fall time

OUTPUT

The maximum value is plus one and the minimum value is zero.



APPLICATION

This model is a function and may be used in expressions.

MODEL	GROUP ID	PAGE	DATE
SQUARE WAVE GENERATOR	Sig. Gen.	1	April, 1972

LIBRARY MODEL NAMES

SQ
SQUARE WAVE

USAGE

N1 < SQ(RATE) > N2

where: RATE = frequency of the square wave output at node N2

OUTPUT

A square wave of period 1/RATE whose maximum value is plus one and whose minimum value is minus one.

MODEL	GROUP ID	PAGE	DATE
TABLE	Sig. Gen.	1	April, 1972

LIBRARY MODEL NAMES

TABLE

DESCRIPTION

This function provides a piece-wise linear function for modeling both driving functions and non-linearities.

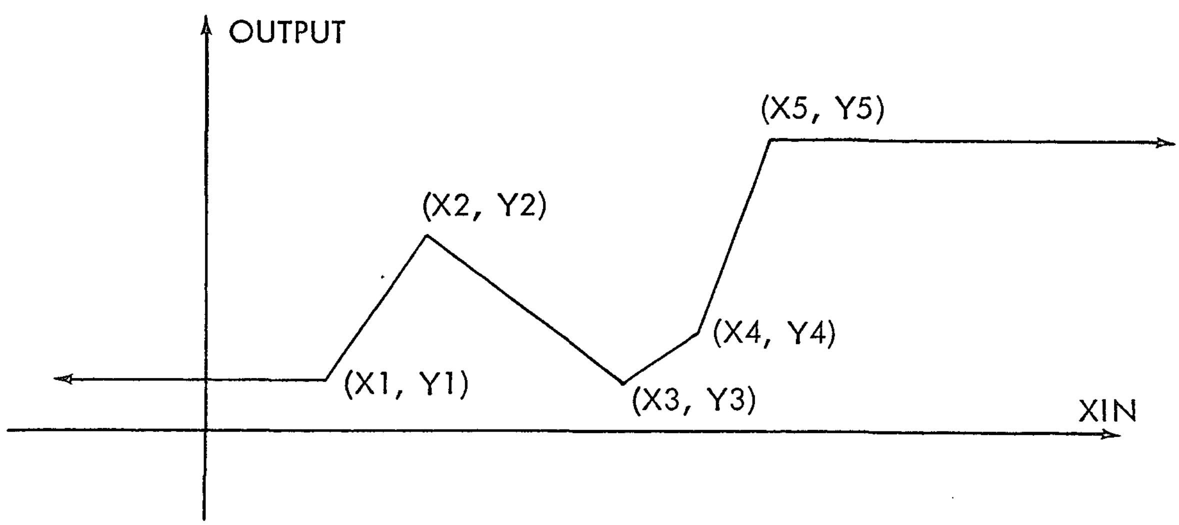
USAGE

N1 < TABLE (XIN, X1, Y1, X2, Y2, X3, Y3, X4, Y4, X5, Y5 > N2

where: XIN = independent variable

X1, Y1 ... X5, Y5 — Five point pairs describing the function (Note: Out-of-range values assume the end point values)

OUTPUT (Example)



MODEL	GROUP ID	PAGE	DATE
TABLE	Sig. Gen.	2	April, 1972

APPLICATION

This model is a function and may be used in expressions.

MODEL	GROUP ID	PAGE	DATE
PERIODIC TABLE FUNCTION	Sig. Gen.	1	April, 1972

LIBRARY MODEL NAMES

PERIODIC TABLE FUNCTION
PTABLE

DESCRIPTION

This model provides the periodic function capability. The output is periodic with period T5.

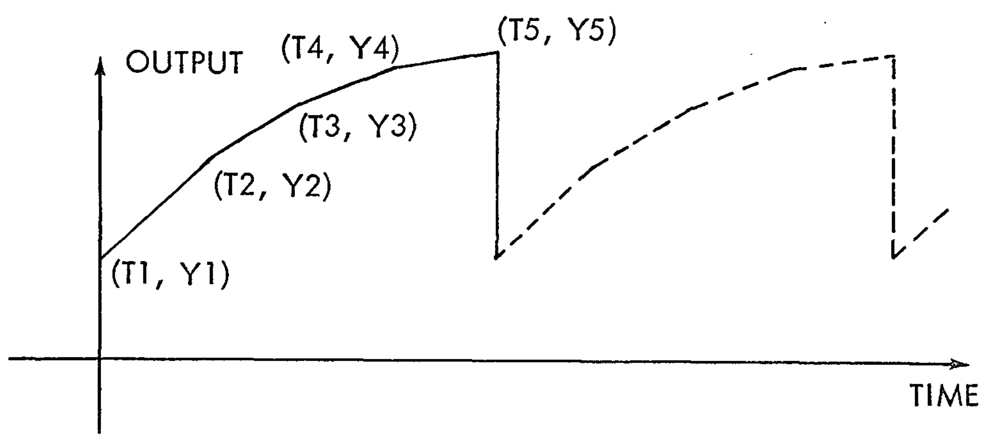
USAGE

N1 < PTABLE (T1, Y1, T2, Y2, T3, Y3, T4, Y4, T5, Y5) > N2

where: T1, Y1 ... T5, Y5 } Five point-pairs describing the function

and T5 = Period of the function

OUTPUT (Example



In this example T1=0; if T1≠0, the output is set to Y1 for 0 ≤TIME ≤T1

APPLICATION

This model is a function and may be used in expressions.

MODEL	GROUP ID	PAGE	DATE
TABL2	Sig. Gen.	1	April, 1972

LIBRARY MODEL NAMES

TABL2

DESCRIPTION

This function provides a piece-wise linear function for modeling both driving functions and non-linearities, and provides zero output levels for out of range conditions.

USAGE

N1 <TABL2 (XIN, X1, Y1, X2, Y2, X3, Y3, X4, Y4, X5, Y5)> N2

where: XIN = independent variable

X1, Y1 ... X5, Y5 } five point pairs describing the function

OUTPUT

This model is the same as model "TABLE" except out-of-range values (XIN<X1 or XIN ≥ X5) yield an output of zero.

APPLICATION

This model is a function and may be used in expressions.

MODEL	GROUP ID	PAGE	DATE
TRANSCENDENTAL FUNCTIONS	Sig. Gen.	1	April, 1972

LIBRARY MODEL NAMES

SIN	SINE
COS	COSINE
TAN	TANGNT

DESCRIPTION

These elements are FORTRAN transcendental functions or utilize FORTRAN functions.

USAGE (Example)

N1 < SIN ($) > N2

N2 is set to the trigonometric sine of N1

SIN (x) COS (x) TAN (x)	x in radians	SINE (y) COSINE (y) TANGNT (y)	y in cycles

APPLICATION

These elements are functions and may be used in expressions.

MODEL	GROUP ID	PAGE	DATE
AMPLITUDE MODULATOR	Modulator	1	April, 1972

LIBRARY MODEL NAMES

AM MODULATOR
AMMOD

DESCRIPTION

The Linear Amplitude Modulator provides classical modulation capability to the ASYSTD user. This element is the baseband model.

USAGE

N1 <AMMOD(BETA, FC)> N2

where: BETA = Modulation Index (ratio)

FC = Carrier frequency

OUTPUT

Let INPUT(t) be the real input to the model (value at node N1) and Vo(t) be the real output of the model, then

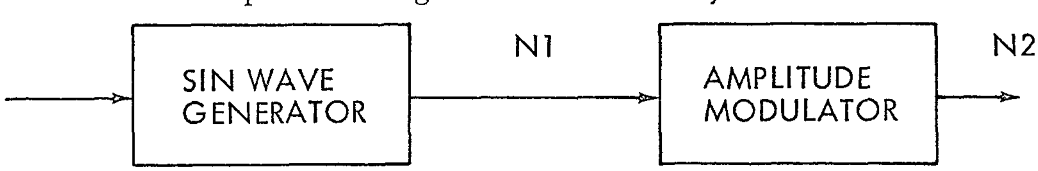
$$Vo(t) = (1.0 + BETA \cdot INPUT\ (t)) \cdot \cos\ (2\pi FC \cdot Time)$$

RESTRICTIONS

BETA*INPUT(t) ≤ 1.0 for no over-modulation.

APPLICATION

An example of using the model in a system is as follows:



INPUT < SINE (T) > N1
N1 '< AMMOD (1.0, 100 E3) > N2

MODEL	GROUP ID	PAGE	DATE
RF AMPLITUDE MODULATOR	Modulator	1	April, 1972

LIBRARY MODEL NAMES

RF AMPLITUDE MODULATOR
RAMMOD

DESCRIPTION

The Linear Amplitude Modulator provides classical modulation capability to the ASYSTD user. The output of this model is a complex baseband signal which represents the modulated carrier in the baseband.

USAGE

N1 < RAMMOD (BETA, PHI, MODE) > N2

where: BETA = Modulation Index (ratio)

PHI = Instantaneous Phase Jitter

MODE = 1.0 for Double Sideband (DSB)

MODE = 0.0 for Double Sideband-Suppressed Carrier

OUTPUT

Let i(t) be the real input to the model (node N1),

then: $$Vo(t) = \left[MODE + BETA*i_{(t)}\right] e^{j(\omega_c t + PHI)}$$

translating Vo(t) to baseband:

$$Vo'(t) = Vo(t) e^{-j\omega_c t}$$

or $$Vo'(t) = \left|MODE + BETA*i(t)\right| e^{jPHI} = Vr(t) + jVi(t)$$

where: $Vo'(t)$ is the complex baseband output signal at N2

with $$Vr(t) = \left|MODE + BETA*i(t)\right| COS(PHI)$$

$$Vj(t) = \left|MODE + BETA*i(t)\right| SIN(PHI)$$

RESTRICTIONS

$|BETA*i(t)| \leq 1.0$ for no over-modulation

APPLICATION

This model is used in place of the amplitude modulator when carrier translation is required.

MODEL	GROUP ID	PAGE	DATE
LINEAR FREQUENCY MODULATOR	Modulator	1	April, 1972

LIBRARY MODEL NAMES

FM MODULATOR
FMMOD

DESCRIPTION

The Linear Frequency Modulator Model provides a classical model for this type of angle modulation. The carrier output magnitude is defined as unity.

USAGE

N1 < FMMOD(DF, FC) > N2

where: DF = frequency deviation (Hz) of the carrier per unit input

FC = carrier frequency

OUTPUT

Let the signal at N1 at time T be i(t).
Let the signal at N2 at time T be Vo(t).

then: $$Vo(t) = SINE(FC*t + DF*\int_{TSTART}^{t} i(\tau)\, d\tau)$$

NOTE: The arguments of the SINE function is in Hz.

APPLICATION

FMMOD is for use when no carrier translation is required in the simulation.

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE
RF LINEAR FREQUENCY MODULATOR	Modulator	1	April, 1972

LIBRARY MODEL NAMES

RF FM MODULATOR
RFMMOD

DESCRIPTION

The Linear Frequency Modulator model provides a classical model for this type of angle modulation. The carrier output magnitude is defined as unity.

USAGE

N1 < RFMMOD(DF) > N2

where: DF = frequency deviation (Hz) of the carrier per unit input.

OUTPUT

Let i(t) be the real input to the model (node N1), then for the ideal FM modulator

$$Vo(t) = e^{j(\omega_c t + \beta\int_o^t i(t)dt)}$$

where: $\beta = Z \ \pi DF$

translating Vo(t) to baseband:

$$Vo'(t) = Vo(t)\, e^{-j\omega_c t}$$

or

$$Vo'(t) = e^{j\beta\int_o^t i(t)dt} = Vr(t) + jVj(t)$$

where: $Vo'(t)$ is the complex baseband output signal at N2

with: $Vr(t) = COS\ (\beta\int_o^t i(t)dt)$

$Vj(t) = SIN\ (\beta\int_o^t i(t)dt)$

APPLICATION

This model is used in place of the FREQUENCY MODULATOR model when carrier translation is required in the simuation.

MODEL	GROUP ID	PAGE	DATE
SQUARE WAVE FREQUENCY MODULATOR	Modulators	1	

LIBRARY MODEL NAMES

SQUARES WAVE FREQUENCY MODULATOR
SQFMOD

DESCRIPTION

The Square Wave Frequency Modulator provides a model for this type of angle modulation with ideal limiting.

USAGE

N1 < SQFMOD(DF, FC) > N2

where: DF = frequency deviation (cycles) of the carrier per unit input

FC = carrier frequency

OUTPUT

Let the signal at N1 at time T be INP(T).
Let the signal at N2 at time T be OUT(T).

Then: $$F(T) = \text{SINE}\left(FC*T + DF*\int_{TSTART}^{T} INP(t)dt\right)$$ [linear frequency modulation]

NOTE: The arguments of the SINE function is in cycles

OUT(T) = +1 when F(T) ≥ 0
OUT(T) = -1 when F(T) < 0

MODEL	GROUP ID	PAGE	DATE
LINEAR PHASE MODULATOR	Modulator	1	April, 1972

LIBRARY MODEL NAMES

PHASE MODULATOR
PMMODD
PMMOD

DESCRIPTION

The Linear Phase Modulator provides a classical model for this type of angle modulation. The carrier output level is defined as unity.

USAGE

N1 < PMMOD(BETA, TC) > N2

where: BETA = Phase (Radians) deviation per unit input

FC = Carrier frequency (Hz)

NOTE: Modulation Index = BETA * $Input_{max}$

OUTPUT

Let the signal at N1 at time T be i(t)
Let the signal at N2 at time T be Vo(t)

then: $Vo(t) = \sin(2\pi * FC * t + BETA * i(t))$

APPLICATION

This model is for use when no carrier translation is required in the simulation.

MODEL	GROUP ID	PAGE	DATE
RF LINEAR PHASE MODULATOR	Modulator	1	April, 1972

LIBRARY MODEL NAMES

RF PHASE MODULATOR
RPMMOD

DESCRIPTION

The Liner Phase Modulator provides a classical model for this type of angle modulation. The output is the complex baseband signal.

USAGE

N1 < RPMMOD(BETA) > N2

where: BETA = Phase (Radians) deviation per unit input

OUTPUT

Let i(t) be the real input to the model, then:

$$Vo(t) = e^{j\{\omega_c t + \beta \cdot i(t)\}}$$

translating Vo(t) to baseband:

$$Vo'(t) = Vo(t)e^{-j\omega_c t}$$

or

$$Vo'(t) = e^{j\beta \cdot i(t)} = V_r(t) + jV_j(t)$$

where: $Vo'(t)$ is the complex baseband output signal at N2

with $Vr(t) = COS(\beta \cdot i(t))$

$Vj(t) = SIN(\beta \cdot i(t))$

APPLICATION

This model is used in place of the FM MODULATOR model when carrier translation is required in the simulation.

MODEL	GROUP ID	PAGE	DATE
SQUARE WAVE PHASE MODULATOR	Modulator	1	April, 1972

LIBRARY MODEL NAMES

SQUARE WAVE PHASE MODULATOR
SQPMOD

DESCRIPTION

The Square Wave Phase Modulator provides a model for this type of angle modulation with ideal limiting.

USAGE

N1 <SQPMOD(BETA, FC)> N2

where: BETA = Phase (Radians) deviation per unit input

FC = Carrier frequency

NOTE: Modulation Index = $BETA*Input_{max}$

OUTPUT

Let the signal at N1 at time T be INP(T).
Let the signal at N2 at time T be OUT(T).

$F(T) = \sin(2\pi * FC * T + BETA * INP(T))$ [linear phase modulation]

$OUT(T) = +1$ when $F(T) \geq 0$

$OUT(T) = -1$ when $F(T) < 0$

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE
DELTA MODULATION	Modulator	1	April, 1972

LIBRARY MODEL NAMES

DELTA MODULATOR
DELMOD

DESCRIPTION

Delta modulation is a coded modulation system nearly as efficient as PCM, requires more bandwidth than PCM, but has much simpler circuitry. These advantages make delta modulation quite attractive as a standard model. The output waveform magnitude is defined as ± 1.

USAGE

N1 <DELMOD(PW, PPS)> N2

where: PW = Pulse width (unit time)

PPS = Pulse repetition rate (pulses/unit time)

OUTPUT

In a delta modulation system, only the changes in signal amplitude from sample to sample are output. The process consists of utilizing a pulse generator (clock), one shot multi-vibrator, an integrator, and a difference circuit.

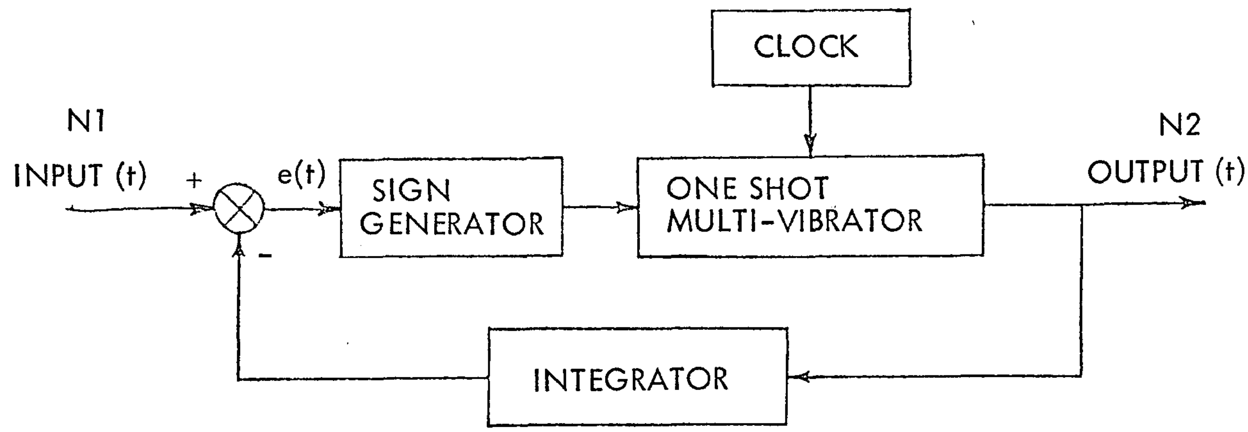
$$\text{let } e_{(t)} = \text{input}_{(t)} - \int \text{Output}_{(t)}\, dt$$

$$\text{where output}_{(t)} = \text{Sign}\,(e_{(t)}) * \delta(t)$$

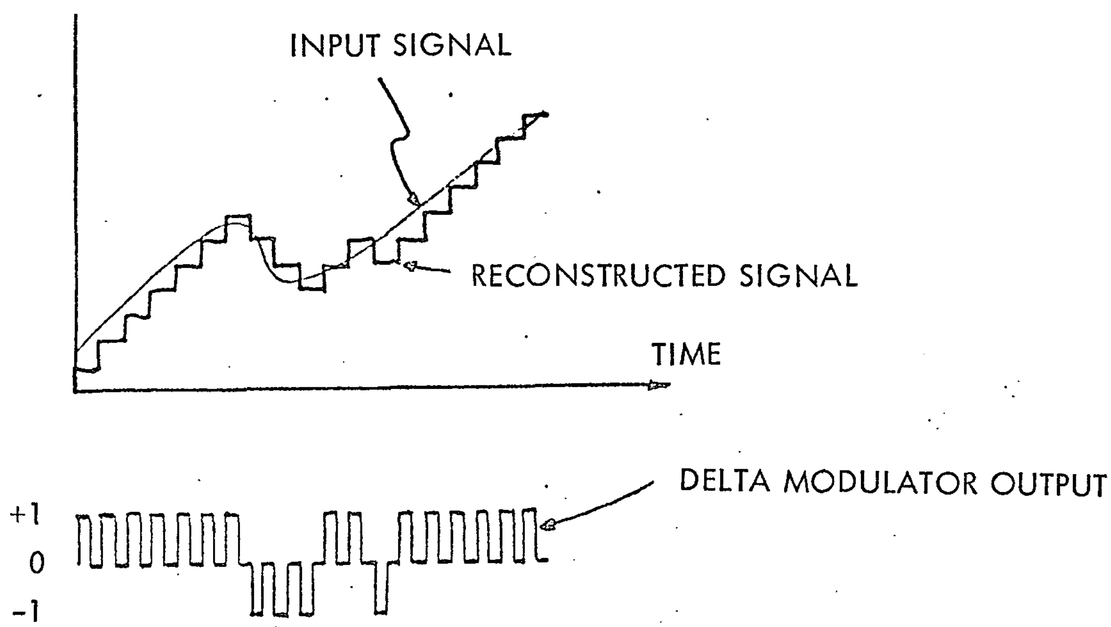
where $\delta(t)$ is a finite pulse of width PW

The above process is clocked at a repetition rate of PPS

MODEL	GROUP ID	PAGE	DATE
DELTA MODULATION	Modulator	2	April, 1972



EXAMPLE:



The width of the pulses in the Delta Modulator output is PW. There is one positive or negative pulse every 1/PPS. The input signal and the signal which can be reconstructed from the Delta Modulator output have the same scale. Each step is 1/PPS wide and PW high.

RESTRICTIONS

In general, the delta modulator cannot follow the input signal whenever:

$$\left| \frac{d}{dt} \text{ input(t)} \right| > PW*PPS$$

To prevent overload, PW and PPS should be adjusted so that:

$$\left| \frac{d}{dt} \text{ input(t)} \right|_{max} \leq PW*PPS$$

which is usually satisfied if:

$$A\omega \leq PW*PPS$$

where: A = Peak value of input

ω = Maximum frequency of input

If PW and PPS cannot be adjusted to meet this condition (note PWS≤1/PPS), then the input to the delta modulator must be properly scaled.

In the case where:

$$\left| \frac{d}{dt} \text{ input(t)} \right|_{max} << PW*PPS$$

PW may be decreased or the input signal scaled to provide more information in the delta modulator output.

APPLICATION

The delta modulator is normally used in pulse coding an audio signal for subsequent transmission via a modulated carrier. As such, this model will be mainly used in generating a baseband signal.

REFERENCE

For a description of delta modulation and a bibliography, see:

H. R. SCHINDLER, "Delta Modulation", IEEE SPECTRUM, October 1970, pp. 69-78

MODEL	GROUP ID	PAGE	DATE
AMPLITUDE DEMODULATOR	Demod.	1	April, 1972

LIBRARY MODEL NAMES

AMPLITUDE DEMODULATOR
AMDEM

DESCRIPTION

This linear Amplitude Demodulator provides a rudimentary model for use in the ASYSTD library. The basic model is a full wave rectifier followed by a user selected filter function chosen for the particular application.

USAGE

N1 < AMDEM > N2

NOTE: N2 must be the input to a filter.

OUTPUT

The output signal at N2 is simply the absolute value of the signal at N1

$$N2(T) = |N1(T)|$$

APPLICATION

This model is for use in the baseband region. Its output must be fed through an averaging filter to eliminate the carrier.

MODEL	GROUP ID	PAGE	DATE
RF AMPLITUDE DEMODULATOR	Demod.	1	April, 1972

LIBRARY MODEL NAMES

RF AM DEMODULATOR
RAMDEM

DESCRIPTION

This model is a rudimentary Amplitude Demodulator for use in modeling in the RF region.

USAGE

N1 < RAMDEM(GAIN) > N2

where: GAIN = Amplification of the model (typically 1/BETA, see RF AMPLITUDE MODULATOR)

OUTPUT

Let X be the complex signal at N1

$$X = X_r + jXi$$

$$|X| = \left(X_r^2 + X_i^2\right)^{1/2}$$

Let Y be the output at N2

$$Y = GAIN * \left(|X| - 1.\right)$$

APPLICATION

This model should be followed with a user selected filter for accurate simulation.

MODEL	GROUP ID	PAGE	DATE
PHASE DEMODULATOR	Demod.	1	April, 1972

LIBRARY MODEL NAMES

PHASE DEMODULATOR
PMDEMM

DESCRIPTION

This Phase Demodulator simply takes the integral of an FM demodulator output.

USAGE

N1 < PMDEMM(DV, FC) > N2

where: FC = Center Frequency

DV = Output Magnitude per Unit Phase Deviation (Volts/Radians)

OUTPUT

The Phase Demodulator output is given by DV $*\int$ FMDEMOD(t) dt where FMDEMOD(t) is the output of an FMDEMOD with a sensitivity of 1v/radian.

APPLICATION

This model is to be used with external filtering.

MODEL	GROUP ID	PAGE	DATE
RF PHASE DEMODULATOR	Demod.	1	April, 1972

LIBRARY MODEL NAMES

RF PHASE DEMODULATOR
RFPDEM

DESCRIPTION

This model represents an ideal wide band phase demodulator.

USAGE

N1 < RFPDEM(DV) > N2

where: DV = Output Magnitude per Unit Phase Deviation (Volts/Radians)

OUTPUT

Let the complex input at N1 be $X = X_r + jX_i$

Then the output of this model is $DV*TAN^{-1}\left(\frac{Xi}{X_r}\right)$

APPLICATION

This model is to be used with external filtering.

MODEL	GROUP ID	PAGE	DATE
FREQUENCY DEMODULATOR WITH FEEDBACK	Demod.	1	April, 1972

LIBRARY MODEL NAMES

FMFB

DESCRIPTION

The Frequency Demodulator with Feedback Model (FMFB) provides an alternate demodulation process capability. The basic model consists of a multiplier, IF filter, FM discriminator (FMDEMOD) and a Voltage Controlled Oscillator (FMMOD). The RF filter and post detection low pass filter are external to the model.

USAGE

N1 <(FMFB (NIF,NTYPE,AR,EM,BIF,FIF,GAIN,FC,DV,DF)> N2

where: NIF -IF Filter Order (≤ 10)

NTYPE -Type of Filter Function:
- = 1 for Butterworth
- = 2 for Chebyshev
- = 3 for Bessel
- = 4 for Butterworth-Thomson
- = 5 for Elliptic

AR -Amplitude Ripple (dB)

EM -M-Factor for Butterworth-Thomson
Stop Band Ratio for Elliptic (if positive)
Modular Angle (Degrees) for Elliptic (if negative)

BIF -IF Filter Bandwidth

GAIN -Detector Gain + VCO Amp Gain

FIF -IF Frequency

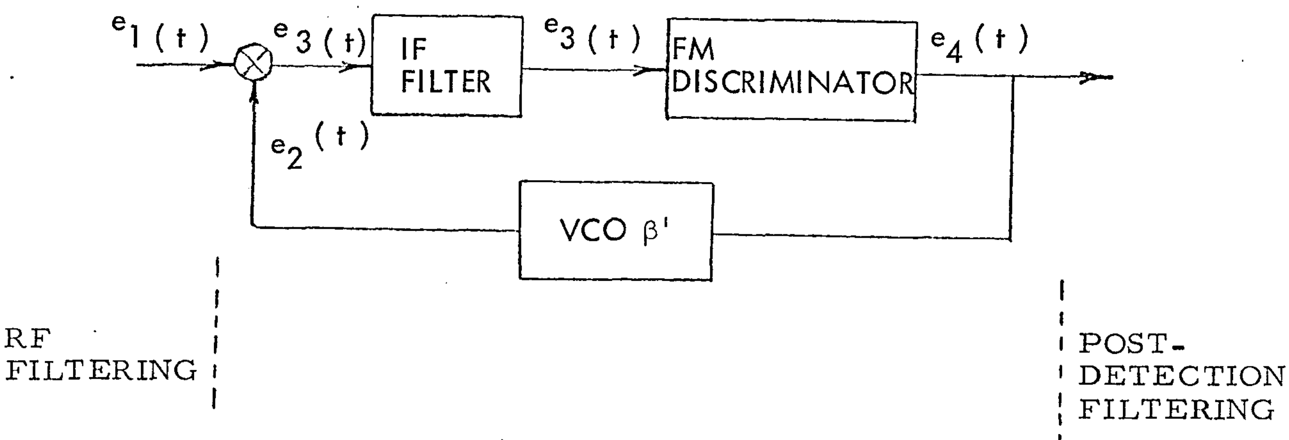
FC -Carrier Frequency

DV -FM Discriminator Constant (Volts/Hz)

DF -VCO Deviation (e.g., Hz/Volts)

NOTE: N1 is the output of an RF filter and N2 is the input to a low pass filter.

DETAILED DESCRIPTION



Let

$$e_1(t) = A(t) \cos (\omega_{\not{c}} t + \phi_i(t))$$

and

$$e_2(t) = -B \sin (\omega_{VCO} t + \theta(t))$$

where

$$\phi_i(t) = \beta \int S(t)$$

$$\theta(t) = \beta' \int e_4(t)$$

and

$$e_3'(t) = \frac{A(t)B}{2} \Big\{ \sin[\omega_{IF} t + \phi_i(t) - \theta(t)] - \sin[(\omega_{IF} + \omega_{\not{c}})t + \phi(t) + \theta(t)] \Big\}$$

assuming the IF filter passes only the first term

$$e_3'(t) = \frac{A(t)B}{2} \sin\left(\omega_{IF}t + \phi_i(t) - \theta(t)\right)$$

the output of the FM discriminator is ideally

$$e_4(t) = \frac{d}{dt}\left[\phi_i(t) - \theta(t)\right] DV = DV\left[\beta S(t) - \beta' e_4(t)\right]$$

$$e_4(t) = DV \frac{\beta}{1 + \beta'} S(t)$$

APPLICATION

The FMFB demodulator utilizes a tracking principle to achieve good SNR performance.

MODEL	GROUP ID	PAGE	DATE
FILTER	Filter	1	April, 1972

LIBRARY MODEL NAMES

FILTER

DESCRIPTION

The modeling of filters, or continuous functions relies on several computer routines previously developed by SAI which perform the various functions described in Appendices A and C. For ease in their use, an interface routine is written called FILTER, with several entry points as is explained below.

When utilizing any Filter model in a simulation of an RF link, translation of the filter to the baseband region is necessary for efficient simulation. The translation parameters are reflected in the reference to the Filter model.

DETAILED DESCRIPTION

The detailed description for generating the various filter functions in the s domain is described in Appendix A. Once the function of s is known, the bilinear z transform is derived. In order to reduce round-off errors, the function is represented by second degree sections, or quadratic factors. The bilinear z-transform converts a factor of s to a factor of the same degree in z, that is:

$$\frac{O(s)}{I(s)} = \frac{a_2 s^2 + a_1 s + a_o}{b_2 s^2 + b_2 s + b_o} \Bigg| \frac{F_2 z^{-2} + F_1 z^{-1} + F_o}{D_2 z^{-2} + D_1 z^{-1} + D_o} = \frac{O(z)}{I(z)}$$

NOTE: D_o is normalized to entry

z^{-1} is a unit delay

The difference equation for one of the quadratic factors will then be:

$$O(t) = F_2 I(t - 2DT) + F_1 I(t - DT) + F_o I(t)$$

$$- D_2 O(t - 2\,DT) - D_1 O(t - DT)$$

where: DT is the sampling time.

If the filter is not being translated, the quadratic factors are cascaded. However, when translating the filter (described in Appendix A), both real and imaginary coefficients of s result and the function is represented by parallel quadratic factors. The representation of the function as a sum of terms rather than a product eliminates the necessity of computing the roots of a polynominal in determining $K_r(s)$ and $K_i(s)$, (see Appendix A).

When using a translated filter, the run time can be reduced significantly in trade for exact representation of the filter. Reduction of approximately one-fourth is realized by using an equivalent low pass function; or one-half by assuming symmetry of the filter (i.e., $K_i(s) = 0$). This is accomplished when referencing one of the functions as described below.

USAGE

N1 <FILTER(NP, IF, IG, FX, BW, FC, AMP, AR, EM)> N2

All variables must be included whether they are applicable or not.

where: NP = filter order

IG = filter geometry
- = 1 for Low Pass
- = 2 for High Pass
- = 3 for Band Pass
- = 4 for Band Stop

AR = amplitude ripple (dB)

EM = M-factor for Butterworth-Thomson or stop-band ratio (>0) or modulator angle (<0) for Elliptic functions

FX = arithmetic center frequency

BW = bandwidth

FC = translation frequency (i.e., translate such that FC becomes zero)

AMP = voltage gain at FX

IF = filter function
= 1 for Butterworth
= 2 for Chebyshev
= 3 for Bessel
= 4 for Butterworth-Thomson
= 5 for Elliptic

Alternate references to filters are as follows:

BUTTERWORTH (NP, IG, FX, BW, FC, AMP)

CHEBYSHEV (NP, IG, FX, BW, FC, AMP, AR)

BESSEL (NP, IG, FX, BW, FC, AMP)

BUTTERWORTH THOMSON (NP, IG, FX, BW, FC,-AMP, EM)

ELLIPTIC (NP, IG, FX, BW, FC, AMP, AR, EM)

Special Cases:

a) A model is available to characterize a filter from frequency response data (see GENERAL FILTER).

b) QFACTOR (AMP, A1, A2, A3, A4, A5, A6)

used to describe:

$$AMP * \frac{A1s^2 + A2s + A3}{A4s^2 + A5s + A6}$$

c) LEADLAG (AMP, F1, F2, F3, F4)

used to describe:

$$AMP * \frac{\left(\frac{s}{2\ F1} + 1\right)\left(\frac{s}{2\ F2} + 1\right)}{\left(\frac{s}{2\ F3} + 1\right)\left(\frac{s}{2\ F4} + 1\right)}$$

if:

F2 = 0 then one zero is eliminated

F1 = 0 then both zeros are eliminated

F4 = 0 then one pole is eliminated

F3 = 0 then both poles are eliminated

d) LEAD FUNCTION (AMP, F1, F2, F3)

used to describe:

$$AMP * \frac{\left(\frac{s}{2\ F1} + 1\right)\left(\frac{s}{2\ F2} + 1\right)}{\left(s\frac{s}{2\ F3} + 1\right)}$$

and is otherwise the same as the LEADLAG function.

APPLICATIONS AND RESTRICTIONS

When utilizing the above function, any time FC>0, an RF filter is referenced (i.e., complex inputs and outputs) rather than a baseband filter (i.e., real inputs and outputs). The following table describes the conditions set up by FC and FX.

FC	FX	IG	Result
0	-	-	Baseband filter simulation
>0	>0	3	RF translated filter
>0	<0	3	Symmetric translated filter (Q = ∞)
>0	0	? 1	Equivalent low pass function

MODEL	GROUP ID	PAGE	DATE
BUTTERWORTH FILTER	Filter	1	April, 1972

LIBRARY MODEL NAMES

BUTTERWORTH
BUTWTH
BUFUNCTION

DESCRIPTION

For a description of Butterworth filters, see Appendix C, Section C.2.2.

USAGE

N1 <BUTTERWORTH(NP, IG, FX, BW, FC, AMP)> N2

- NP = filter order
- IG = filter geometry
  - = 1 for Low Pass
  - = 2 for High Pass
  - = 3 for Band Pass
  - = 4 for Band Stop
- FX = arithmetic center frequency
- BW = bandwidth
- FC = translation frequency (i.e., translate such that FC becomes zero)
- AMP = voltage gain at FX

APPLICATION

For applications and restrictions in using this model, see the discussion on the model FILTER.

MODEL	GROUP ID	PAGE	DATE
CHEBYSHEV FILTER	Filter	1	April, 1972

LIBRARY MODEL NAMES

CHEBYSHEV
CHEBY
TCHEBYCHEFF

DESCRIPTION

For a description of Chebyshev filters, see Appendix C, Section C.2.3.

USAGE

N1 <CHEBYSHEV(NP, IG, FX, BW, FC, AMP, AR)> N2

where: NP = filter order

IG = filter geometry
= 1 for Low Pass
= 2 for High Pass
= 3 for Band Pass
= 4 for Band Stop

AR = amplitude ripple (dB)

FX = arithmetic center frequency

BW = bandwidth

FC = translation frequency (i.e., translate such that FC becomes zero)

AMP = voltage gain at FX

APPLICATION

For applications and restrictions in using this model, see the discussion on the model FILTER.

MODEL	GROUP ID	PAGE	DATE
BESSEL FILTER	Filter	1	April, 1972

LIBRARY MODEL NAMES

BESSEL
BE FUNCTION

DESCRIPTION

For a description of Bessel filters, see Appendix C, Section C.2.4.

USAGE

N1 <BESSEL(NP, IG, FX, BW, FC, AMP)> N2

NP = filter order

IG = filter geometry
 = 1 for Low Pass
 = 2 for High Pass
 = 3 for Band Pass
 = 4 for Band Stop

FX = arithmetic center frequency

BW = bandwidth

FC = translation frequency (i.e., translate such that FC becomes zero)

AMP = voltage gain at FX

APPLICATION

For applications and restrictions in using this model, see the discussion on the model FILTER.

MODEL	GROUP ID	PAGE	DATE
BUTTERWORTH THOMSON FILTER	Filter	1	April, 1972

LIBRARY MODEL NAMES

BUTTERWORTH THOMSON
BT FUNCTION
BUTOM

DESCRIPTION

For a description of Butterworth Thomson filters, see Appendix C, Section C.2.5.

USAGE

N1 <BUTTERWORTH THOMSON(NP, IG, FX, BW, FC, AMP, EM)> N2

NP = filter order
IG = filter geometry
= 1 for Low Pass
= 2 for High Pass
= 3 for Band Pass
= 4 for Band Stop
FX = arithmetic center frequency
BW = bandwidth
FC = translation frequency (i.e., translate such that FC becomes zero)
AMP = voltage gain at FX
EM = M-factor

APPLICATION

For applications and restrictions in using this model, see the discussion on the model FILTER.

MODEL	GROUP ID	PAGE	DATE
ELLIPTIC FUNCTION FILTER	Filter	1	April, 1972

LIBRARY MODEL NAMES

ELLIPTIC
ELIPTC
EL FUNCTION

DESCRIPTION

For a description of Elliptic Function Filters, see Appendix C, Section C.2.6.

USAGE

N1 < ELLIPTIC(NP, IG, FX, BW, FC, AMP, AR, EM) > N2

where: NP = filter order

IG = filter geometry
= 1 for Low Pass
= 2 for High Pass
= 3 for Band Pass
= 4 for Band Stop

FX = arithmetic center frequency

BW = bandwidth

FC = translation frequency (i.e., translate such that FC becomes zero)

AMP = voltage gain at FX

AR = amplitude ripple (dB)

EM = stop-band ratio if positive modular angle if negative

APPLICATION

For applications and restrictions in using this model, see the discussion on the model FILTER.

MODEL	GROUP ID	PAGE	DATE
GENERAL FILTER	Filter	1	April, 1972

LIBRARY MODEL NAMES

GENERAL FILTER
GENRAL

DESCRIPTION

The General Filter model determines an arbitrary element transfer function by the complex curve fitting method. The procedure is entirely analogous to familiar curve fitting techniques except that the quantities involved are complex numbers. It requires an assumption on the part of the user as to the number of poles and zeros which characterize the transfer function of the element in question. From an appropriate number of representative amplitude and phase response samples over the frequency range of interest, the values of these complex poles and zeros which characterize the element transfer function may then be determined.

DETAILED DESCRIPTION

(See also Appendix C)

A generalized element transfer function H(s) is customarily written in the form:

$$H(s) = \frac{A \prod_{i=1}^{NZ} (s - Z_i)}{\prod_{i=1}^{NP} (s - P_i)} \qquad (1)$$

where:
- s = $j\omega$, complex frequency
- P_i = complex pole $(s + j\omega_i)$
- Z_i = complex zero $(s + j\omega_i)$
- NP = number of poles (also filter order)
- NZ = number of zeros
- A = multiplicative real constant

The poles and zeros are always either complex conjugate pairs or single real values.

For each response sample at frequency ω_k the poles and zeros are related to the empirically determined amplitude and phase through

$$\frac{A \prod_{i=1}^{NZ} (j\omega_k - Z_i)}{\prod_{i=1}^{NP} (j\omega_k - P_i)} = \alpha_k + j\beta_k \qquad (2)$$

where

$$\text{Magnitude response} = \sqrt{\alpha^2 + \beta^2}$$

$$\text{Phase response} = \tan^{-1} \beta/\alpha$$

Hence, for each response sample point, the right side of Equation (2) is determined and the left side is a complex expression in the P_i's and Z_i's. If the form of H(s) is assumed by specifying the number of poles and zeros, the P_i's, Z_i's, and A are determined by the set of NP+NZ+1 such complex equations which result from the substitution of corresponding phase and amplitude response samples at a like number of frequencies. The actual solution of this system of complex linear equations is effected by standard library subroutines.

USAGE

N1 <GENRAL(NP,NZ,IDB,FC,IG,BW,AMP)> N2

where: NP = assumed number of poles

NZ = assumed number of zeros

IDB = 1 if response amplitude data is to be in dB

= 0 if response amplitude data is normalized to 1.0

FC = translation frequency (i.e., translate such that FC becomes zero)

IG = filter geometry
= 1 for Low Pass
= 2 for High Pass
= 3 for Band Pass
= 4 for Band Stop

BW = bandwidth

AMP = voltage gain (ratio) midway between the upper and lower cutoff frequencies

INPUT

The number of frequency response data points should equal NP+NZ+1.

Input frequency response data is read in through the subroutine POLZER, one card per response data point, in the order: Frequency (Hz), Amplitude (dB or relative magnitude), Phase (degrees). The input data format is (1PE15.4, OP2F15.4). An input data set for a general 3rd order filter with two zeros (NP+NZ+1 = 6) is as follows:

FREQUENCY	AMPLITUDE	PHASE
1.4320-05	.0000	-10.3210
4.7750-05	-.0030	-34.9240
1.0980-04	-.4420	-87.3390
1.6710-04	-3.6820	-141.9130
2.8970-04	-15.7100	157.0050
3.3740-04	-19.6160	146.5060

RC

MODEL	GROUP ID	PAGE	DATE
QUADRATIC FACTOR	Filter	1	April, 1972

LIBRARY MODEL NAMES

QFACT
QFACTOR
QUADRADIC FACTOR

DESCRIPTION

This model is used to describe the transfer function:

$$H(s) = AMP * \frac{A1s^2 + A2s + A3}{A4s^2 + A5s + A6}$$

(see Appendix C, Section C.1)

USAGE

N1 <QFACTOR(AMP, A1, A2, A3, A4, A5, A6)> N2

where: AMP = amplification constant

A1-A6 = constants specifying the transfer function

MODEL	GROUP ID	PAGE	DATE
LEAD LAG	Filter	1	April, 1972

LIBRARY MODEL NAMES

LEAD LAG
LOOP FILTER

DESCRIPTION

This model is used to describe the transfer function:

$$H(s) = AMP * \frac{\left(\frac{s}{2\pi F1} + 1\right)\left(\frac{s}{2\pi F2} + 1\right)}{\left(\frac{s}{2\pi F3} + 1\right)\left(\frac{s}{2\pi F4} + 1\right)}$$

(see Appendix C, Section C.1)

USAGE

N1 <LEADLAG(AMP, F1, F2, F3, F4)> N2

where: AMP = amplification constant

F1-F4 = constants specifying the transfer function, and if:

F2 = 0 then one zero is eliminated

F1 = 0 then both zeros are eliminated

F4 = 0 then one pole is eliminated

F3 = 0 then both poles are eliminated

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE
LEAD FUNCTION	Filter	1	April, 1972

LIBRARY MODEL NAMES

LEAD FUNCTION

DESCRIPTION

This model is used to describe the transfer function

$$H(s) \ = \ AMP * \frac{\left(\frac{s}{2\ F1} + 1\right)\left(\frac{s}{2\ F2} + 1\right)}{\left(s\frac{s}{2\ F3} + 1\right)}$$

(see Appendix C, Section C. 1)

USAGE

N1 < LEAD FUNCTION(AMP, F1, F2, F3) > N2

where: AMP = amplification constant

F1-F3 = constants specifying the transfer function, and if:

F2 = 0 then one zero is eliminated

F1 = 0 then both zeros are eliminated

F3 = 0 the pole is eliminated

MODEL	GROUP ID	PAGE	DATE
MATCHED FILTER	Filter	1	April, 1972

LIBRARY MODEL NAMES

MATCHED FILTER
MFLTER

DESCRIPTION

The Matched Filter model is a simple integrate and dump routine clocked to the Bit Time.

USAGE

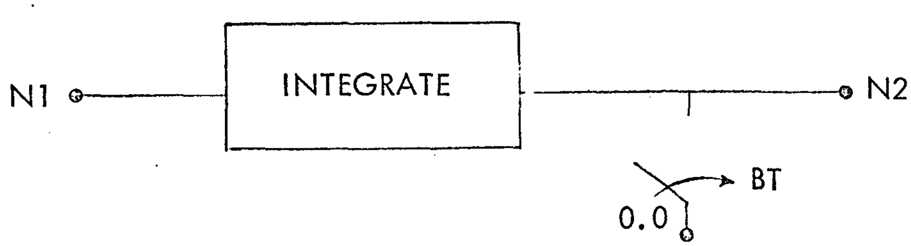
N1 <MFLTER(BT)> N2

where: BT = bit time

OUTPUT

The output at N2 is the trapezoidal approximation of the integral of the signal at N1. Every BT, this integral is reset to zero.

BLOCK DIAGRAM



RESTRICTIONS

The integrate-Dump process is asynchronous and starts at time equal to zero, requiring the user's discretion in its use.

MODEL	GROUP ID	PAGE	DATE
SOFT LIMITER	Limiter	1	April, 1972

LIBRARY MODEL NAMES

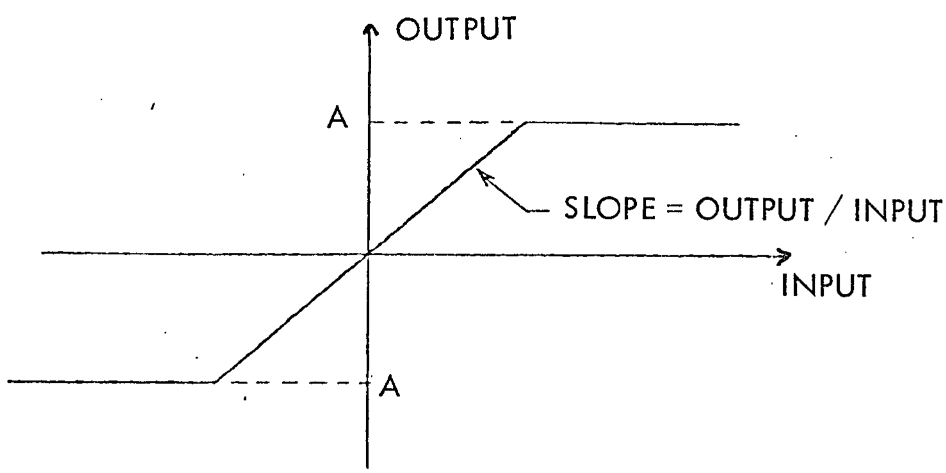
SOFT LIMITER
SOFTY

USAGE

N1 <SOFTY(A,SLOPE)> N2

where: A = maximum amplitude of output
SLOPE = slope of transition limit

OUTPUT



APPLICATION

This element is for use in baseband modeling.

MODEL	GROUP ID	PAGE	DATE
HARD LIMITER	Limiter	1	April, 1972

LIBRARY MODEL NAMES

HARD LIMITER
HARD

USAGE

N1 <HARD> N2

OUTPUT

The absolute value of the output at N2 is 1. The sign of the output is the same as the input at N1.

APPLICATION

This element is for use in baseband modeling.

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE
RF SOFT LIMITER	Limiter	1	April, 1972

LIBRARY MODEL NAMES

RF SOFT LIMITER
RFSOFT

USAGE

N1 <RFSOFT(A,SLOPE)> N2

where: A = maximum amplitude of output
SLOPE = slope of transition limit

OUTPUT

Let $X = X_r + jX_i$ be the complex signal at N1
and Let $Y = Y_r + jX_i$ be the complex signal at N2

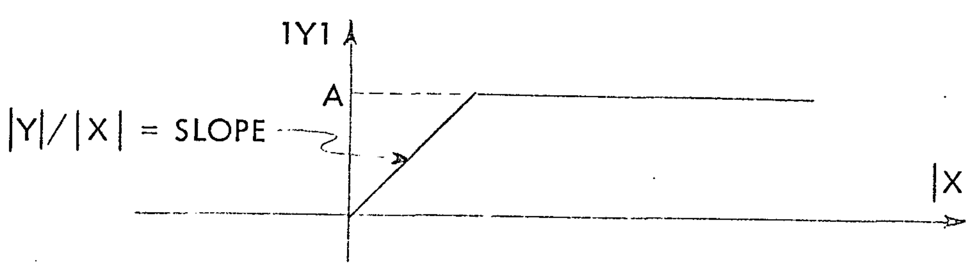
$$|X| = (X_r{}^2 + X_i^2)^{1/2} = \text{magnitude of } X$$

$$\varnothing(X) = \tan^{-1}(X_i/X_r) = \text{phase of } X$$

Then the output at N2 is such that:

1) $\varnothing(Y) = \varnothing(X)$

2) $|Y| = F(|X|)$ as described by the graph below:



APPLICATION

This element is for modeling in the RF region.

MODEL	GROUP ID	PAGE	DATE
RF HARD LIMITER	Limiter	1	April, 1972

LIBRARY MODEL NAMES

RF LIMITER
RFLIMT

USAGE

N1 <RF LIMITER> N2

OUTPUT

Assume the input at N1 is described by:

$$X = X_r + jX_i = \left|X_r^2 + X_i^2\right|^{1/2} e^{j \tan^{-1}(X_i/X_r)}$$

then the output at N2 is:

$$Y = Y_r + jY_i = C e^{j \tan^{-1}(Y_i/Y_r)}$$

where:

$$Y_r = \frac{CX_r}{\left|X_r^2 + X_i^2\right|^{1/2}}$$

$$Y_i = \frac{CX_i}{\left|X_r^2 + X_i^2\right|^{1/2}}$$

In this model C = 1.

In other words, the phase of the signal is maintained but the magnitude is set to 1.

APPLICATION

The element is for use in modeling in the RF region.

MODEL	GROUP ID	PAGE	DATE
FOURIER TRANSFORM	Transforms	1	April, 1972

LIBRARY MODEL NAMES

FOURT FOURIER	IFOURT INVERSE FOURIER

DESCRIPTION

This model samples an input signal and calculates the Discrete Fourier Transform of these samples. Its output is the coefficients of this transform. IFOURT performs the inverse transform, reconstructing the input signal to FOURT.

USAGE

N1 <FOURT(N, TW)> N2

or

N1 <IFOURT(N, TW)> N2

where: N = number of samples per transform

TW = time window per transform

NOTE: Although N can be any positive integer, the transform is much faster if all the prime factors of N are 2, 3 or 5.

NOTE: The actual time window used is the multiple of N*DT nearest TW.

OUTPUT

Each TW, this model takes N samples of the input at N1, and calculates the Discrete Fourier Transform (or the inverse of the transform). The output of the model during this TW is the N transformed values produced during the previous TW.

NOTE: The input and the output of this model are complex.

REFERENCE

For a discussion of this transform, see COMSAT Laboratories, "Orthogonal Transform Feasibility Study, Final Report", Contract NAS9-11240, November 1971, Section 3.2.

APPLICATION

The models INTLEV and DELEAV have been provided to deal with the complex inputs and outputs of this model. During the first TW, FOURT outputs a sync pulse of all ones to lock in DELEAV and IFOURT, compensating for any delays in the system.

MODEL	GROUP ID	PAGE	DATE
HAAR TRANSFORM	Transforms	1	April, 1972

LIBRARY MODEL NAMES

HAAR	IHAAR INVERSE HAAR

DESCRIPTION

This model samples a signal and outputs the HAAR Transform of these samples. (IHAAR-Inverse HAAR Transform - reconstructs the signal given the transformed values.)

USAGE

N1 <HAAR(N, LOG2N, TS)> N2

or N1 <IHAAR(N, LOG2N, TS)> N2 for the inverse

where: N = number of samples per transform (N must be a power of 2)

LOG2N = log base 2 of N, $N = 2^{LOG2N}$

TS = time window per transform

NOTE: The actual time window used is the integer multiple of N*DT nearest TS.

OUTPUT

Each TS, this model takes N samples of its input, and takes the transform of these samples (or the inverse transform), producing N transformed values. The output of this model during this time window is the N transformed values calculated during the previous TS.

REFERENCE

For a discussion of HAAR functions and the HAAR Transform, see COMSAT Laboratories, "Orthogonal Transform Feasibility Study, Final Report", Contract NAS9-11240, November 1971, Section 3.4.

MODEL	GROUP ID	PAGE	DATE
HAAR TRANSFORM	Transforms	2	April, 1972

APPLICATION

The output of the HAAR Transform during the first TS is +1. IHAAR uses this string of ones as a sync pulse to lock on to the transformed coefficients.

MODEL	GROUP ID	PAGE	DATE
HADAMARD TRANSFORM	Transform	1	April, 1972

LIBRARY MODEL NAMES

HDMRD HADAMARD	IHDMRD INVERSE HADAMARD

DESCRIPTION

This model samples a signal and outputs the HADAMARD Transform of these samples. (IHDMRD-Inverse HADAMARD Transform-reconstructs the signal given the transformed values.)

USAGE

N1 <HDMRD(N, LOG2N, TS)> N2

or N1 <IHDMRD(N, LOG2N, TS)> N2 for the inverse

where: N = number of samples per transform (N must be a power of 2)

LOG2N = log base 2 of N, $N = 2^{LOG2N}$

TS = time window per transform

NOTE: The actual time window used is the integer multiple of N*DT nearest TS.

OUTPUT

Each TS, this model takes N samples of its input, and calculates the transform (or the inverse transform) of these samples, producing N transformed values. The output during this time window is the N transformed values determined during the previous TS.

REFERENCE

For a discussion of this transform, see COMSAT Laboratories, "Orthogonal Transform Feasibility Study, Final Report", November 1971, Contract NAS9-11240, Section 3.3.

MODEL	GROUP ID	PAGE	DATE
HADAMARD TRANSFORM	Transform	2	April, 1972

APPLICATION

The output of the HDMRD Transform during the first TS is +1. IHDMRD uses this string of ones as a sync pulse to lock on to the transformed values.

MODEL	GROUP ID	PAGE	DATE
ORDERED HADAMARD TRANSFORM	Transforms	1	April, 1972

LIBRARY MODEL NAMES

OHMRD ORDERED HADAMARD	IOHMRD INVERSE ORDERED HADAMARD

DESCRIPTION

This model samples the input signal and outputs the Ordered Hadamard Transform of these samples. (IOHMRD - Inverse Ordered Hadamard Transform - reconstructs the signal given the transformed values.

USAGE

N1 <OHMRD(N, LOG2N, TS)> N2

or

N1 <IOHMRD(N, LOG2N, TS)> N2 for the inverse

where: N = number of samples per transform (N must be a power of 2)

LOG2N = log base 2 of N, $N = 2^{LOG2N}$

TS = time window per transform

NOTE: The actual time window used is the integer multiple of N*DT nearest TS.

OUTPUT

Each TS, this model takes N samples of its input and calculates the transform (or the inverse transform) of these samples, producing N transformed values. The output during this time window is the N transformed values determined during the previous TS.

APPLICATION

The output of the OHMRD Transform during the first TS is +1. IOHMRD uses this string of ones as a sync pulse to lock on to the transformed values.

MODEL	GROUP ID	PAGE	DATE
ANALOG-TO-DIGITAL CONVERTER	Coders	1	April, 1972

LIBRARY MODEL NAMES

ATOD

DESCRIPTION

The A/D Converter model determines a bit sequence based upon the analog input level at sampling time. The output is a binary bit-stream (0 or +1). Flexibility is provided by inputting the number of bits per word, peak input level, minimum input level, and bit time.

USAGE

N1 <ATOD(NBIT, FLOOR, CEILING, BT)> N2

where:

- NBIT = number of digital bits per analog word
- FLOOR = a lower bound of the analog input
- CEILING = an upper bound of the analog input
- BT = bit time (the actual bit time is the multiple of DT closest to BT)

DETAILED DESCRIPTION

ATOD represents an analog value with a string of NBIT binary bits (0 or +1). This string of bits may be considered a binary number, B, whose value is within the range 0 (all 0 bits) to $2^{NBIT}-1$ (all one bits). FLOOR is presented by the value 0, and CEILING is represented by the value 2^{NBIT}. Each word time (NBIT*BT), ATOD constructs this number B such that the expression

$$FLOOR + B*\left((CEILING-FLOOR)/2^{NBIT}\right)$$

is as close to the analog input as the range and resolution allow.

MODEL	GROUP ID	PAGE	DATE
ANALOG-TO-DIGITAL CONVERTER	Coders	2	April, 1972

OUTPUT

Each BT, ATOD selects the next most significant bit of the digital word, to track the value of the analog input.

APPLICATION

ATOD may be used with either DTOA or SHDTOA as the decoder.

MODEL	GROUP ID	PAGE	DATE
DIGITAL-TO-ANALOG CONVERTER	Coders	1	April, 1972

LIBRARY MODEL NAMES

DTOA

DESCRIPTION

The D/A Converter determines an analog value from a binary bit stream (0 or +1) and the model parameters. This model assumes the output is continuous and attempts to produce output with as few abrupt changes as possible.

N1 < DTOA(NBIT, FLOOR, CEILING, BT) > N2

where:

NBIT	=	number of digital bits per analog word
FLOOR	=	a lower bound of the analog output
CEILING	=	an upper bound of the analog output
BT	=	bit time (the actual bit time is the multiple of DT closest to BT)

NOTE: These parameters are related to the A/D Converter which coded the signal.

OUTPUT

Each bit time, DTOA examines the new bit of the digital word. If, by some combination of remaining bits, the current analog output value can be represented, then the output remains the same. Otherwise, the output is adjusted up or down until it is within range. For a description of the relationship between the input bit stream and the analog value, see Analog to Digital Converter.

APPLICATION

Since this model tries to maintain the analog output at a constant level, only changing it when it has to, this model is best for use where the analog signal is continuous.

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE
SAMPLE-HOLD DIGITAL-TO-ANALOG CONVERTER	Coders	1	April, 1972

LIBRARY MODEL NAMES

SHDTOA

DESCRIPTION

The Sample-hold DIA Converter determines an analog value from a binary bit stream and the model parameters. This model determines a value during one word time and outputs that value during the entire next word time.

USAGE

N1 < SHDTOA(NBIT, FLOOR, CEILING, BT) > N2

where: NBIT = number of digital bits per analog word

FLOOR = a lower bound of the analog output

CEILING = an upper bound of the analog output

BT = bit time (the actual bit time is the multiple of DT closest to BT)

NOTE: These parameters are related to the A/D converter which coded the signal.

OUTPUT

Each digital word time (NBIT*BT), SHDTOA forms an analog value from the input and its parameters (for a description of the relationship between the input bit stream and the analog value, see Analog to Digital Converter). During this word time, its output is the analog value calculated during the previous word time.

APPLICATION

This model is for use when discrete values are being decoded, such as output from the orthogonal transform models.

MODEL	GROUP ID	PAGE	DATE
MULTI LEVEL PCM	Coders	1	April, 1972

LIBRARY MODEL NAMES

MULTI LEVEL PCM
MLTPCM

DESCRIPTION

This code modulator produces an m-level signal based upon a serial input bit stream (polar or binary).

USAGE

N1 <MLTPCM(BT,M)> N2

where: BT = bit time

M = number of levels (symbols)

N = LOG_2M

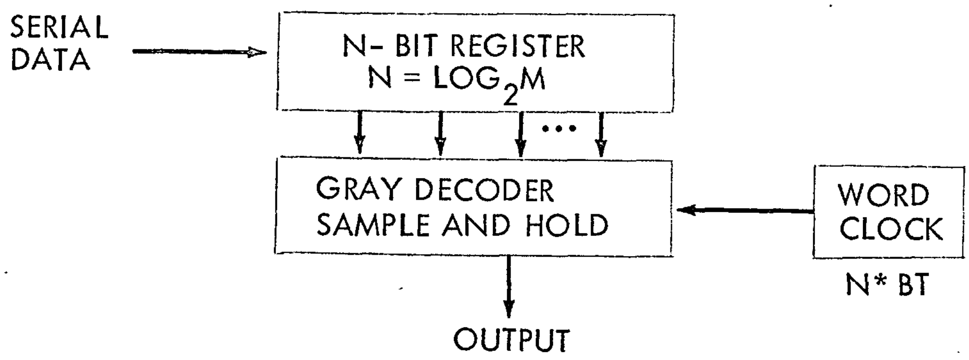
The signal at the input mode (e.g., N1) is assumed to be a polar or binary bit stream.

OUTPUT

A serial-to-parallel conversion is made and a gray code level selection is used in generating an output bit stream at a rate of N*BT. Output levels are normalized to a peak value of unity (+1), represented by equal levels separated by 1/(M-1). The minimum level (0) corresponds to the null symbol.

BLOCK DIAGRAM

Functionally, the model is represented as follows:

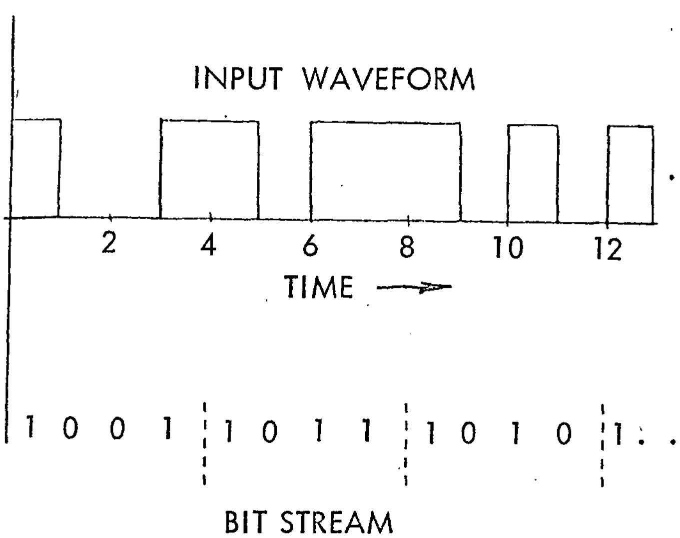


DETAILED DESCRIPTION (Example)

The serial bit stream is loaded into an N-bit register. At time intervals of N*BT, the register is sampled and the output waveform value (0 to +1) is determined from the reflected (gray) code in the register. The output is held at this level for N*BT, at which time the register is sampled for the next word. The following diagram depicts the time sequence for an arbitrary input bit stream. The parameters for this example are:

M = 16 (4 Bits)

BT = 1



Register History

Time	Contents 1234	Output
0	1---	0
1	10--	0
2	100-	0
3	1001	0
4	1---	14/15
5	10--	14/15
6	101-	14/15
7	1011	14/15
8	1---	13/15
9	10--	13/15
10	101-	13/15
11	1010	13/15
12	1---	12/15
13	10--	12/15
14	100-	12/15
15	1001	14/15
16	0---	14/15
17	00--	14/15
18	001-	14/15
19	0011	14/15
20		2/15

MODEL	GROUP ID	PAGE	DATE
MULTI LEVEL PCM	Coders	3	April, 1972

APPLICATION

The multi-level coder may be used to drive the FM and PM modulators to produce m-ary PM carrier signals. Attention should be made to appropriate scaling of the MLTPCM output to produce the correct modulating signal magnitudes.

MODEL	GROUP ID	PAGE	DATE
COMPLEX ADDER	MATH	1	April, 1972

LIBRARY MODEL NAMES

COMPLEX ADDER
CADD

DESCRIPTION

The complex input to this model is added to a specified complex number passed through the argument list.

USAGE

N1 < CADD(XREAL, XIMAGE) > N2

where: XREAL = the real part of the addend
XIMAGE = the imaginary part of the addend

OUTPUT

The complex output at N2 is the complex sum of (XREAL + i*XIMAGE) and the input at N1.

MODEL	GROUP ID	PAGE	DATE
COMPLEX MULTIPLIER	MATH	1	April, 1972

LIBRARY MODEL NAMES

COMPLEX MULTIPLIER
CMULT

DESCRIPTION

The complex input to this model is multiplied by a specified complex number passed through the argument list.

USAGE

N1 < CMULT(XREAL, XIMAGE) > N2

where: XREAL = the real part of the multiplier
XIMAGE = the imaginary part of the multiplier

OUTPUT

The complex output at N2 is the complex product of (XREAL + i*XIMAGE) times the input at N1.

MODEL	GROUP ID	PAGE	DATE
DIFFERENTIATOR	MATH	1	April, 1972

LIBRARY MODEL NAMES

DIFFERENTIATOR
DIFFER
DIF

DESCRIPTION

This model differentiates the input signal utilizing the bilinear z-transform of s.

USAGE

N1 < DIFFER > NZ

OUTPUT

Let the signal at N1 at time T be INP(T).
Let the signal at N2 at time T be OUT(T).

Then:

$$OUT(T) = \left[2*(INP(T)-INP(T-DT)/DT \right] - OUT(T-DT)$$

where OUT (TSTART-DT) = 0

With this scheme, the output may alternate wildly above and below the actual differential. The fluctuation is proportional to the derivative error at time = 0, since it is initialized to zero. If the output of the differentiator is fed into a filter, this fluctuation does not matter. This scheme has the advantage that if the output of the differentiator is fed into one of the system integrators, the input samples at N1 can be reconstructed exactly.

MODEL	GROUP ID	PAGE	DATE
INTEGRAL WITH INITIAL CONDITIONS	MATH	1	April, 1972

LIBRARY MODEL NAMES

INTEGRAL WITH INITIAL CONDITIONS
INTGIC

DESCRIPTION

This model provides for integration with initial conditions.

USAGE

N1 < INTGIC(FV) > N2

where: FV = initial value of the integral at TIME=TSTART

OUTPUT

This model integrates the input at N1 using the trapezoidal rule approximation. The output at N2 is this integral plus FV, the first value (initial value) of the integral.

MODEL	GROUP ID	PAGE	DATE
INTEGRATOR	MATH	1	April, 1972

LIBRARY MODEL NAMES

INTEGRATOR
INTGRT

DESCRIPTION

This model integrates the input signal using the trapezoidal rule, which is the bilinear z-transform of 1/s.

USAGE

N1 <INTGRT> NZ

OUTPUT

The output at N2 is the trapezoidal approximation of the input signal at N1.

MODEL	GROUP ID	PAGE	DATE
INTERLEAVER	MISC.	1	April, 1972

LIBRARY MODEL NAMES

INTLEV
INTERLEAVE

DESCRIPTION

The Interleaver allows complex values to be carried over a single line. During the first half of one sample time, the output of this model is the real part of the input. During the remainder of the sample time, the output is the imaginary part of the input.

USAGE

N1 < INTLEV(N, TW) > N2

where: N = number of samples per TW

TW = Time Window

NOTE: This model is usually used in conjunction with a model such as the Fourier Transform, where N and TW would be identical to the parameters of the Fourier Transform.

OUTPUT

Each sample time (TW/N), INTLEV samples the complex input and stores the real and imaginary parts. It them outputs the real part of the input for (TW/N)/2 and the imaginary part for the remainder of the sample time.

APPLICATION

This model was designed to appear after the Fourier Transform, or other models with complex outputs. It converts a complex valued signal to a real signal for use by other models which can only deal with real signals. See also DE-INTERLEAVER.

MODEL	GROUP ID	PAGE	DATE
DE-INTERLEAVER	MISC.	1	April, 1972

LIBRARY MODEL NAMES

DELEAV
DEINTERLEAVE

DESCRIPTION

The De-interleaver decoder real signals output by the Interleaver back into their complex form. It causes a phase delay of one sample time.

USAGE

N1 <DELEAV(N, TW)> N2

where: N = number of samples per TW
TW = time window

NOTE: These parameters should be the same as those in the Interleaver

OUTPUT

Each sample time (TW/N) this model samples the input for the real value. One half sample time later it gets the imaginary value from the input. The output of the model during this sample time is the complex value found in this way during the previous sample time.

APPLICATION

This model is used in conjunction with the Interleaver. Particularly to reconstruct complex values for input to the Inverse Fourier Transform. It is started by the sync pulse from the Fourier Transform.

MODEL	GROUP ID	PAGE	DATE
DELAY	MISC.	1	April, 1972

LIBRARY MODEL NAMES

DELAY

DESCRIPTION

This model introduces a specified time delay into a real signal.

USAGE

N1 < DELAY(LAG) > N2

where LAG = an integer number of DT's to delay the signal

OUTPUT

The output at N2 at time T equals the input at N1 at time T-LAG*DT.

MODEL	GROUP ID	PAGE	DATE
PHASE SHIFTER	MISC.	1	April, 1972

LIBRARY MODEL NAMES

PHASE SHIFTER
PHSHFT

DESCRIPTION

This model causes a phase shift (delay) of a specified number of degrees.

USAGE

N1 < PHSHFT(DGREES, FC) > NZ

where DGREES = number of degrees to delay signal

FC = frequency of input signal

MODEL	GROUP ID	PAGE	DATE
SPLIT	MISC.	1	April, 1972

LIBRARY MODEL NAMES

SPLIT

DESCRIPTION

The input signal is split into its real and imaginary parts.

USAGE

N1 < SPLIT > N2

OUTPUT

The real part of N2 is set to COS (N1). The imaginary part of N2 is set to SIN (N1).

SYSTEMS ASSOCIATES

MODEL	GROUP ID	PAGE	DATE
TIME LATCH	MISC.	1	April, 1972

LIBRARY MODEL NAMES

CALINB

DESCRIPTION

This model provides a time dependent latching function which sets the output equal to the input for time LAG*DT.

USAGE

N1 < CALINB (LAG) > N2

where LAG = an integer number of DT's before the signal can pass through this model.

OUTPUT

N2 = 0 for T LAG*DT

N2 = N1 for T LAG*DT

MODEL	GROUP ID	PAGE	DATE
ZERO CROSSING DETECTOR	MISC.	1	April, 1972

LIBRARY MODEL NAMES

ZERO CROSSING DETECTOR
ZRODET

DESCRIPTION

This model detects when the input signal becomes zero or crosses the zero reference.

USAGE

N1 <ZRODET> NZ

OUTPUT

N2 is generally set to zero. If, during one simulation step (DT) the signal at N1 goes to zero or crosses zero, N2 is set to 1 for that DT only.

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE

LIBRARY MODEL NAMES

SAI 72-001-1

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE

LIBRARY MODEL NAMES

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE

LIBRARY MODEL NAMES

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE

LIBRARY MODEL NAMES

SAI 72-001-1

SYSTEMS ASSOCIATES

ASYSTD LIBRARY

MODEL	GROUP ID	PAGE	DATE

LIBRARY MODEL NAMES

SAI 72-001-1

APPENDIX C

DISTRIBUTION LIST

NASA/MSC
Facility and Laboratory Support Branch
Houston, Texas 77058
Attention: Mr. Jerry Carlson
Mail Code BB321 (77)

1 Copy

NASA/MSC
Technical Library Branch
Houston, Texas 77058
Attention: Ms. Retha Shirkey
Mail Code JM6

4 Copies

NASA/MSC
Management Services Division
Houston, Texas 77058
Attention: Mr. John T. Wheeler
Mail Code JM7

1 Copy

NASA/MSC
Space Electronics System Division
Houston, Texas 77058
Attention: Mr. C. T. Dawson
Technical Monitor, EE8

4 Copies