AD-A066 806 BDM CORP ALBUQUERQUE N MEX F/G 20/14
EXEMPT USERS MANUAL.(U)
JAN 79 F29601-76-C-0122
UNCLASSIFIED BDM/A-76-140-TR-R2 AFWL-TR-77-207 NL

1 OF 2
AD A066806

AFWL-TR-77-207

AFWL-TR-77-207

ADE200262

(2) LEVEL III

EXEMPT USERS MANUAL

BDM Corporation
Albuquerque, NM 87106

January 1979

Final Report

Approved for public release; distribution unlimited.

AIR FORCE WEAPONS LABORATORY
Air Force Systems Command
Kirtland Air Force Base, NM 87117

AFWL-TR-77-207

This final report was prepared by the BDM Corporation, Albuquerque, New Mexico, under Contract F29601-76-C-0122, Job Order 37630113 with the Air Force Weapons Laboratory, Kirtland Air Force Base, New Mexico. Mr R. M. Pelzl (ELT) was the Project Officer-in-Charge.

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related Government procurement operation, the Government thereby incurs no responsibility nor any obligation whatsoever, and the fact that the Government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report has been authored by a contractor of the US Government. Accordingly, the US Government retains a nonexclusive royalty-free license to publish or reproduce the material contained herein, or allow others to do so, for the US Government purposes.

This report has been reviewed by the Office of Information (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

Robert M. Pelzl
ROBERT M. PELZL
Project Officer

FOR THE COMMANDER

J. Philip Castillo
J. PHILIP CASTILLO, PhD
Technology Branch

Donald A. Dowler
DONALD A. DOWLER
Colonel, USAF
Electromagnetics Division

DO NOT RETURN THIS COPY. RETAIN OR DESTROY.

(18) HFWL, SBIE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(12) **REPORT DOCUMENTATION PAGE**

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFWL-TR-77-207, AD-E200 262	2. GOVT ACCESSION NO. / 3. RECIPIENT'S CATALOG NUMBER
(6) 4. TITLE (and Subtitle): EXEMPT USERS MANUAL	(9) 5. TYPE OF REPORT & PERIOD COVERED: Final Report
	(14) 6. PERFORMING ORG. REPORT NUMBER: BDM/A-76-140-TR-R2
7. AUTHOR(s)	(15) 8. CONTRACT OR GRANT NUMBER(s): F29601-76-C-0122
9. PERFORMING ORGANIZATION NAME AND ADDRESS: BDM Corporation, 2600 Yale Blvd SE, Albuquerque NM 87106	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 64711F, 37630113, 14 01
(16) 11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Weapons Laboratory (ELT), Kirtland AFB, NM 87117	(11) 12. REPORT DATE: January 1979
	13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
(12) 104p.	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Aircraft susceptibility to EMP
System analysis of EMP effects on aircraft
EMP coupling to aircraft subsystems
Computer program modeling of aircraft response to EMP

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This volume describes the use of the EXEMPT computer program for the creation, maintenance and modification of a data base to support electromagnetic coupling analysis for linear systems. Use of the EXEMPT Command Language, FORTRAN Mathematical functions, EXEMPT library functions, and data display operations is presented. Interface requirements for user provided subroutines and other programs are defined.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

391884

79 03 22 067

PREFACE

This volume describes the capabilities and use of the EXEMPT code which function as the executive processor for SLAM and B-1AM or as a stand-alone processor for data base maintenance and data manipulation.

The BDM Program Manager was Mr R. J. Balestri and the Lead Programmer was Mr R. L. Tyler. Mr Robert Pelzl of AFWL/ELA was the COTR for this effort.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION I
INTRODUCTION

The purpose of this manual is to provide the EXEMPT user with an understanding of the capability of the EXEMPT code and the information necessary to correctly implement the program.

The remainder of this section discusses the EXEMPT code at a general level. Subsequent sections discuss the EXEMPT Command Language (ECL), the external file structures, the internal data structures, the user interface requirements, and the computer requirements.

1. FUNCTION OF EXEMPT

The function which the EXEMPT code is designed to perform is that of a data base interface processor. By manipulating the data provided by the user and generated internally, a number of mathematical operations may be performed on the data. The operations are commanded by the user and are performed in the sequence specified. The result of the operations will be the analysis desired on the system to which the data pertains. Therefore, EXEMPT stores and retrieves data, provides data arrays to user supplied programs and subroutines, performs algebraic operations on data, and will perform EXEMPT library functions on data.

2. DATA SET DESCRIPTION

The basic element of EXEMPT is the data set. This is a collection of numbers which have some physical or logical reason to be included in a set. In order to assure the integrity of the data and the sanity of an operation, a certain amount of information about the data set is required and will be referred to as data set attributes. The attributes of a data set are used to identify the type, source, length, and location of each data set referenced. The user may reference these data sets using symbolic names (limited to ten characters). When a reference is made to perform an operation on a data set, the attributes will be recovered to determine the correctness of the operation, auxiliary data

sets required, and the appropriate input/output (I/O) action to recover the data and store current data. EXEMPT will distinguish between three basic types of data sets. These are frequency domain, time domain, and raw or untyped data. Associated with frequency domain data are two subsets, the dependent or response data and the independent or frequency data. Similarly, a time domain data set will have dependent or response data and independent or time data. Raw data constitute a user provided array of data which is not a frequency or time domain data set. Raw data may be used in direct algebraic manipulations as an arithmetic operand but may not have frequency or time domain operations performed upon it. Data sets provided by the user from some external source must have the type specified. Data sets generated internally by operations on existing data sets will have the same type attributes as the existing or parent data sets as modified by the operation. For instance, if a data set is generated by performing a Fourier transform upon a time domain data set, the resultant data set will have the attributes of a frequency domain data set. However, in the direct point by point multiplication of two like data sets, or a raw and typed data set, the result will be a data set of the same type as involved in the operands. In order to avoid unnecessary storage, the independent array is not regenerated for the resultant array unless the operation alters the array contents. Instead, the resultant dependent data array is linked to the independent data array of its parents. This type of implementation results in EXEMPT being an attribute linked/set theoretic data base manipulator. As such, it is easily expanded to include other operations by providing the attributes required for the operations and the software to implement the operation.

SECTION II

EXEMPT COMMAND LANGUAGE

EXEMPT is a highly modularized code with distinct classes of functions performed in submodules. The top level structure of EXEMPT is shown in Figures 1 and 2. As illustrated, all codes required for data interfacing with system devices (tape files, disk files, and punch cards) are located in the main segment. The first level of modularization is represented by the Language Input Processor (LIP) and the Command Language Processor (CLP). These two modules execute independently of each other. The LIP is called to read the users ECL and generate CLP decodable tasks in the task table. After encountering an END or RUN command, control is returned to EXEMPT which subsequently causes the CLP to be loaded. The structure of the CLP is shown in Figure 2. There are three classes of operations illustrated. The logical operations are I/O oriented. They simply format and transfer data to the appropriate device. The SLAM operations generate and modify data as required by the particular submodel being used. The arithmetic operations are the primary operations within EXEMPT. These operations are employed in performing the analysis desired.

The following discussion of the ECL is presented by module and submodule. Each command is presented on a separate page to facilitate use of the manual and the continued development of EXEMPT. All commands are input in a free field format with the command name specified as required by the particular command.

The commands of the language are listed in Tables 1 and 2. Each command is separated by at least one blank from its parameter field and may start at any column in the scan field. The parameters for each command are separated by commas or blanks. There may be multiple commands on a single card. Multiple commands must be separated by a double slash (//).

Continuation is implied by a termination with a comma or an arithmetic operator (+, -, /, *, =). Comments are allowed by placing a $ in any column and the comment following the $. There is no continuation for comment cards.

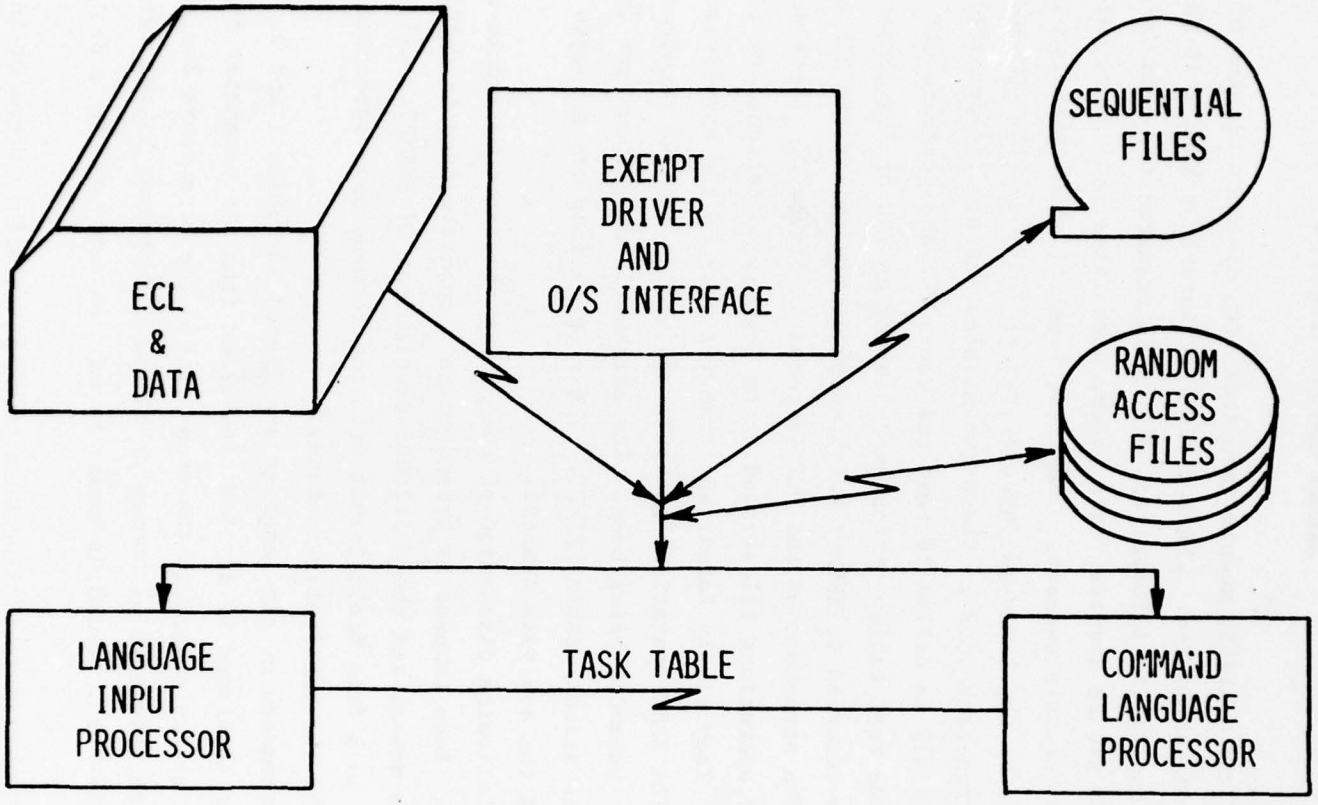


Figure 1. EXEMPT Program Structure

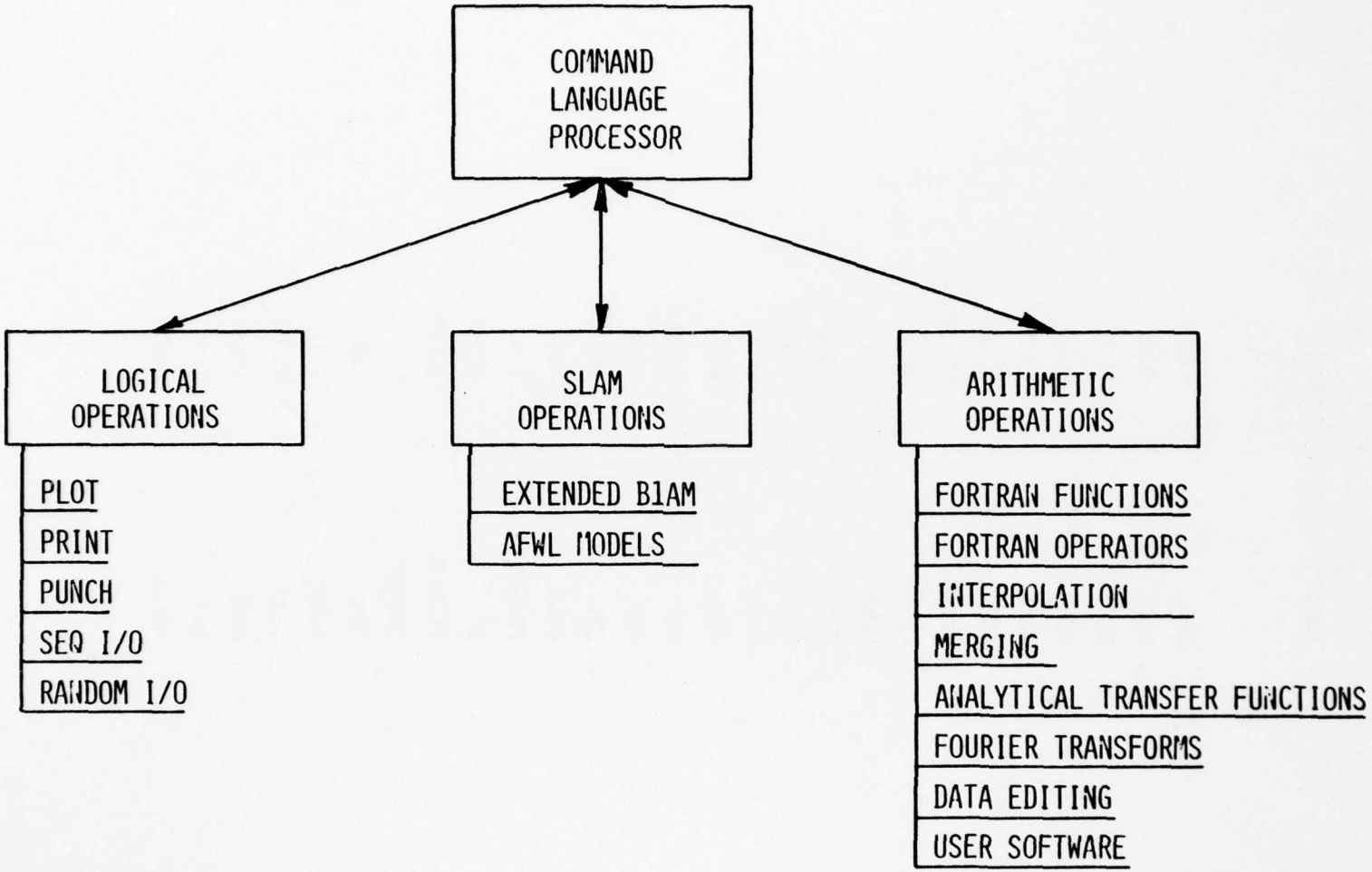


Figure 2. EXEMPT CLP Structure

Table 1

EXEMPT COMMAND KEYWORDS

ABS	MIN
ALOG	MOD
ALTFIL	PLOT
ATAN	POLY
ATAN2	PRTPLT
CALL	PUNCH
CLEAN	PURGE
CMPLX	READ
CONJG	READC
COS	READF
DELETE	REWIND
DROP	RUN
EDIT	SAVE
END	SAVRAN
EOF	SCAN
EXP	SHIFT
FILE	SIGN
FORMAT	SIN
FT	SKIPB
GETRAN	SKIPF
INTERP	SQRT
INVFT	TAN
LABEL	TITLE
LIST	TRDEXP
LOOP	TRDSIN
MAX	TYPE
MERGE	WRITE

Table 2

SLAM COMMAND KEYWORDS

AS1918	PITOT
AVBAY	PNLJNT
AVGSKT	RADALT
COAX	RADAR
COCKPT	RVBEA
CONDUIT	SFIELD
COUPLER	SHIELD
DBCAB	SNGLIN
DBCAB1	WEABAY
ENGINE	WEBAOP
HFANT	WROOT
INDCOU	WWELL
MARBEA	WWGND

The following character set is recognized by EXEMPT.

A - Z	All alpha characters
0 - 9	All number characters
+	Plus sign
-	Minus sign
*	Multiply sign
/	Divide sign
(	Left parenthesis
)	Right Parenthesis
$	Dollar sign
=	Equal sign
ß	Blank
,	Comma
.	Period

A number may be an integer, a real number, or a number in E format, and may contain a unary operator.

Defaults for missing parameters are described where applicable; otherwise, the parameter must be present.

The symbols on the commands are:

{ }	Multiple choice of required field
[ ]	Optional fields
Name	User defined symbolic name
N	Integer number
V,Value	Real number

1. LIP COMMANDS

In order to provide the user additional flexibility during LIP execution, four commands are recognizable by the LIP. These commands must be the only commands on the card; however, they may appear anywhere on the card. They are used by the prescanning program which collects all of the user's input (ECL and coded data) from the card input files and user specified alternate input file and rewrites the data onto a file to be read by the LIP. These commands will not be transferred to the CLP. The normal input file is FORTRAN logical unit 5 and the alternate file available is FORTRAN logical unit 4. If used, the user must attach this file as TAPE4 and it must be a coded card image file. EXEMPT reads all coded data from these files and writes the data on FORTRAN logical unit 3. Therefore, the user should avoid using a local file name of TAPE3 unless he desires to save the EXEMPT input data after initial processing.

a. Scan Limit Directive

SCAN [N1] [,N2]

This directive establishes the columns of the input cards to be searched for ECL directives. N1 specifies the first column and if omitted is assumed to be 1. N2 specifies the last column and is not defaulted. If a SCAN command is not provided, card columns 1-80 are used. If N1 is not provided, a comma must precede N2. All data preceding column N1 and following column N2 are ignored. If an END or RUN command is encountered, the scan limits will be reset to columns 1-80 in order to read data. Thus, a SCAN directive is required after each RUN command to reestablish the desired scan limits.

Examples:

Establish the scan limit of columns 1 to 50.

```
SCAN ,50
```

Establish the scan limit of columns 10 to 60 with RUN directive.

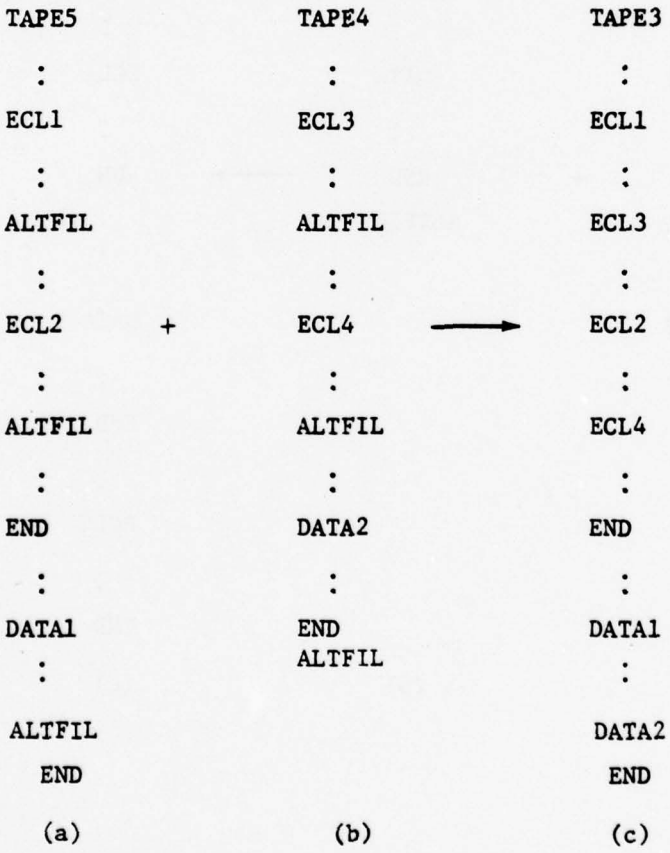
```
SCAN 10, 60
.
.
.
ECL
.
.
RUN
.
.
CARD DATA
.
.
END OF DATA
SCAN 10, 60
.
.
END OF ECL
```

b. Alternative File Directive

ALTFIL

This directive causes the scanner to read from the "other" card input file. For convenience, the user input files available are on SCOPE files TAPE4 and TAPE5. When ALTFIL is encountered in the ECL, subsequent input is read from TAPE $\begin{Bmatrix} 4 \\ 5 \end{Bmatrix}$ depending upon which file is currently being read. EXEMPT starts reading TAPE5, the normal system card input. When the first ALTFIL directive is encountered, subsequent ECL is read from TAPE4. When an END or ALTFIL is encountered on TAPE4, further input is read from TAPE5 until another ALTFIL or END directive is encountered. All input is transferred to TAPE3 before processing. Note, TAPE4 must be terminated by an ALTFIL directive.

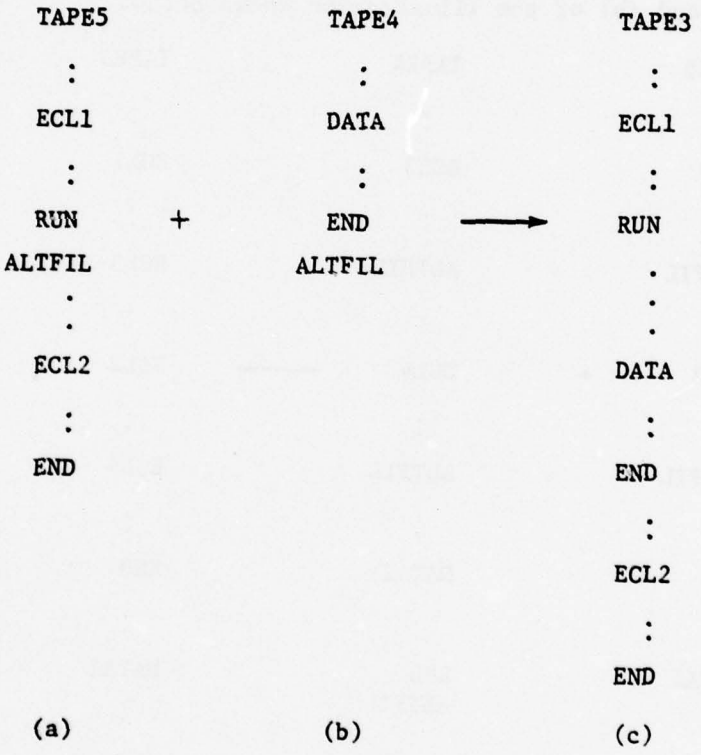
Example: Assume that TAPE4 and TAPE5 are structured as illustrated in (a) and (b) of the illustration shown below.



c. Run Directive

RUN

This directive causes the LIP to terminate processing and execute those commands encountered from the beginning of the ECL or since the last RUN directive. If data cards are to be read using the EXEMPT READC command, they must be placed following the RUN directive. Card input to be read by the SLAM or user provided subroutines appear after the 7/8/9 card following the ECL. When all ECL down to the RUN command has been executed, control returns to LIP to process subsequent commands. Note that a RUN directive followed by an ALTFIL directive will place the contents of the alternate file immediately following the RUN directive on TAPE3, the input file used for execution. This is illustrated in the following example.



END

d. End Directive

END

This directive is used to separate blocks of ECL and coded data. When encountered by the LIP, control is transferred to the CLP and will not be returned to the LIP. As illustrated below, DATA would be read during execution of ECL1. Control would return to the LIP to process ECL2.

```
ECL1
 .
 .
RUN
 .
 .
DATA
 .
 .
ECL2
 .
 .
END
```

2. COMMAND LANGUAGE PROCESSOR

When control is relinquished by the LIP, the user's ECL has been translated into a coded execution list. The data reside intact on TAPE3 for reading during execution of a READC command if present. There are four basic types of operations. These are control, logical, SLAM, and arithmetic operations.

These commands will be treated in separate sections which follow.

a. Command LOOP/LABEL

LOOP $\left\{\begin{matrix}\text{Name}\\ n\end{matrix}\right\}$,n

LABEL $\left\{\begin{matrix}\text{Name}\\ n\end{matrix}\right\}$

These are the control commands present in this release of EXEMPT. They must occur in pairs and cause all commands between them to be executed the number of times specified by n. The name or number used to identify the LABEL must be unique (i.e., multiple LOOPs cannot terminate on the same LABEL). However, LOOPs may be nested to any level as long as the maximum number of LOOP/LABEL commands does not exceed 100. This is the internal dimension of the loop control table and may be easily changed if required. Examples of both simple and nested loops are presented in the example below. Example (a) presents a simple loop using an integer label identifier while example (b) illustrates nested loops using alpha label identifiers.

```
.                    .
.                    .
ECL                  ECL
.                    .
.                    .
LOOP 5, 10           LOOP OUTER, 10
.                    .
.                    .
LABEL 5              LOOP NEST1, 5
.                    .
.                    .
END                  LOOP NEST2, 5
                     .
                     .
                     LABEL NEST2
                     LABEL NEST1
                     LABEL OUTER
```

(a) (b)

b. EXEMPT Logical Commands

There are two main functions fulfilled by the logical operations in EXEMPT. There are file manipulation and data set I/O. The underlying commonality of these operations is that they do not alter the internal data, they simply format them and read them into EXEMPT or write them to some device. These commands will be further classified as Random Access File Commands, Sequential File Commands, and Data Set I/O Commands.

(1) Random Access File Commands

EXEMPT uses FORTRAN logical unit 99 for random access I/O. All data manipulated internally is sorted on this logical unit. The detailed structure of this logical unit is presented in Section III. The user may have up to 20 files identified by a name or number on this logical unit. Each of these 20 files may contain up to 54 data arrays. A data array is an element of a data set. For instance, frequency and time domain data sets contain two data arrays, the dependent (response) data and the independent (frequency or time) data. A typeless data set contains only one data set (itself). In order to conserve space, whenever an independent data array is stored it is compared with existing data arrays of the same type, and if found to be identical to an existing array it will not be stored. This prevents clogging up the file with duplicate copies of the independent data which are usually the same for many data sets.

The following commands are to aid the user in creating, maintaining, and modifying his random access data base. The data to reside on the data base may have several sources. It may be read from cards, generated by EXEMPT itself, or interfaced from another computer program using the sequential I/O commands. Usually, some combination of these methods is used. When EXEMPT starts execution, it checks to see if a data base exists. If one does, it is loaded into EXEMPT. The user must have attached the data base as SCOPE local file TAPE99. If no data base exists, EXEMPT creates one for use during this particular run. If the user has requested that TAPE 99 reside on a permanent file device (REQUEST, TAPE99, * PF.), the created data base may be cataloged as a permanent file when the run concludes (CATALOG, TAPE99, MYFILENAME, etc.).

(a) File Identifier

FILE $\left\{\begin{matrix}\text{NAME}\\ \text{N}\end{matrix}\right\}$

This command designates the random access file to be used for data retrieval and storage. All data generated subsequent to this command will be stored on the file identified by NAME or N. If data already exist on this file, they will be made available to EXEMPT. As stated earlier, 20 such files may be designated by the user. If a file is never identified, the file name TEMP is assumed.

In the following example, assume that file EXPDAT contains experimental data to be used and it is desired to save the results obtained on a file identified as F15ANL.

```
FILE                EXPDAT
 .
 .
ECL TO RETRIEVE DATA
 .
 .
FILE F15ANL                    $SUBSEQUENT DATA STORED
 .
 .                             $ON FILE NAMED F15ANL
ECL TO PERFORM ANALYSIS
 .
 .
END
```

(b) Save Data Command

SAVE Name [,FILE = {Name / n}]

This command will result in the data associated with the name specified being written on the random access file specified. If no file is specified, the current file is used. If the file has not been previously created via a FILE command, one will be automatically created before the data are transferred.

This command is generally used to transfer data from the working file TEMP to another file to be saved on a permanent basis. All data sets which are created during an execution will be stored on the current file. Unless specified otherwise, this file has the name TEMP and may become full if data are not selectively moved to another file when not required in the subsequent commands. This is illustrated below where the file TEMP is used by default to generate data set ANT47A and the data are subsequently transferred to file ANTRSP with the SAVE command.

```
ECL TO GENERATE DATA SET AFTFRQ
  .
  .
ANT47A = INVFT (ANTFRQ)
SAVE ANT47A, FILE = ANTRSP
  .
  .
END
```

The data set name (and data) ANT47A will no longer reside on file TEMP. However, any future reference to ANT47A will cause the data to be loaded from file ANTRSP.

(c) Purge Random File Command

$$\text{PURGE} \quad \text{FILE} = \left\{ \begin{matrix} \text{Name} \\ \text{N} \end{matrix} \right\} \quad [,\text{SAVEFN}]$$

This command causes all data stored on the specified file to be purged. Unless SAVEFN is specified, the file name will also be purged from the master file directory. In practice, this command can be used to clean up a data base by purging data no longer required. It may also be used to eliminate the data of file TEMP before the conclusion of a run to prevent subsequent users of the data base from using temporary data saved during the current run.

(d) Delete Data Set Command

$$\text{DELETE NAME1} \left[\text{,NAME2, ...}\right] \left[\text{,FILE} = \left\{\begin{matrix}\text{NAME} \\ \text{N}\end{matrix}\right\}\right]$$

This command causes the data set(s) named to be deleted from the current file or one specified by the user. This command can be used to clean up a file on the data base by selective purging of data sets no longer required.

(e) Read File Data

(1) READF Name [,FILE = {Name / n}]

(2) READF FILE = {Name / n}

The two forms of this command perform two distinct functions and must not be confused. In order to aid in the discrimination of these functions, the commands will be discussed separately.

(1) READF Name [,FILE = {Name / n}]

This command causes the data set specified to be recovered from the random access file specified. If no file is specified, then the current file is searched for the data set. A very important action takes place when the data set is recovered. First, the data set is stored on the current random access file if it was retrieved from another random access file. Secondly, the global independent array corresponding to the data set type is overwritten with the independent data of the data set. Thus, if the data set is a frequency domain type, FREQ will be overwritten with the frequency values associated with the data set. If it was a time domain type, TIME would be overwritten. The primary purpose of this command is to allow the user to use the same FREQ and TIME data for data sets which reside on different files. The names FREQ and TIME may not be used directly. This is due to the fact that these are special names and there may be several arrays on each file with these names. EXEMPT keeps track of the different data by linking these independent arrays to a particular response data array to form a complete data set.

For example:

READF ANT47A FILE = ANTRSP

would retrieve the data associated with ANT47A and replace the global TIME array with that associated with data set ANT47A on file ANTRSP. If ANTRSP is the current file name, then READF ANT47A would simply overwrite TIME with the independent data of ANT47A.

The second form of this command is:

(2) READF FILE = { Name / n }

This command informs EXEMPT of the data set names present on the file specified. During LIP execution, all data set names are entered in a symbol table; however, the file location is unknown. When this command is executed, the names of the data sets on the specified file are recovered and if a match is found in the symbol table, EXEMPT is informed that the data associated with the symbol reside on the designated file. This allows the user to have data sets with the same name on different files. If a data set is referenced before it is defined either by appearance in a subroutine call or as the result of an operation, a fatal error will occur. This is analogous to the use of an undefined symbol in FORTRAN. If the user wishes to use data from a previously generated data base, a READF command for each of the files containing the data sets to be used is appropriate. This feature also allows the use of different data referenced by the same name at different points in the execution. This occurs since the ECL is executed sequentially and the READF command will only effect those occurrences of the data set name which follow it. This is illustrated in the example below.

```
READF          FILE = E4LIB
  .
  .
ECL            $DATA SETS ON E4LIB WILL BE USED
  .
  .
READF          FILE = B1LIB
  .
  .
ECL            $DATA SETS ON B1LIB WILL BE USED
  .
  .
END
```

NOTE: Unique data set names contained on E4LIB are still available to the ECL following the READF FILE=B1LIB command. However, those names that are the same will be available only from B1LIB.

(f) Save Random Access Files

SAVRAN [FILE = { Name / N }]

This command permits the random access data base to be saved on a sequential file which would normally be a magnetic tape. This is required since the data base may grow to a size which demands a lot of computer resources. Also, this provides a backup capability for the data base. If the file is specified, it must be present as a SCOPE local file with the name specified. If the numeric designator is used, TEMPn must be the SCOPE file name. If n is a single digit number, it must be preceded by a zero. If the file designator is omitted, the default name is NEWDAT and a SCOPE file with this local name must be available.

The sequential file format and content is presented as follows.

Record	Contents
1	Active file units table and number of active file units
2	File header and director, first active file
3	Frequency/time response or independent array, data set 1
4	Frequency/time response or independent array, data set 2
.	.
.	.
.	.
N+2	Frequency/time response or independent array, data N ($N \leq 54$)
N+3	File header and directory, second active file
N+4	Frequency/time response or independent array, data set 1
.	.
.	.
.	.
M+N+3	Frequency/time response or independent array, data set M ($M \leq 20$)
.	.
.	.
.	.

(g) Recover Random Access Data Base

$$\text{GETRAN} \quad \left[,\text{FILE} = \left\{ \begin{matrix} \text{Name} \\ n \end{matrix} \right\} \right]$$

This command allows recovery of the random access data base saved by the SAVRAN command. The same rules apply for the SCOPE local file names, except the default name is OLDDAT. GETRAN can only recover files when they are written exactly as specified in the SAVRAN command.

(h) List File Contents

$$\text{LIST INDEX} \quad \left[,\text{FILE} = \left\{ \begin{matrix} \text{Name} \\ \text{n} \end{matrix} \right\} \right]$$

This command allows the user to list the file names and data set directories for the random access data base. If the particular file is designated in the FILE field, the index is restricted to that file specified. Note that the contents of the data sets are not listed. Another LIST format is presented under OUTPUT commands for this feature. The output from this form identifies the file name, data set name, data set type, location of independent array, and the data set title.

(i) Data Set Title Specification

TITLE [,Name] [,FILE = {Name / n} ,] *Hollerith string*

This directive associates the Hollerith string of characters with the file or data set specified. The Hollerith string must contain 120 or less characters.

If the Name and FILE are omitted, the title is associated with the current file. If Name and FILE are both specified, then the title is associated with the data set on the specified file. When only one of the fields is missing, the title will be associated with the field provided.

(j) Rename File or Data Set

RENAME [Name,] [FILE = { Name / n } ,] Newname

This command changes the name of the item specified as discussed below.

RENAME Newname

This form changes the name of the current file to that specified by Newname.

RENAME FILE = { Name / n } Newname

This form renames the specified file to the Newname provided.

RENAME Name, Newname

This will rename the data set Name on the current file to be Newname.

RENAME Name, FILE = { Name / n } , Newname

This changes the data set Name on the designated file to Newname.

(2) Data Display Commands

Due to the large amounts of data which may be encountered in an execution, output is kept to a minimum except in the debug mode. Almost all output must be directed by the user. The data display commands permit the user to obtain a list of the data on the printer, punched cards, and various plot devices. These actions are in response to the LIST, PUNCH, and PLOT commands discussed in the following sections.

(a) Data Formatting

FORMAT Name [,Name,...], $\left\{\begin{matrix}\text{MP}\\\text{RI}\\\text{DB}\end{matrix}\right\}$ $\left[\left\{\begin{matrix}\text{DGREES}\\\text{RAD}\end{matrix}\right\}\right]$

This command directs the format of the output for the data sets specified. It applies only to complex data since real data are rigidly formatted. The command will specify that whenever a data set is displayed, it will be in real/imaginary (RI) or dB (DB) format. If not specified, data is displayed in magnitude/phase (MP) format. If RADS is specified, the phase will be displayed in radian measure. If not specified, the phase will be in degrees. In all cases, phase is bounded by $\pm 180°$.

(b) List Data Set Contents

(1) LIST Name 1, Name 2, ... [DB / RI / MP]

(2) LIST CURFIL

(3) LIST FILE = {Name / n}

Form 1 of this command is used to output the data associated with the data sets Name 1, Name 2 ... The data will be listed sequentially using the format specified. DB implies dB and the data will be normalized to the maximum value in the array. RI specifies listing the real and imaginary components separately while MP directs the output to be in magnitude and phase format. The default format is MP. These formats apply only to complex domain data. Noncomplex data, being real values only, are printed as they exist in the computer. All typed data lists will contain both the independent and dependent data.

Form 2 of the command will cause the data associated with all data sets on the current random access file to be listed. The output format used will be that specified on a FORMAT directive. If no format directive is specified, the data will be displayed in magnitude/phase for complex data and real for noncomplex data.

Form 3 lists the same data as form 2 for the file specified.

(c) Punch Data Output

PUNCH Name 1 [,Name 2,...] [,SINGLE] [,INDPNT]

This command will punch the data associated with the data set names specified. In all cases, the data from Name 1 will be punched before any data from Name 2. If SINGLE is specified, a single data entry will be punched per card. If SINGLE is not specified, there will be six fields per card containing data determined by the data set type and the presence of the INDPNT directive. If INDPNT is specified, the value of the independent variable will be punched with the value of the dependent data. If INDPNT is not specified, only the response data will be punched. All data are punched in 12 column fields with the first column of each field left blank. The data are punched on a 1XG11.5 format for each field and columns 76 through 80 will contain the Julian date.

If INDPNT is specified, the independent data will be punched preceding each value of the dependent data. For a frequency domain data set, this corresponds to three numbers for each element of the data set while two numbers suffice for time domain data. If SINGLE is also specified, this is the limit for each card. However, if SINGLE is not specified, two elements of a frequency domain data set, or three elements of a time domain data set will be punched on each card.

If INDPNT is not specified, only the dependent data will be punched. This corresponds to two numbers for frequency domain data and one number for time domain or typeless data. In this case, if SINGLE is not specified, there will be three elements per card for frequency domain data and six elements per card for time domain or typeless data.

Punched header cards will precede each punched data set. The header cards will contain the data set name, number of points, and the data set title.

(d) Plot Display Commands

```
{ PLOT   }
{ PRTPLT }   Name 1 [,Name 2, ..., Name, ...]   [, { LINLIN  }   ]
                                                   { LINLOG  }
                                                   { LOGLIN  }
                                                   { LOGLOG  }
                                                   { DBLIN   }
                                                   { DBLOG   }
[,TITLEX =  * Hollerith String *]  [,TITLEY =  * Hollerith String *]
[,OVRLAY]
```

These commands will plot the data sets specified on the device indicated. PLOT indicates a METALIB plot file is to be created; whereas PRTPLT indicates an EXEMPT printer plot is to be placed on the output file. The disposition of the METALIB plot file is controlled by the user prior to EXEMPT execution. The plot file will reside on LFN PLTFIL which may be disposed of immediately or cataloged as a permanent file for later disposition. (See Section V for an example of the METALIB file disposition).

To receive any plots through the METALIB system, the user should do the following steps prior to execution:

(1) Request PLTFIL to be a Q file.

(2) Dispose PLTFIL to the proper plot device.

For example, to receive a CALCOMP plot, the following JCL cards are inserted prior to EXEMPT execution:

6600	7600
REQUEST(PLTFIL,*Q)	DISPOSE(PLTFIL,*MF=PCF,ST=ANY)
DISPOSE(PLTFIL,*MF=PCF)	LGO.
LGO.	

The following plot devices are available to the user:

CODE	DEVICE	CHARACTERISTICS
PCF	CALCOMP DRUM	FULL SIZE, .01 INCREMENTS, PLAIN WHITE PAPER
PCG	CALCOMP DRUM	FULL SIZE, .01 INCREMENTS, RED RULES PAPER
PG4	GOULD 4800	
PVT	VERSATEC	
PB1	FR80	B/W 16 MM FILM
PB3	FR80	B/W 35 MM FILM
PF8	FR80	48 X MICROFICHE

Also, if the user wishes to have a printer plot of the META PLOT file, the following card is needed after EXEMPT execution:

DIRECT(PLTFIL,,PRINTER)

When PLTFIL is directed to the printer, the META system converts the file for printing to 8 lines per inch rather than the normal 6 lines per inch.

The axis scaling will be determined by the LINLIN through DBLOG parameters. The first acronym refers to the Y axis and the second refers to the X axis. When DB is specified, the data will be normalized to the maximum value of all data sets in the list. The value plotted will be $20 \log_{10} \frac{V}{V_{max}}$.

The axis titles are controlled by the TITLEX and TITLEY fields. If these are absent, the X axis title will be FREQUENCY or TIME, as appropriate. The Y axis title will be DB MAGNITUDE on LN MAGNITUDE, or MAGNITUDE as directed by the axis scaling. If the TITLEX or TITLEY fields are provided, the Hollerith string between the asterisks will appear as axis titles. Maximum title length for PRTPLT is 30 characters and for PLOT it is 20 characters.

Overplotting of data sets is controlled by the OVERLAY field. For PRTPLT, only one data set per plot will result; however, all plots will have the same scale for easy comparison. For PLOT, up to four data sets may be overplotted per plot frame. The reason for the limitation is that for CALCOMP plots there is only enough room on a frame for a plot plus four data sets' characteristics. These characteristics include the data set name and a 120 character title. More than four may be overplotted; however, only the first four data sets will have any of its characteristics printed on the plot frame.

(3) Sequential File Commands

In addition to the random access file capability, EXEMPT also will manipulate sequential (typically magnetic tape) files. This type of file is the primary interface with other user programs. The EXEMPT subroutine EXINTO may be incorporated in a user program to write an EXEMPT compatible sequential file. This routine is discussed further in Section III. In general, the initial input data for EXEMPT will be experimental or generated by another modeling program. The user may smooth or modify the data before they are written to tape for future incorporation into the EXEMPT data base. There are two levels of control allowable on the sequential file. Entire data sets (header, dependent, and independent data) are referenced as blocks. Groups of blocks are delineated by FORTRAN end of file marks and the file may be manipulated in either blocks or files. This permits the user to minimize his magnetic tape library since he can direct EXEMPT to position the tape at the correct data set (block) and subsequently read the data and associate them with a specific internal EXEMPT symbolic name.

The following pages present a description of the sequential I/O commands and illustrations of their use. Detailed illustrations of the sequential file structure are presented in Section III.

FRQFIL

TIMFIL

(a) File Designation Command

$$\text{FRQFIL} = \left\{ \begin{matrix} \text{Name} \\ \text{n} \end{matrix} \right\}$$

$$\text{TIMFIL} = \left\{ \begin{matrix} \text{Name} \\ \text{n} \end{matrix} \right\}$$

This command establishes the sequential files for frequency and time domain data. These data must be stored on separate sequential files. All subsequent sequential file identifiers may then be FRQFIL or TIMFIL and the proper user file will be referenced. Therefore, in the following command descriptions, FRQFIL or TIMFIL may be used as the file identifier. The file identified in this command must be attached as a SCOPE local file with the same name identifier as the TEMPn designator when n is used. When n is a single digit, it must be preceded by zero.

(b) Skip File Command

$$\text{SKIPF} \left\{ \begin{matrix} \text{Name} \\ \text{n} \end{matrix} \right\} \left[, \pm N \right]$$

This command will skip the number of FORTRAN files determined by $|N|$. If N is positive, the file will be positioned forward while if N is negative, the file will be positioned backwards to a preceding file. If N is omitted, it will default to +1. The file identifier $\left\{ \begin{matrix} \text{Name} \\ \text{n} \end{matrix} \right\}$ must be attached by the user as a SCOPE local file name. If the numeric file identifier n is used, the local file name is TEMPn. If Name is used, then the local file must have the same name.

For example, suppose that a user has two sequential files on disk for convenience. Then the SCOPE control cards and ECL would resemble

```
ATTACH, NAMFIL, SEQFIL1, ID = USRID.
ATTACH, TEMP23, SEQFIL2, ID = USRID.
EXEMPT.
7/8/9
ECL
SKIPF NAMFIL   5 $ SKIP FØRWARD 5 FILES
SKIPF  23   1    $ SKIP FØRWARD 1 FILE
ECL
SKIPF NAMFIL -2 $ SKIP BACK 2 FILES
```

(c) Skip Blocks Command

SKIPB $\left\{\begin{matrix}\text{Name}\\ \text{n}\end{matrix}\right\}$ $\left[, \pm N\right]$ $\left[, \left\{\begin{matrix}\text{FRQFIL}\\ \text{TIMFIL}\end{matrix}\right\}\right]$

This command will cause EXEMPT to reposition the file by N data sets. Positive N results in forward movement, while negative N will reposition the file to a previous data set. The file identifier may be a name or number n and must be a SCOPE local file name. If the file identifier is not FRQFIL or TIMFIL, then one of these must be specified to inform EXEMPT of the type of data contained on the file. Only one type of data is allowed when using the SKIPB command.

For example, the following SCOPE and ECL would position the file before the third data set of the second file on SCOPE local file FRQRSP.

```
ATTACH, FRQRSP, USRPFNAME, ID = USERID.
EXEMPT.
7/8/9
FRQFIL = FRQRSP
SKIPF, FRQFIL, 2
SKIPB, FRQFIL, 2
```

If the FRQFIL command had not been used, then the ECL would be

```
SKIPF, FRQRSP, 2
SKIPB, FRQRSP, 2, FRQFIL
```

(d) Read Sequential File

READ Name 1 [, Name 2, ... Name], FILE = { Name / N }

This command will cause the number of data sets named in the list to be read from the designated file and stored on the current random access file with the specified names. The data sets will be read sequentially; that is, the first will be identified as Name 1, the second as Name 2, etc. The type of data and format are specified in the header record. Subsequent reference to the data sets will use the data read from the sequential file.

Continuing the sample from the SKIPB command, we could read the next three data sets into EXEMPT with the ECL instruction

READ RSP1, RSP2, RSP3 FILE = FRQFIL

(e) Read Cards Command

READC Name [, N1], Name [, N2] ...

READC Name, Name, ... [, N]

This command will result in a coded card input read. The data will be decoded for entry into the specified array storage area. If the number N is specified with the array, the arrays are filled sequentially. If the number is specified at the end, the arrays are filled in parallel. The equivalent FORTRAN statement is shown below.

```
EXEMPT:     READC A, 5, C, 8
FORTRAN:    READ*, (A(I), I = 1, 5),
            (C(I), I = I, 8)
EXEMPT:     READC A, B, C, 10
FORTRAN:    READ*, (A(I), B(I), C(I),
            I = 1, 10)
```

If the named data sets have independents, then those independents are read as well as the imaginary part of a frequency domain data set.

The cards to be read are imbedded in the ECL and on the card input file on the ALTFIL file. This is illustrated below.

```
.
.
READC  A, B, C, 10
.
.
END
.
.
DATA FOR A, B, C in free field format
.
.
END
7/8/9
```

(f) Write Sequential File

WRITE Name 1 [, Name 2, ... Name], FILE = $\left\{\begin{matrix}\text{Name} \\ \text{N}\end{matrix}\right\}$

This command will write the data associated with the data sets named on the file specified in EXEMPT format. This format is discussed with the EXINTO subroutine in Section III.

Continuing the example from the READ command, assume that RSP1, RSP2, and RSP3 had been modified by the execution and have the same names. It is desired to save the new data in place of the old data, and they can be the last data on the file. The ECL would be

```
SKIPB FRQFIL -3
WRITE RSP1, RSP2, RSP3
```

(g) FORTRAN End of File

$$\text{EOF} \left\{ \begin{matrix} \text{Name} \\ \text{N} \end{matrix} \right\}$$

This command will write a FORTRAN end of file (EOF) on the file specified. This may be used to separate groups of data and also to aid in positioning the tape with SCOPE control cards. It also aids EXEMPT in positioning the file efficiently when more than one file is on the file.

(h) Rewind Sequential File

REWIND { Name / N }

This command will rewind the designated file and reposition it at the beginning of the first file. The FORTRAN rules apply in that if the last operation on the file was a WRITE. an EOF will be written before the file is rewound. If the last operation was a READ, the file is simply repositioned.

(i) Return Sequential File

$$\text{DROP} \left\{ \begin{matrix} \text{Name} \\ \text{N} \end{matrix} \right\}$$

This command returns the designated file to the system. Its primary purpose is to prevent EXEMPT from tying up a tape servo after the data have been read. The DROP command will free the unit for other users while EXEMPT executes.

(4) EXEMPT Library Functions

EXEMPT contains functions to perform certain numeric operations on previously defined data and to generate analytical data sets to support linear systems analysis. All of these functions may apppear in arithmetic expression. When used with an arithmetic expression, the total function text must be enclosed within parentheses. The two forms are illustrated below.

```
Name = FUNCTION (Name 1, Name 2), P1 = n, P2 = m
Name = A * (FUNCTION (Name 1, Name 2), P1 = n, P2 = m)
```

In the first example, the function will generate the data set name by operating data sets Name 1 and Name 2 using the parameters P1 and P2. In the second example, the data generated by the function will be multiplied by A before being stored as Name.

As a rule, the parameters in the list are optional and if no parameters are associated with the command format, the command need not have the enclosing parenthesis when used in an arithmetic expression. If parameters are in the command format but not used, the enclosing parenthesis are required.

(a) Data Interpolation

Name = INTERP (Name 1 [, Name 2]) [, NPTS = n] [, PT1 = v,] [PT2 = v]

This command will interpolate the dependent data associated with Name 1 onto a new independent data base associated with Name 2 and store the result as Name. If Name 2 is not provided, then the global frequency or time array will be used depending on the type of data in Name 1. The interpolation is an Aitken-Lagrange polynomial of degree specified NPTS = n. NPTS may have any value from 1 to 10. The default value is NPTS = 1, or linear interpolation. The parameters PT1 and PT2 specify the first and last values of the new independent data. If PT1 is not provided, interpolation will start with the first point of the new independent data. If PT2 is not provided, interpolation will continue to the last point of the new independent data.

If the new independent data of Name 2 extend beyond the domain of the data in Name 1, a warning will be given to the user. The new value will be the last value in the domain of Name 1. That is, the extended data set will contain constant values in the extended region.

In the following example, assume VOLTS is a frequency domain response function for unequally spaced values of frequency from 1 to 100 MHz. If a Fast Fourier Transform (FFT) is to be used, then the independent data must be equally spaced and start at a value of zero.

```
.
.
ECL
.
.
FREQ = 0., LIN512, 100E6
VOLTS = INTERP (VOLTS)
.
.
END.
```

The resulting values associated with VOLTS for frequencies between 0 and 1 MHz will be the original value at 1 MHz.

(b) Data Editing

Name = EDIT (Name 1) [, PT1 = v] [, PT2 = v]

This command removes bad and redundant data from the data set Name 1 and stores them as data set Name 2. The parameters PT1 and PT2 have the same meaning as the INTERP function.

ZERO is a user definable parameter intended to represent a reasonable criteria for evaluating equality between floating point numbers. ZERO default value is 1.E-10.

Redundant points occur when two consecutive values of the independent data I_n and I_{n+1} are such that

$$\left| \frac{I_n - I_{n+1}}{I_n} \right| < \text{ZERO}$$

Bad data points occur when two consecutive values of the dependent data D_n and D_{n+1} are such that

$$\frac{D_n - D_{n+1}}{D_n} > \frac{1}{\text{ZERO}}$$

(c) Merging Data Sets

Name = MERGE (Name 1, Name 2) [{ MAXVAL }]
                               [{ MINVAL }]

This function allows the user to combine data from two data sets into one data set. The source data associated with Name 1 and Name 2 will be combined in ascending order of their independent variable. When two values of the independent variable are found to be the same (within an error of ZERO), the action directed by MAXVAL, MINVAL, or the absence of these parameters will be taken. If MAXVAL is specified, the maximum dependent value will be chosen. If MINVAL is specified, the minimum will be chosen. If neither parameter is present, the resultant value will be the average of the values.

From the INTERP example, VOLTS could be extended to zero frequency using a MERGE function. Suppose NULL is a data set containing a zero response for frequencies of 0 and 1 MHz. Then the following ECL would generate VOLTS as an equally spaced data set between 0 and 100 MHz.

```
.
.
FREQ = 1.,LIN101, 100.E6
VOLTS = INTERP (VOLTS)
VOLTS = MERGE (VOLTS, NULL)
.
.
END
```

(d) Fourier Transform

Name = FT (Name 1 [, Name 2])

Name = INVFT (Name 1 [, Name 2])

The commands will transform the data sets Name 1 to the time (INVFT) or frequency (FT) domain. If Name 2 is provided, the output data set, Name, will have the same independent data as Name 2. If Name 2 is not provided, the output independent data will be generated using the Nyquist criteria on the independent data of Name 1. For example, if I_{max} and I_{min} are the extremes of the independent data associated with Name 1 which contains N data points, then the independent data of Name will have increments δ computed from

$$\delta = \frac{1}{2\left((I_{max} - I_{min})/N\right)}$$

If Name 2 is not provided and the independent data for Name 1 are:

1 - Equally spaced

2 - $I_{min} = 0$,

then the output data Name will be generated using the FFT. This is the only method to employ the FFT. All other input will cause the use of a full Fourier integration. The INTERP function is provided to allow the user to transform data from unequally spaced independent arrays to equally spaced independent arrays.

In this release of EXEMPT, there is no checking performed to assess the validity of the transform.

In the following ECL examples, EQUAL is assumed to be an equally spaced frequency domain data set with a zero frequency value. UNEQ is an unequally spaced time domain data set.

```
EQTIM = INVFT (EQUAL)
NTPUNQ = INTERP (UNEQ, TIME)
EQFRQ = FT (NTPUNQ)
FRQUNQ = FT (UNEQ)
```

EQTIM will be an equally spaced time domain data set generated by the FFT. NTPUNQ will be the data from UNEQ interpolated onto the global TIME array which we will assume has a zero value and is equally spaced. Consequently, the FFT will generate EQFRQ as an equally spaced frequency domain data set with a zero frequency value. FRQUNQ will be generated using the full Fourier transform of the data in UNEQ.

(e) Double Exponential Transform

Name = TRDEXP (V1, V2, V3)

This function generates the Fourier transform of:

$$V1 \left(e^{-V2t} - e^{-V3t} \right)$$

for the frequency values in the global frequency array FREQ.

The output data set Name is a frequency domain data set which may be used as any other frequency domain data sets. The primary purpose of this function is to allow the user to have an analytic function commonly encountered in EMP or EMC analysis. With the proper values of V2 and V3, this function can represent either an EMP or lightning pulse.

(f) Damped Sine Transform

Name = TRDSIN (V1, V2, V3)

This function generates the Fourier transform of:

$$V1\ e^{-V2t}\ \sin\ (2\pi V3t)$$

for the frequency values in the global frequency array FREQ.

The output data set Name is a frequency domain data set. A damped sine wave is a commonly encountered excitation in EMP and EMC analysis. The output data may be used as an analytic excitation for system analysis.

(g) General Polynomial Functions

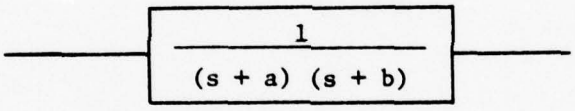
Name = POLY (V1, V2, V3, V4)

This function is used to generate a complex frequency dependent function which can represent a factor of a polynomial or two terms of a polynomial. POLY generates a real and imaginary part with user designated factor and power of ω (frequency in radians per second). This method will calculate factors which can be multiplied to form a transfer function. POLY can also be used for unfactored polynomials by pairing real and imaginary terms of ω. Each pair can produce a term which can be added to form a polynomial. ω is the radian frequency derived from the global frequency array as 2π*FREQ.

$$\text{POLY(V1, V2, V3, V4)} = V1\omega^{V2} + j\ V3\omega^{V4}$$

Example:

It is standard notation to represent a transfer function by:



The transfer function can be written and stored in TR by:

```
AB = A*B
 C = A + B
TR = 1./(AB-POLY(1.,2.,0.,0.)+POLY(0.,0.,1.,0.)*C)
```

(h) Phase Shift Function

Name = SHIFT (Namel, t)

This function is used to shift the phase of the frequency domain data set Name 1 by performing the following operation

$$\text{Name} = \text{Namel}, * e^{j\omega t}$$

for those values of ω associated with the frequency data of Namel. If $f(t)$ and $F(\omega)$ are transform pairs, then $f(t + t_o)$ and $F(\omega)\ e^{j\omega t_o}$ are also transform pairs. This, the SHIFT function allows the user to adjust phase without recomputing the transform.

(i) Data Set Typing

$$\text{TYPE Name} = \left\{ \begin{matrix} \text{TIME} \\ \text{FREQ} \end{matrix} \right\}$$

This declarative statement will explicitly define a data set identified by Name to be a frequency or time domain function. If a data set is not referenced in a type declaration, it will be typed implicitly. Implicit typing occurs when a data set name is referenced as the resultant of a function or arithmetic operation. In an arithmetic operations, the first independent data set encountered will be associated with the resultant. If none are encountered and the resultant is not typed, no type will be associated with it; however, if none are encountered and the resultant is typed, the appropriate global independent data will be associated with it.

Attempting to perform an operation or evaluate a function with improperly typed data will result in an EXEMPT fatal error.

LIN
LOG
REP
NXT

(j) Data Initialization

$$\left\{\begin{array}{l}\text{Keyword}\\ \text{NAME}\end{array}\right\} = \text{Value 1} \left\{\begin{array}{l}\text{LINn}\\ \text{LOGn}\\ \text{REPn}\\ \text{NXTn}\end{array}\right\} \text{, Value 2, Value 3 ...}$$

This command can be used to load data into an independent or typeless data set. The most common use will be to initialize the global TIME or FREQ data. These data sets may be redefined anytime during execution. The data starts at value 1 and increments according to the specification. The meaning of the specifications are:

LINn	Load data from the first value to the second value with n linear steps
LOGn	Same as LINn but with logarithmic steps
REPn	Repeat the previous value n times
NXTn	Repeat the next value n times

When initializing the arrays TIME and FREQ, the first and last values need not be specified for the LINn and LOGn option. As can be seen in Table 3, minimum and maximum values for TIME (TIMMIN and TIMMAX) and FREQ (FRQMIN and FRQMAX) are already defined. Additionally, these values as well as all other default valued keywords may be redefined by a simple replacement expression (i.e., A = value).

Examples:

FREQ = LIN50

This command loads the array FREQ with 50 equal linear steps. The minimum value (FMIN) is 0. and the maximum value (FMAX) is 30.E+6.

The command:

FREQ = .01, LIN10, .1, LIN10, 1., LIN10, 10., LIN10, 100.

will load the following data into the global frequency array..

.01, .02, .03, .04, .05, .06, .07, .08, .09, .1,
.2, .3, .4, .5, .6, .7, .8, .9, 1.,
2., 3., 4., 5., 6., 7., 8., 9., 10.,
20., 30., 40., 50., 60., 70., 80., 90., 100.

Table 3

KEYWORD DEFAULT LIST

KEYWORD NAME	INITIAL VALUE	FUNCTION
FREQMAX	30.E+6	MAXIMUM FREQUENCY DATA VALUE
FREQMIN	0.	MINIMUM FREQUENCY DATA VALUE
TIMMAX	1.E-5	MAXIMUM TIME DATA VALUE
TIMMIN	0.	MINIMUM TIME DATA VALUE
ZERO	1.E-10	CLOSENESS TEST FOR TIME OR FREQUENCY DATA
FRQFIL	-	CONTAINS THE FREQUENCY ONLY SEQUENTIAL FILE UNIT
TIMFIL	-	CONTAINS THE TIME ONLY SEQUENTIAL FILE UNIT
PAGLIN	51	DEFAULT VALUE FOR THE PAGE LENGTH IN LINES PER PAGE
PAGWID	81	DEFAULT VALUE FOR THE PAGE WIDTH IN COLUMNS PER PAGE
R1-R20	0	GLOBAL REAL ARRAY FOR USER SUBROUTINES

(k) Mathematical Expressions

Quantities used in constructing the expression include numerical constants, symbolic constants, function references, data sets, and global variables. Also available are a variety of mathematical symbols and several mathematical functions.

The expression is written using constructions similar to those used in FORTRAN. EXEMPT has the capability to evaluate mathematical expressions containing data set names, subroutine names, constants, and arithmetic functions.

1) Arithmetic Operations

The arithmetic symbols +, -, *, /, and **, are used to denote addition, subtraction or negativity, multiplication, division, and exponentiation, respectively.

All arithmetic operators must explicitly appear in mathematical expressions. No operations are assumed by juxtaposition of quantities.

To order the sequence of operations, parentheses may be used. For example, in division, only the quantity immediately following the division sign is considered to be in the denominator. Compound denominators must be enclosed in parentheses. These examples illustrate the point.

X6 = R1/.5*R3 is interpreted as $X6 = \frac{(R1)(R3)}{.5}$

X7 = R1/(.5*R3) is interpreted as $X7 = \frac{R1}{.5R3}$

The minus sign may be used to indicate the negative of a quantity:

X9 = -N(1)

Exponentiation is indicated by the symbol **, where the quantity appearing immediately after the symbol is the exponent. Exponents may be constant or variable. Compound exponents must be enclosed in parentheses. Exponents consisting only of integer constants should be written without the decimal point.

Data sets contained in expressions must have the same associated independent data. However, a data set that is neither frequency nor time dependent may be mixed within an expression.

There are two categories of mathematical operations. If a lower case letter represents a simple variable or constant (real or complex), and upper case letter represents an array (real or complex), and a ° represents the operators +, -, *, and /, then the categories may be described as follows.

a. First type. The arguments may be simple variables, arrays, or both.

$c = a \circ b$

$C = a \circ B$

$= (a \circ b_1, a \circ b_2, \ldots, a \circ b_n)$

$C = A \circ B$

$= (a_1 \circ b_1, a_2 \circ b_2, \ldots, a_n \circ b_n)$

b. Second type (exponentiation). The arguments may be simple variables or a simple variable and an array.

$c = a^{**}b$

$C = A^{**}b$

$= (a_1^{**}b, a_2^{**}b, \ldots, a_n^{**}b)$

2) Parentheses

Left and right parentheses are available for grouping purposes. Many levels of parentheses may be employed to a maximum of ten. It is important to check for proper pairing of left and right parentheses so that ambiguous or incorrect groupings are avoided. An example of correct usage is:

```
X1 = R1/(R2*(N+R3))
```

which represents the expression

$$X1 = \frac{R1}{R2(N+R3)}$$

3) Mathematical Functions

The following mathematical functions are available for use in mathematical expressions:

ABS(x)	Absolute value of x
ATAN(x)	Inverse tangent of x, result in radians
ATAN2(x,y)	Inverse tangent of y/x, result in radians
CMPLX(x,y)	Complex function
CONJG(x)	Complex conjugate of x
COS(x)	Cosine of x, x in radians
EXP(x)	Exponential function of x
ALOG(x)	Natural logarithm of x
MAX(x,y)	Maximum value of x and y
MIN(x,y)	Minimum value of x and y
MOD(x,m)	x modulo m
SIGN(x,y)	Magnitude of x with sign of y
SIN(x)	Sine of x, x in radians
SQRT(x)	Square root of x, x positive
TAN(x)	Tangent of x, x in radians

Function arguments must be enclosed in parentheses. All EXEMPT functions may also be included in arithmetic statements. EXEMPT functions with parameter list must be totally enclosed in parenthesis. Also, if a data set is complex, EXEMPT will perform the FORTRAN complex function.

There are three categories of functional operations. If a lower case letter represents a simple variable or constant (real or complex) and an upper case letter represents an array (real or complex), then the categories may be described as follows:

a. First type. If the argument is a simple variable, the result is a simple variable. If the argument is an array, the function is applied to each element of the array separately, and the result is an array.

b = f(a)

B = f(A)

= $\{f(a_1), f(a_2), \ldots, f(a_n)\}$

b. Second type. Functions with two arguments. If <u>both</u> arguments are simple variables, the result is a simple variable. If <u>either</u> argument is an array, the result is an array.

c = f(a,b)

C = f(A,b)

= $\{f(a_1,b), f(a_2,b), \ldots, f(a_n,b)\}$

or

C = f(A,B)

= $\{f(a_1,b_1), f(a_2,b_2), \ldots, f(a_n,b_n)\}$

c. Third type. The argument <u>must</u> be an array and the result is an array or both arguments <u>must</u> be arrays and the result is an array.

B = f(A)

C = f(A,B)

(5) User Subroutines

EXEMPT is provided with the capability to interface with user provided software using two methods. The first method involves replacing existing EXEMPT subroutines with the user subroutine of the same name. There are 45 of these available and they are identified as subroutine SUBA → SUBZ, OPR0 → OPR9, and MYCOM0 → MYCOM9. The interface for these is presented in Section IV. They may be treated as any other EXEMPT function with respect to usage. The user must replace the EXEMPT subroutine with his own of the same name using the SCOPE COPYL command. The identifiers SUBa and OPRn may be referenced directly; however, the MYCOMn is a LIP equivalency statement. The user defines MYCOMn as a symbolic name, subsequent reference to the name will result in subroutine MYCOMn being called by EXEMPT. This is illustrated below. When the name GTD is referenced after defining MYCOM5 as GTD, MYCOM5 will be called to perform the user's operations on the input data.

```
MYCOM5 = GTD
.
.
ECL
.
.
ANTRSP = GTD (FREQ)
```

The user must have provided a subroutine MYCOM5 interfaced as described in Section IV.

The second method for implementing the user's software is to append the entry point for a NAME subroutine. This is done by replacing the EXEMPT subroutine CNGSUB on the relocatable binary file by using the COPYL SCOPE utility with the append option specified (see example, Section V). The replacement must include an EXTERNAL declaration followed by a call to SETUP. The arguments to SETUP, shown below, include the user subroutine name in Hollerith form followed by the subroutine name. As many pairs of arguments are required as there are user-named subroutines, up to a maximum of 36 pairs.

```
SUBROUTINE CNGSUB
EXTERNAL RUN1, RUN2
CALL SETUP(4HRUN1,RUN1,4HRUN2,RUN2)
RETURN
END
```

. Further information and detailed description of the subroutine uses are given in Section IV. It is recommended that the user read Section IV before trying a user subroutine.

The general types of calls are illustrated below where A, B, and C represent data set names and Name is used as a user subroutine name.

$$C = \left\{\begin{matrix}\text{Name}\\ \text{SUBa}\\ \text{OPRn}\\ \text{MYCOMn}\end{matrix}\right\} \left(\left[\left\{\begin{matrix}\text{TIME}\\ \text{FREQ}\\ \text{A}\end{matrix}\right\} \quad \left[\text{, B}\right]\right]\right)$$

or

$$\text{CALL} \quad \left\{\begin{matrix}\text{Name}\\ \text{SUBa}\\ \text{OPRn}\\ \text{MYCOMn}\end{matrix}\right\} \quad \left(\left[\text{C ,} \quad \left\{\begin{matrix}\text{TIME}\\ \text{FREQ}\\ \text{A}\end{matrix}\right\} \quad \left[\text{, B}\right]\right]\right)$$

SECTION III
EXTERNAL FILE STRUCTURES

There are two data file structures that the user encounters. These are binary sequential files and random access files. These file structures are used automatically through the command execution. Also, it is possible for a user to format a binary sequential file from an external program for use on EXEMPT.

A master index will contain the information needed for accessing each data set on the random access file. The master index points to a subindex directory for the particular data set.

1. EXEMPT SEQUENTIAL BINARY FILE FORMAT

One of the ways in which the EXEMPT executive computer code can exchange data with other independent program is through standard format sequential binary files. Since the files are binary representations of floating point numbers, information interchange is limited to machines with the same word length and floating point number structure. Currently, EXEMPT standard formats are defined for recording frequency domain or time domain signals along with some title information.

The external unit names that are used for these files are determined by the user. The unit names can be integer numbers between 1 and 99 or standard FORTRAN file names in a 6L format. If the files are referenced by number, the external name for these files will be TEMPXX. The files for numbers between 1 and 9 will be written as TEMPØ1 through TEMPØ9. Therefore, any set of numbers can be used with these files and they will not be confused with standard FORTRAN unit numbers.

The structure of a sequential file is in terms of data set. Motion of the sequential file is done by sets rather than records. A READ will read one data set and a WRITE will write one data set.

The first record of a set will contain the number of points, data description, date, program name, and the title information. A time domain data type will have two other records and a frequency domain data type will contain three other records. The structures of these data types are:

a. Frequency Domain Data (Four Records)

1. Record 1: 17 words long
   (a) One word integer (N) indicating number of words in each of the following records.
   (b) One word containing data description in CDC display code.
   (c) One word containing data description of data values are 8HMAGPHASE, 8HREALIMAG, 7HDBPHASE.
   (d) One word containing a date in display code.
   (e) One word containing display code or name of program which wrote data file (e.g., 7HSCEPTRE, 6HYSYNAP, etc.).
   (f) 12 words of title information in CDC display code.
2. Record 2: N words long containing the array of frequency points.
3. Record 3: N words long containing the real part or amplitude of a frequency function.
4. Record 4: N words long containing the imaginary part or phase of a frequency function.

b. Time Domain Data (Three Records)

1. Record 1: 17 words long; same as for frequency data.
2. Record 2: N words long containing the array of time points.
3. Record 3: N words long containing the values of the time function.

When writing an external file, the user should declare his file to be record type, RT=W through the FILE and LDSET SCOPE utilities or use BUFFER IN/OUT.

To properly write an EXEMPT formatted sequential file as described above, a subroutine EXINTO is available. EXINTO is a FORTRAN subroutine which is callable by other FORTRAN routines. Its purpose is to write EXEMPT format data files.

The proper call is:

```
CALL EXINTO (ITYPE,LFN,YR,YI,FT,N,IOP,TITLE,IERR,NAME)
```

Arguments are as follows:

ITYPE: A two-word integer array identifying the type of data.
ITYPE(1) = 0 for time data or 1 for frequency data
ITYPE(2) = 8HREALIMAG or 8HMAGPHASE or 7HDBPHASE
as appropriate to describe frequency data.

LFN: Logical file name of the file on which data are to be written. It is either an integer from 1 to 99 or a name with a maximum of six characters in L format. The system file name for an integer specified file will be TEMPnn where nn is the integer. The file name must be declared on the FORTRAN program card. To reduce memory requirement, set buffer size to 0.

YR: Data array containing real part (or magnitude) of a frequency dependent function or amplitude of time dependent function.

YI: Data array containing imaginary part (or phase) of a frequency dependent function (not used by time).

FT: Data array containing frequency or time values.

N: Number of points in each array.

IOP: Indicates type of operation.
1-Rewind
2-Skip data block
3-Backspace data block
4-Skip file
5-Read data block
6-Write data block
7-Write EOF
8-Drop File

TITLE: Up to 12 words of Hollerith information.

IERR: Return error codes:
1-ITYPE not 0 or 1
2-Illegal operation code
3-Illegal file name
4-EOF encountered
5-End of information (EOI) encountered

6-Read parity error

7-End of record (EOR) encountered, data array not N words long

8-Write parity error

NAME: Name of program calling EXINTO in left justified display code (e.g., 7HSCEPTRE or 6HYSNAP)

Example:

CALL EXINTO (0,3,AMP,0,TIME,679,6,TITLE,IERR,NAME)

These arguments have EXINTO write 679 points of TIME array data with a title of LFN TEMPØ3.

EXEMPT will read two types of tape format. The first type was previously discussed. The second type, EXEMPT old format, may be read only. The first record will contain the number of points and title information.

An old format EXEMPT data block is defined as follows:

a. Frequency Domain Data (Four Records):

1. Record 1 - 13 words long; (1) one word integer (N) indicating number of words in each of the following records. (2) 12 words of title information in CDC display code.
2. Record 2 - N words long containing the real part or magnitude of a frequency function.
3. Record 3 - N words long containing the imaginary part or phase of a frequency function.
4. Record 4 - N words long containing the array of frequencies to match the function of the previous two records.

b. Time Domain Data (Three Records):

1. Record 1 - 13 words long; same as frequency data.
2. Record 2 - N words long containing the values of a time function.
3. Record 3 - N words long containing the time array corresponding to the values in previous record.

EXEMPT will use the following method to determine whether a data set is time or frequency domain data:

a. A data set not previously defined will be assumed to be frequency domain data. A data set that has been previously defined will retain its domain type. (To classify a data set see the TYPE command.)

b. To avoid reading wrong data, an EXEMPT error will occur when the number of points in the first record is the same as the length of the first record. Also, when the number of points read do not agree with the number of points specified, an EXEMPT error will occur.

Example:

Figure 3 is a diagram of sequential file containing frequency data sets. There are four files ordered W, Y, X, and Z. Each data set on these files are ordered WA, WC, WB, WD, YA, YB, YC, YD, etc. It is desired to read the files and data sets alphabetically (i.e., read file W in the order WA, WB, WC, WD followed by file Y).

To read the data sets in the order desired, one must be able to skip files and blocks. In EXEMPT, to skip n files means to skip over n EOF marks and position the head at the beginning of the current file. To skip n blocks (note: one block is one complete data set) means to skip over n data blocks. Also, skipping zero files or blocks (i.e., n = 0) is quite legitimate; however, the meanings are different. For blocks, this is a null operation; but for files, the head is set at the beginning of the current file.

By following the EXEMPT command language flow of Figure 4, one can see how the data sets are read in the desired order.

2. RANDOM ACCESS FILES

The random access mass storage file is structured as a master index/subindex file. The master index will include one subindex containing all active file names. There can be at most 20 user named files on the master index. For each file name there is a directory containing each subindex and its header whose location is the first subindex of the file. A file has 54 other subindexes that are used as data array locations. The header for each data array is 50 words long and contains the data array

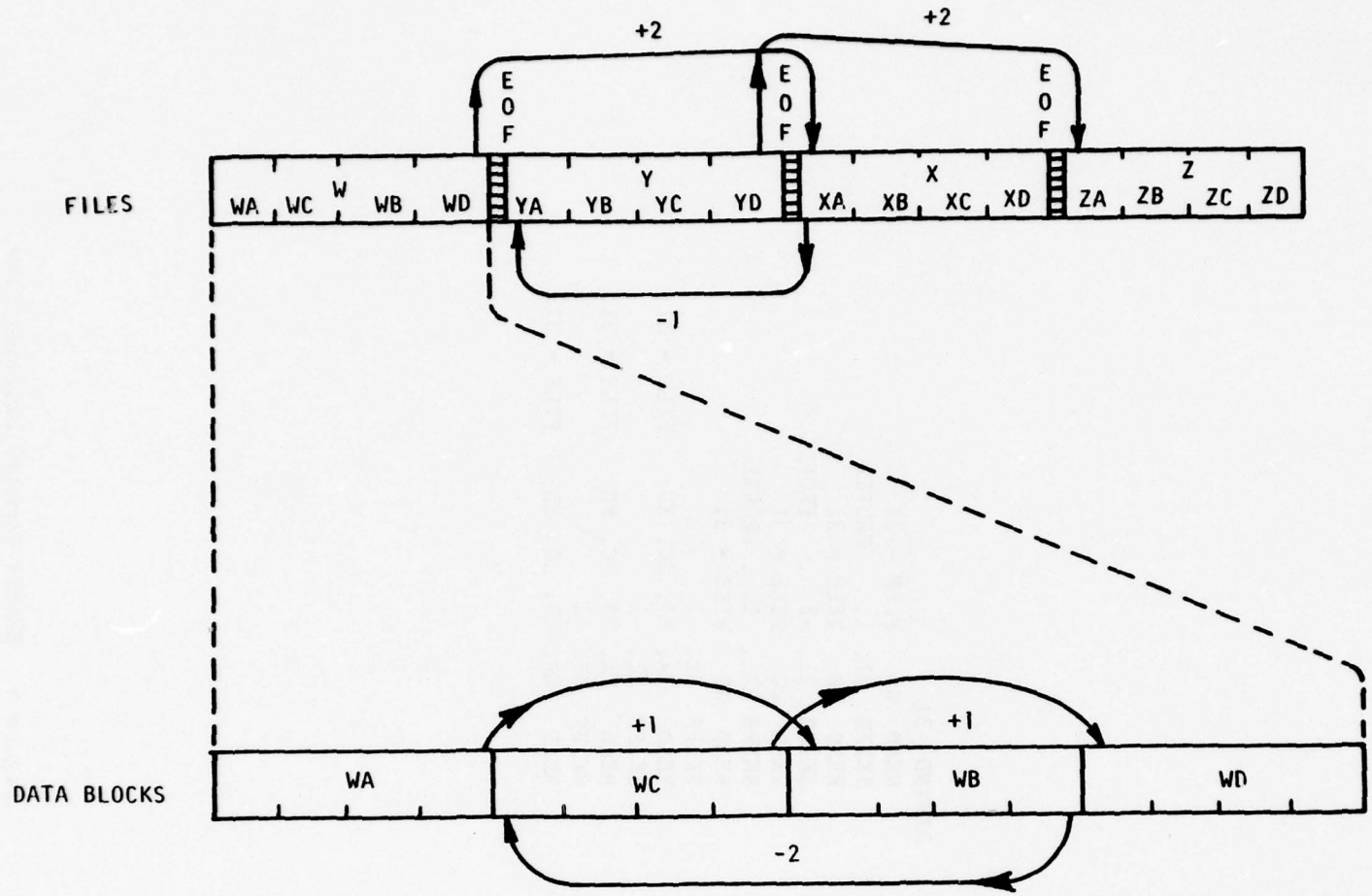


Figure 3. Skipping Files and Blocks

```
REWIND  31
    READ  WA   FILE = 31
    SKIPB  31, 1,    FRQFIL
    READ  WB   FILE = 31
    SKIPB  31, -2  ,   FRQFIL
    READ  WC   FILE = 31
    SKIPB  31,  1  , FRQFIL
    READ  WD   FILE = 31
    SKIPF  31,  2
    READ   XA, X3, XC, XD    FILE = 31
    SKIPF  31, -1
    READ   YA, Y3, YC, YD    FILE = 31
    SKIPF  31.  2
    READ   ZA, ZB, ZC, ZD    FILE = 31
```

Figure 4. EXEMPT Command Language Flow

name, 12 words of title information, the number of points, subindex location for the data array response type (i.e., time, frequency, or no type), subindex location for the independent data, the format type, and 31 words of blank information. Figure 5 shows the file structure of the random access data base. NOTE: In discussions in this document, the individual subfiles are termed files and are referred to as separate files. In fact, they are subfiles of the one random access mass storage FORTRAN file unit 99.

EXEMPT keeps a list of names associated with the 20 files. This set of names plus the number of active files is written as the 21st record. Thus, the 21st record contains only 21 words which are the active file names and the number of active files. When a new file is added, the first zero entry in the active file namelist is used for the index for the subfile so that a file will be created corresponding to the index in the list of active file names.

The random access file is opened with the master index in effect. The master index is kept in the common block/MSTRG/ and a subindex corresponding to the proper active file name is read from the random access file. Only one subindex is contained within EXEMPT at any time. Thus, when referencing other files in the subindex the file currently in use is rewritten onto the random access file before a new subindex is read.

Particular care is made in using the random access disk in this master index/subindex form to always write the current subindex onto the record indicated from the master index before a new subindex is read. In this way, the current values for the record pointers are maintained correctly so that data are not lost on the random access file because of an index error. Each file is organized in the same way, containing the file header and directory and then the block records for the 54 data arrays on each file.

The first subindex is always used for the temporary file. When opening the mass storage file, the temporary file is always created first and then the file that the user desires. If the user is accessing the temporary file, the program does not open another file. However, if the user has referenced a particular file by name or number, the user's file will be opened as the second file using subindex 2. This process

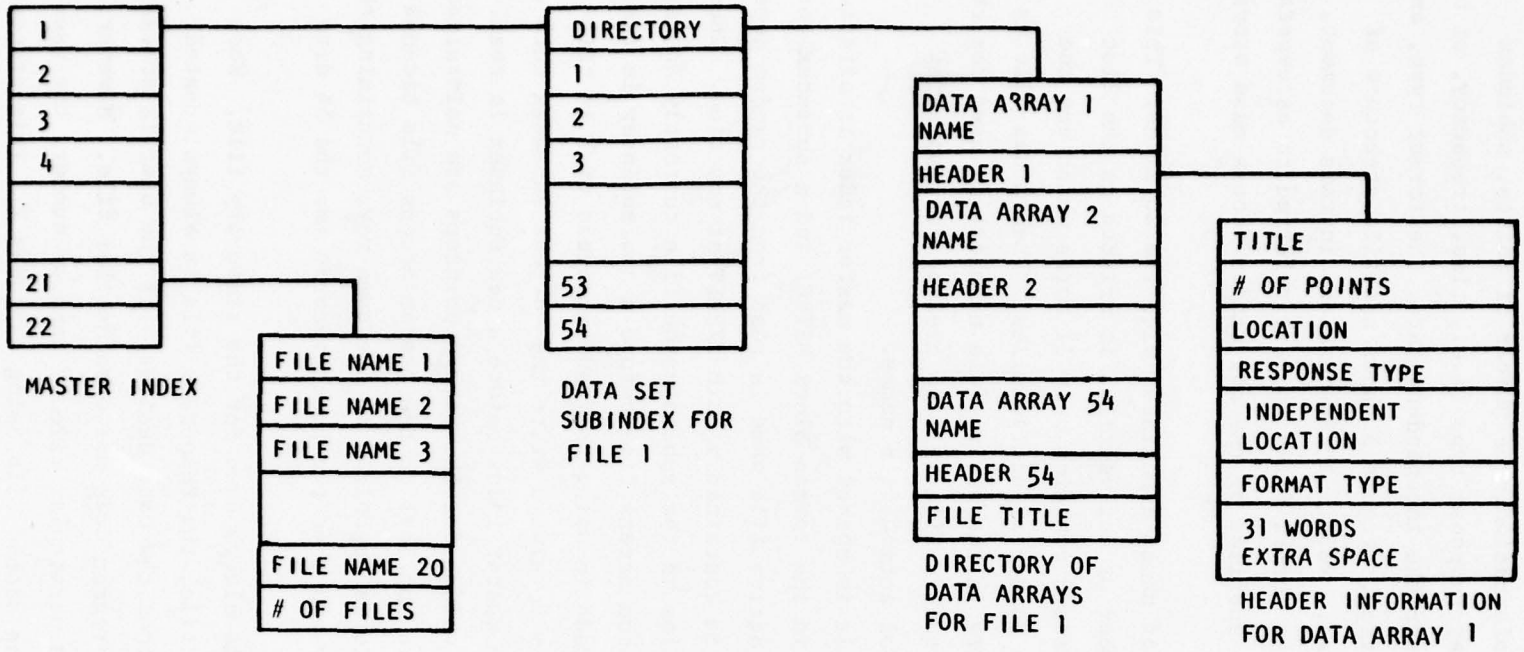


Figure 5. Random Access File Structure

continues as the user opens new files until all 20 files are in use. When the user requests that a file be purged, two procedures are followed. If the entire file is to be unloaded, the subindex pointing to the file is zeroed, the name in the active file namelist is zeroed, the number of files active is decremented, and the resulting subindex and new list of file names are written onto the random access disk. The second procedure occurs when the file is to be set empty. In this case, the file block directory is changed to indicate that all the blocks are empty. Only the file header and directory information are rewritten. The subindex, which is in use after these operations, is then written onto the disk and operation continues. The next time that particular file is accessed, the program recognizes that the file is empty and the first block will be used. Purge file operations are an irreversible process. If the user indicates a purge file operation, the contents of that file are removed completely (see PURGE command, Section II-B).

Lastly, when the particular EXEMPT job has been completed and a STOP is executed, the FORTRAN I/O operations automatically write the master index to the end of the file. This means that if no FORTRAN file errors occurred during operation, the random access file can be cataloged as a permanent file on the Air Force Weapons Lab computer system. When EXEMPT completes operation, the master index is always restored and the last used subindex is written onto the file. If an error occurs that is not associated with a FORTRAN file operation, the SCOPE operating system automatically calls a routine contained in EXEMPT called CLOSRM. No attempt is made to change the index to the master index and close the random access file correctly. If a fatal error is detected by the FORTRAN I/O, the routines associated with the FORTRAN I/O will automatically try to close all FORTRAN files. This means that the index in use at the time of the fatal error will be written as the last record on the random access file. If this happens to be a subindex, then the master index and the file structure is completely lost due to the error. This is a systems function with no user control to override the FORTRAN error processing. Consequently, it is not recommended that the random access file be used as a permanent file. It can be temporarily used, but a backup must be maintained. This can be done through the permanent data master tape.

The permanent data master tape is used to copy the random access disk to a sequential file which can be saved either as a permanent file or as a magnetic tape backup. This particular file is written in such a way as to minimize the storage requirements for the file and yet at the same time allow restoration of the random access file to the exact condition at the time when the master tape was written. The format for the tape structure is found in Table 4. The first record on the master data tape contains the active file names and the number of active files. This is the same record written on the random access file as record 21 of the master index. This allows the same active files to be regenerated in the same indexing originally obtained. Each active file is then written on the master data tape. Each file is formatted on the master tape by writing the file header and directory, and all data arrays referenced in the directory. Only those blocks containing data will be written onto the master tape and each block will be written as one record. Frequency dependent data will contain both real and imaginary data. If the block is empty, the data arrays are not written. This minimizes the storage requirements for the permanent master tape. In addition, only the number of points indicated on the file are written onto the master data tape. After all the records are copied from one of the random access files, the next active file is found and the records for this file are written beginning with the file header and the directory which contains 2715 words, followed by the data arrays. It should be noted that every block header and directory from a file is written on a master data tape. This allows easy regeneration of the random access file at the beginning of execution of EXEMPT.

The master data tape can be obtained at the end of execution using the EXEMPT command SAVRAN and, when it is written, the data tape is written onto a file name specified by the user or the default name, NEWDAT. At the beginning of execution, a previously existing master data tape can be read into the random access file at the time that the random access file is opened by the EXEMPT command GETRAN and requesting the user's named file or the default name OLDDAT for the master data tape. The master data tape is automatically unloaded by the EXEMPT program after the random access file is generated.

Table 4

PERMANENT DATA MASTER TAPE STRUCTURE

RECORD	NO. OF WORDS	CONTENTS
1	21	ACTIVE FILE NAMES, NUMBER OF ACTIVE FILES
2	2715	FILE HEADER AND DIRECTORY FOR TEMPORARY FILE
3	N	DATA ARRAY 1 FOR TEMPORARY FILE
4	N	DATA ARRAY 2 FOR TEMPORARY FILE
5	N	DATA ARRAY 3 FOR TEMPORARY FILE
.	.	.
.	.	.
.	.	.
J	2715	FILE HEADER AND DIRECTORY FOR NEXT ACTIVE FILE
J+1	M	DATA ARRAY 1 FOR NEXT ACTIVE FILE
J+2	M	DATA ARRAY 2 FOR NEXT ACTIVE FILE
		FOR BLOCK 1
		DATA FOR BLOCK 1
.	.	.
.	.	.
.	.	.

The random access data base will reside on LFN TAPE99. If the user has an existing EXEMPT compatible random access data base he wishes to use, it must be attached with a LFN of TAPE99. Likewise, to save an EXEMPT random access data base for later use, a LFN of TAPE99 must be used. However, if the user has attached a random access data base on LFN TAPE99, he must use the EXTEND or ALTER option at the end of the EXEMPT run in order to save the modified data base. As mentioned previously, it is recommended that the permanent master data tape system be used.

EXEMPT has the capability of writing binary data files which may be saved as external data tapes if required. In addition, specific files, in particular the master data discussed earlier in this subsection, are written. Any of the sequential data tapes written using the standard EXEMPT sequential file commands can be requested as an external tape and saved. The file names associated with these tapes are the names given by the user unless they are specified with a number between 1 and 99. In that case, the file name will be TEMPXX where the number is coded into the position XX. The tens digit is not removed for the numbers between 1 and 9 so that these files become TEMPØ1 through TEMPØ9. The random access data file is saved on disk and can be cataloged as a permanent file, but cannot be used with a magnetic tape copy. In order to save this information permanently, the user should use the GETRAN/SAVRAN option to write a permanent data master tape.

SECTION IV
USER INTERFACE REQUIREMENTS

The user may interface data directly with EXEM PT by using the rigid interface specification provided by the OPRn, SUBa, MYCOMn, and NAME CALLS discussed in the ECL of Section II. In addition, data from external or internal programs may be retrieved from or written to a peripheral sequential file using the format specified in Section III.

When using the OPRn, SUBa, MYCOMn, or NAME subroutine call, the interface must be as specified below:

Subroutine Interface

SUBROUTINE { Name / OPRn / SUBa / MYCOMn } (RSP,M,N,X,C[,A],B]])

RSP - Global real array of up to 20 values used as parameters in the subprogram

M Maximum number of points found in X, C, A, or B

N Specifies type of resultant

1 Time or no type

2 Frequency

X Independent array

C Resultant array

A Optional input array

B Optional input array

```
DIMENSION               RSP(20)
LEVEL 2, X, C [, A [, B]]
DIMENSION               X(M)
DIMENSION               C (M,N) [,A(M,N) [,B(M,N)]]
```

All arrays, except RSP, are passed via the array BLNKCM located in blank common. BLNKCM is declared a LEVEL 2 variable so that it may be loaded in LCM on the CDC 7600. As such, all other variables that reference any address in the array BLNKCM must also be declared LEVEL 2. On the CDC 6600, the LEVEL 2 declaration is ignored.

When complex data is passed through the user subroutine or to be returned through the array C, the first column (i.e., C(M,1), A(M,1), and B(M,1)) will contain the real data and the second column (i.e., C(M,2), A(M,2), and B(M,2)) will contain the imaginary data.

The user must replace the EXEMPT dummy subroutines OPRn, SUBa, and MYCOMn on the relocatable binary file by using the COPYL SCOPE utility. To use the NAME subroutine call, the user must append the entry point identified as NAME to the Direct Manipulation Processor segment using the COPYL with the append option specified.

To append the entry point for a NAME subroutine, the user must replace the EXEMPT subroutine CNGSUB on the relocatable binary file by using the COPYL SCOPE utility. The replacement must include an EXTERNAL declaration followed by a call to the EXEMPT subroutine SETUP. This sequence will add the user's subroutine name and address location to a table contained within EXEMPT. The arguments to SETUP, as shown below, include the user subroutine name in Hollerith form followed by the subroutine name. As many pairs of arguments are required as there are user-named subroutines, up to a maximum of 36 pairs.

```
SUBROUTINE CNGSUB
EXTERNAL PULSE1
CALL SETUP(6HPULSE1,PULSE1)
RETURN
END
```

1. NO INPUT ARGUMENTS OR RESULTANT SPECIFIED

The MYCOMn subroutines should be the only subroutines callable in this form. However, since the interface is the same for all user subroutine calls, any subroutine type may be called in this form. The MYCOMn subroutines will be the only user subroutines that may have an alternate name assigned. For example, consider the following ECL:

```
MYCOM1 = GTD
CALL   GTD( )
```

The MYCOMn subroutines are intended to be used as alternate user-defined commands. Since this is a transfer of control, the CALL keyword must be used in conjunction with the alternate command. Also, note the use of the null parentheses. These must be present since they delimit the start and end of a subroutine parameter list, even if a null list.

The interface for the above example and all null parameter list calls is:

```
SUBROUTINE     MYCOM1(GREAL, M, N, X, C)
```

where:

GREAL - Global real array

M - Set to zero

N - Set to zero

X - Dummy argument

C - Dummy argument

2. RESULTANT ONLY SPECIFIED

All subroutines may be used in this manner. However, there are three types of interfaces within this class and all have to do with how the independent is to be passed.

a. If the resultant is a frequency or time domain response, the resultant's associated independent is called through the subroutine interface.

b. If the resultant is classified as time or frequency (e.g., TYPE command) and the resultant has not yet been defined, the currently defined frequency or time independent is called through the subroutine interface.

c. If the resultant is not classified as frequency or time, a null array is passed as the independent with the following input parameters:

   1. M - Set to 1/3 of current BLNKCM size
   2. N - Set to 2. On return, user sets N to the following:
      N = 0, resultant is a no type
      N = 1, resultant is a time function
      N = 2, resultant is a frequency function
   3. X - On return contains independent values for N ≠ 0

The following are examples of calls to subroutines without optional input arrays.

Example 1:

```
A = SUBA( )
```

A is the resultant data set for the subroutine SUBA. A will be classified to be frequency or time domain data or a no type data by the user when the SUBA subroutine call is completed.

Example 2:

```
TIME = LIN101
B = 1./TIME
LIST B
TIME = LIN51
CALL OPR1(B)
```

B is the resultant data set for the subroutine OPR1. Since B had been classified as a time domain response in statement 2, the independent associated with B, statement 1, is called through the interface to OPR1.

3. ONE OR TWO INPUT ARRAY AND RESULTANT SPECIFIED

There are three types of subroutine interfaces within this class and, again, all have to do with the way the independent is to be passed:

a. Same as the no input array interface (1). Unless TIME or FREQ is specified as the first input array, then the specified global independent is passed through the interface and will become the independent data associated with the resultant.

b. Same as the no input array interface (2). Unless TIME or FREQ is specified as the first input array, then the specified global independent is passed through the interface and will become the independent data associated with the resultant.

c. If the resultant is not typed, the independent associated with the input array is called through the interface.

d. If the input array is not typed then a dummy independent of the same size of the input array is passed. Also, the resultant's array is set to the size of the input array. However, since the interface does not know at this point whether the resultant will be frequency or time domain, a doubly dimensioned resultant array will be passed (i.e., dimension C to (M, 2)). On the return, M will be the number of points in independent and/or resultant. N will be the type of resultant generated:

1. N = 0 - No Independent Associated
2. N = 1 - Time Independent
3. N = 2 - Frequency Independent

4. PASSING THE GLOBAL REAL ARRAY AND CURRENT INDEPENDENT ARRAY

The global real array is set externally to the calling sequence through the variables R1-R20. These variables are set prior to the subroutine calls and their values remain in effect until altered by the user. The following example will illustrate the point:

```
TIME = LIN128, 0.5E-6
R2 = 0          $ RESET GREAL(2) = 0.
R3 = 9.366E6    $ SET GREAL (3) = 9.366E6
R4 = 1.953E7    $ SET GREAL (4) = 1.953E7
TYPE Y = TIME   $ CLASSIFY Y AS A TIME FUNCTION
CALL PULSE1(Y)
LIST Y
END
```

In the above example, the user may wish to pass the TIME array and set Y internally as a time function. In this case, the TYPE command would be omitted and the subroutine call would be:

```
CALL PULSE1(Y,TIME)
```

Also, on return from subroutine PULSE1 the user need not set N = 1 to signify a time function. EXEMPT will do this on seeing TIME in the calling sequence. Additionally, the call could have been placed in an arithmetic statement such as:

```
Y = PULSE1(TIME)
    OR
TYPE Y = TIME
Y = PULSE1( )
```

A complete example of passing the global real array and the subroutine interface is shown in Figures 6 and 7.

```
   FILE DMPSIN
$
$  USER DEFINED SUBROUTINE TO CALCULATE A DAMPED SIN BY
$
$     F(T)=R2*EXP(-R3*T)*SIN(2*PI*R4*T)*U(T)
$
   TIME = LIN128,0.5E-6          $ INITIALIZE GLOBAL TIME ARRAY

$ INITIALIZE GLOBAL REAL ARRAY FOR SUBROUTINE PULSE1

   R2 = 0. $  FOR R2=0. THEN F(T) CALCULATED SUCH THAT MAX F(T)=1.0
   R3= 9.366E6  //  R4 = 1.953E7
   R5=0.    $ SPECIFIES A Q RATHER THAN A DAMPING FACTOR FOR R2 NOT EQUAL TO 0.
   R6=0.    $ OPTIONAL TIME SHIFT FOR FUNCTION SHIFT FROM 0

   Y = PULSE1(TIME)              $ USER FUNCTION PULSE1 CALLED

   TITLE Y * DAMPED SIN FUNCTION CALCULATED 1-12-77 FROM EXEMPT WITH PULSE1 *
   LIST Y
   PRTPLT Y
   END
```

Figure 6. EXEMPT Run Stream Using the Global Real Array

```
      SUBROUTINE PULSE1(RSP,M,N,T,FT)
      DIMENSION RSP(10),T(1),FT(1)
C
      A = RSP(2)
      ALPHA = RSP(3)
      F = RSP(4)
      Q = RSP(5)
      TSHIFT = RSP(6)
      NT = M
C
      DATA TWOPI/6.28318531/
      W = TWOPI*F
      IF(Q.EQ.0.0)  GO TO 1
      ALPHA = 0.5*W/Q
    1 CONTINUE
      IF(A.NE.0.0)  GO TO 2
      TP = ATAN(W/ALPHA)/W
      A = 1.0/(EXP(-ALPHA*TP)*SIN(W*TP))
    2 CONTINUE
      DO 10  I = 1,NT
      X = T(I) + TSHIFT
      FT(I) = A*EXP(-ALPHA*X)*SIN(W*X)
   10 CONTINUE
      X = - TSHIFT
      I = 0
   15 I = I + 1
      IF(T(I).GE.X)  GO TO 20
      FT(I) = 0.0
      GO TO 15
   20 CONTINUE
      RETURN
      END
```

Figure 7. User Subroutine Interface

SECTION V
COMPUTER INTERFACE REQUIREMENTS

There are three permanent files with which the user will come in contact.

a. EXEMPT Binary File

b. EXEMPT Segmentation Directive File

c. EXEMPT Absolute File

These files are more clearly defined in the EXEMPT Programmer's Manual. At this point, the user only needs to know that the binary and segmentation directive files will be needed when using the COPYL utility. The user should normally come into contact with only the absolute file. The following examples should illustrate the point.

1. EXAMPLE 1 DESCRIPTION

Example 1, shown in Figure 8, illustrates some of the EXEMPT arithmetic capabilities available. The example computes the transfer function

$$H(\omega) = \frac{1}{(\alpha + j\omega)}$$

for each frequency value. The output listing is shown in Figure 9.

2. EXAMPLE 2 DESCRIPTION

Example 2 calculates the input impedance of a shorted transmission line by using some of the standard functions available within EXEMPT. The ECL is shown in Figure 10, the LIST output is shown in Figure 11, and the resultant METAPLOT is shown in Figure 12.

For a shorted transmission line, the input impedance is given by

$$Z_{SC} = Z_o \tanh(\gamma \ell)$$

However, since hyperbolic functions are not included among the standard functions available to EXEMPT, the exponential function

```
JOB CARD.
ACCOUNT CARD.
ATTACH,ABS,PFNAME (PARAMETER LIST).          EXEMPT ABSOLUTE FILE
ABS.
& 7/8/9 (EOR)          EXEMPT ECL
      FREQ = LIN51,50.0E6
      ALPHA = 10.0E6
      JW = CMPLX(0,6.28318*FREQ)
      HW = 1/(ALPHA + JW)
      LIST HW
      END
# 6/7/8/9 (EOF)        END OF JOB
```

Figure 8. Job Stream and ECL for Example 1

LIST OF DATA SET HW

INDEX	FREQ	MAG	PHASE	INDEX	FREQ	MAG	PHASE
1	0.	1.000E-07	0.	27	2.600E+07	6.110E-09	-86.5
2	1.000E+06	8.467E-08	-32.1	28	2.700E+07	5.884E-09	-86.6
3	2.000E+06	6.227E-08	-51.5	29	2.800E+07	5.675E-09	-86.7
4	3.000E+05	4.687E-08	-62.1	30	2.900E+07	5.480E-09	-86.9
5	4.000E+06	3.697E-08	-68.3	31	3.000E+07	5.298E-09	-87.0
6	5.000E+05	3.033E-08	-72.3	32	3.100E+07	5.127E-09	-87.1
7	6.000E+06	2.564E-08	-75.1	33	3.200E+07	4.967E-09	-87.2
8	7.000E+06	2.217E-08	-77.2	34	3.300E+07	4.817E-09	-87.2
9	8.000E+05	1.951E-08	-78.7	35	3.400E+07	4.676E-09	-87.3
10	9.000E+06	1.741E-08	-80.0	36	3.500E+07	4.543E-09	-87.4
11	1.000E+07	1.572E-08	-81.0	37	3.600E+07	4.417E-09	-87.5
12	1.100E+07	1.432E-08	-81.8	38	3.700E+07	4.298E-09	-87.5
13	1.200E+07	1.315E-08	-82.4	39	3.800E+07	4.185E-09	-87.6
14	1.300E+07	1.215E-08	-83.0	40	3.900E+07	4.078E-09	-87.7
15	1.400E+07	1.130E-08	-83.5	41	4.000E+07	3.976E-09	-87.7
16	1.500E+07	1.055E-08	-83.9	42	4.100E+07	3.879E-09	-87.8
17	1.600E+07	9.898E-09	-84.3	43	4.200E+07	3.787E-09	-87.8
18	1.700E+07	9.321E-09	-84.7	44	4.300E+07	3.699E-09	-87.9
19	1.800E+07	8.808E-09	-84.9	45	4.400E+07	3.615E-09	-87.9
20	1.900E+07	8.347E-09	-85.2	46	4.500E+07	3.535E-09	-88.0
21	2.000E+07	7.933E-09	-85.5	47	4.600E+07	3.458E-09	-88.0
22	2.100E+07	7.557E-09	-85.7	48	4.700E+07	3.384E-09	-88.1
23	2.200E+07	7.215E-09	-85.9	49	4.800E+07	3.314E-09	-88.1
24	2.300E+07	6.903E-09	-86.0	50	4.900E+07	3.246E-09	-88.1
25	2.400E+07	6.617E-09	-86.2	51	5.000E+07	3.181E-09	-88.2
26	2.500E+07	6.353E-09	-86.4				

********** END OF LIST **********

Figure 9. Output for Example 1

```
JOB CARD.
ACCOUNT CARD.
REQUEST(PLTFIL,*Q)
DISPOSE(PLTFIL,*MF=PCF)                          DISPOSE PLOTS TO CALCOMP
ATTACH,ABS,PFNAME (PARAMETER LIST).              EXEMPT ABSOLUTE FILE
ABS.
& 7/8/9 (EOR)          EXEMPT ECL
$
$        IMPEDANCE OF SHORTED TRANSMISSION LINE
$
PI = 3.141592654
FREQ = 1.E6,LIN50,50.E6
W = 2*PI*FREQ
LENGTH = 2
Z0 = 50          $  CHARACTERISTIC IMPEDANCE
EPSR = 3         $  RELATIVE DIELECTRIC CONSTANT
RDC = 0.1E-3     $  DC RESISTANCE PER METER
C = 3.0E8        $  SPEED OF LIGHT

GL = CMPLX(RDC*SQRT(FREQ)/(2*Z0),W*SQRT(EPSR)/C)*LENGTH

                 $  GAMMA X LENGTH

ZSC = Z0*((EXP(GL) - EXP(-GL))/(EXP(GL) + EXP(-GL)))

                 $  ZSC = Z0*TANH(GAMMA*LENGTH)
                 $  TANH(X) = ((EXP(X)-EXP(-X))/(EXP(X)+EXP(-X)))

PLOT ZSC  TITLEY = *IMPEDANCE*
LIST  GL,ZSC
END
# 6/7/8/9 (EOF)        END OF JOB
```

Figure 10. Job Stream and ECL for Example 2

LIST OF DATA SET GL

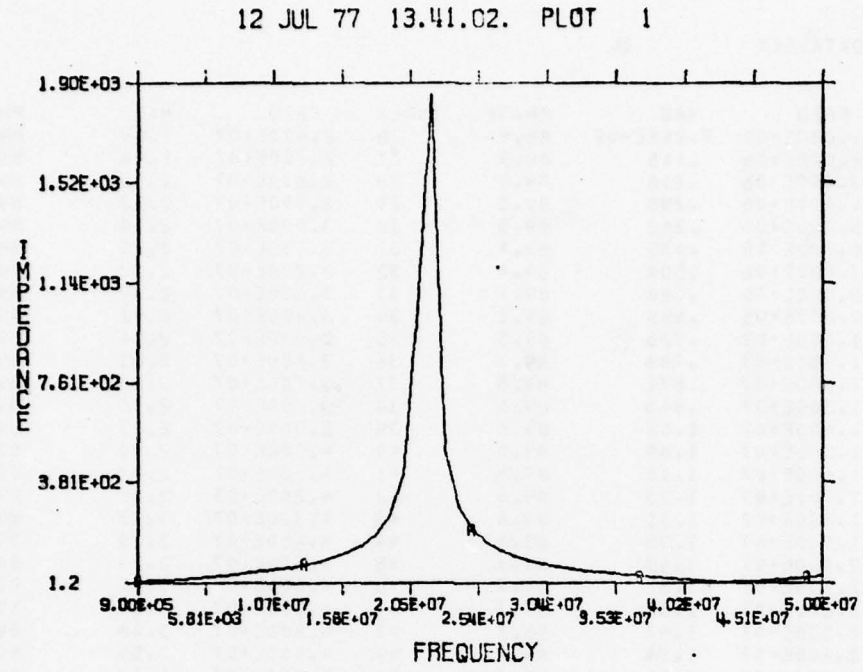
INDEX	FREQ	MAG	PHASE	INDEX	FREQ	MAG	PHASE
1	1.000E+06	7.258E-02	88.4	26	2.600E+07	1.89	89.7
2	2.000E+06	.145	88.9	27	2.700E+07	1.96	89.7
3	3.000E+06	.218	89.1	28	2.800E+07	2.03	89.7
4	4.000E+06	.290	89.2	29	2.900E+07	2.10	89.7
5	5.000E+06	.363	89.3	30	3.000E+07	2.18	89.7
6	6.000E+06	.435	89.4	31	3.100E+07	2.25	89.7
7	7.000E+06	.508	89.4	32	3.200E+07	2.32	89.7
8	8.000E+06	.580	89.4	33	3.300E+07	2.39	89.7
9	9.000E+06	.653	89.5	34	3.400E+07	2.47	89.7
10	1.000E+07	.726	89.5	35	3.500E+07	2.54	89.7
11	1.100E+07	.798	89.5	36	3.600E+07	2.61	89.7
12	1.200E+07	.871	89.5	37	3.700E+07	2.68	89.7
13	1.300E+07	.943	89.6	38	3.800E+07	2.76	89.7
14	1.400E+07	1.02	89.6	39	3.900E+07	2.83	89.7
15	1.500E+07	1.09	89.6	40	4.000E+07	2.90	89.8
16	1.600E+07	1.16	89.6	41	4.100E+07	2.97	89.8
17	1.700E+07	1.23	89.6	42	4.200E+07	3.05	89.8
18	1.800E+07	1.31	89.6	43	4.300E+07	3.12	89.8
19	1.900E+07	1.38	89.6	44	4.400E+07	3.19	89.8
20	2.000E+07	1.45	89.6	45	4.500E+07	3.26	89.8
21	2.100E+07	1.52	89.7	46	4.600E+07	3.34	89.8
22	2.200E+07	1.60	89.7	47	4.700E+07	3.41	89.8
23	2.300E+07	1.67	89.7	48	4.800E+07	3.48	89.8
24	2.400E+07	1.74	89.7	49	4.900E+07	3.56	89.8
25	2.500E+07	1.81	89.7	50	5.000E+07	3.63	89.8

LIST OF DATA SET ZSC

INDEX	FREQ	MAG	PHASE	INDEX	FREQ	MAG	PHASE
1	1.000E+06	3.64	88.4	26	2.600E+07	153.	-88.0
2	2.000E+06	7.31	88.9	27	2.700E+07	122.	-88.3
3	3.000E+06	11.1	89.1	28	2.800E+07	101.	-88.5
4	4.000E+06	14.9	89.2	29	2.900E+07	84.7	-88.6
5	5.000E+06	19.0	89.2	30	3.000E+07	72.2	-88.7
6	6.000E+06	23.3	89.3	31	3.100E+07	62.0	-88.7
7	7.000E+06	27.8	89.3	32	3.200E+07	53.6	-88.7
8	8.000E+06	32.8	89.3	33	3.300E+07	46.3	-88.7
9	9.000E+06	38.2	89.3	34	3.400E+07	40.0	-88.6
10	1.000E+07	44.3	89.3	35	3.500E+07	34.4	-88.5
11	1.100E+07	51.3	89.2	36	3.600E+07	29.3	-88.4
12	1.200E+07	59.3	89.2	37	3.700E+07	24.6	-88.2
13	1.300E+07	68.9	89.1	38	3.800E+07	20.2	-88.0
14	1.400E+07	80.6	89.0	39	3.900E+07	16.1	-87.6
15	1.500E+07	95.4	88.9	40	4.000E+07	12.2	-86.9
16	1.600E+07	115.	88.7	41	4.100E+07	8.45	-85.5
17	1.700E+07	142.	88.5	42	4.200E+07	4.78	-82.1
18	1.800E+07	184.	88.1	43	4.300E+07	1.27	-59.0
19	1.900E+07	257.	87.3	44	4.400E+07	2.62	75.3
20	2.000E+07	414.	85.7	45	4.500E+07	6.23	83.7
21	2.100E+07	1.039E+03	79.0	46	4.600E+07	9.94	85.9
22	2.200E+07	1.850E+03	-69.7	47	4.700E+07	13.8	86.9
23	2.300E+07	507.	-84.4	48	4.800E+07	17.8	87.5
24	2.400E+07	290.	-86.6	49	4.900E+07	21.9	87.8
25	2.500E+07	202.	-87.5	50	5.000E+07	26.4	88.0

********** END OF LIST **********

Figure 11. LIST Output for Example 2



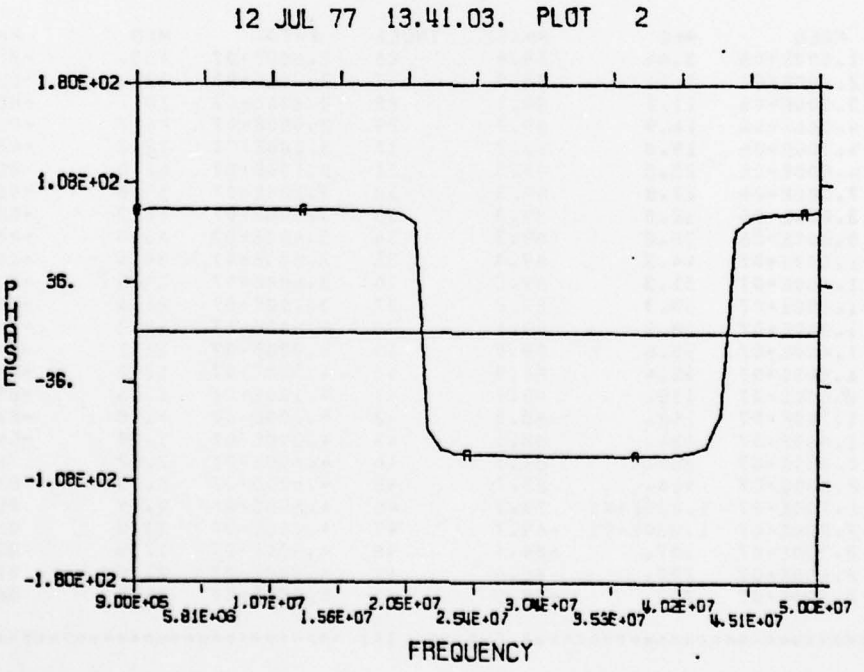


Figure 12. METAPLOT for Example 2

representation of the hyperbolic tangent is used. Note that the exponential function, EXP, can handle a real or complex argument. That is, if

$$X = \text{real number}$$

then

$$EXP(X) = e^{X}$$

but if

$$X = \alpha + j\beta$$

then

$$EXP(X) = e^{\alpha}(\cos\beta + j\sin\beta)$$

3. EXAMPLE 3 DESCRIPTION

Example 3, reference Figure 13 for ECL, is an exercise in reading a data tape generated by another program and storing the generated EXEMPT random file data base on permanent files. The LIST CURFIL output, shown in Figure 14, lists each data set as they occur on the random file including each independent. In this case of the six data sets, GTDF1-GTDF6, there are five independents also listed. There are only five independents listed because data sets GTDF3 and GTDF6 reference the same independent and, therefore, this independent is listed but once. The output for the SAVRAN command is shown in Figure 15. Figure 15 is an index to the data base saved as a result of the SAVRAN command.

AD-A066 806 BDM CORP ALBUQUERQUE N MEX F/G 20/14
EXEMPT USERS MANUAL.(U)
JAN 79 F29601-76-C-0122
UNCLASSIFIED BDM/A-76-140-TR-R2 AFWL-TR-77-207 NL

2 OF 2
AD A066806

END
DATE FILMED
5-79
DDC

```
JOB CARD.
ACCOUNT CARD.
ATTACH,TEMP01,PFNAME, (PARAMETER LIST).      EXTERNAL FILE
REQUEST,NEWDAT,*PF.
REQUEST,TAPE99,*PF.
ATTACH,ABS,PFNAME (PARAMETER LIST).          EXEMPT ABSOLUTE FILE
ABS.
CATALOG,TAPE99,PFNAME (PARAMETER LIST).      DATA BASE SAVED ON RANDOM FILE
CATALOG,NEWDAT,PFNAME (PARAMETER LIST).      DATA BASE SAVED ON SEQUENTIAL FILE
6 7/8/9 (EOR)          EXEMPT ECL
$
$      EXAMPLE TO DEMONSTRATE EXEMPT LINKAGE TO A FILE GENERATED BY
$      ANOTHER PROGRAM IN EXEMPT FORMAT.
$
$      ALSO, THE GENERATED DATA BASE IS SAVED ON A PERMANENT FILE FOR
$      LATER USE.
$
   REWIND 1
   FILE GTDB1
   READ GTDF1 FILE=1
   READ GTDF2 FILE=1
   READ GTDF3 FILE=1
   READ GTDF4 FILE=1
   READ GTDF5 FILE=1
   READ GTDF6 FILE=1
   LIST CURFIL
   SAVRAN
   END
# 6/7/8/9 (EOF)        END OF JOB
```

Figure 13. Job Stream and ECL for Example 3

```
****   C U R R E N T   F I L E   I N D E X   ****


    FILE INDEX=   2     FILE NAME= GTDB1
         DATA SET TITLE

!!!! GTDH RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. JZ  PHI MODE  09.14.47. 01/05/77 B-1 DATA F
    DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTDF1          FREQUENCY             26          1             2

LIST OF DATA SET      GTDF1

 GTDH RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. JZ  PHI MODE  09.14.47. 01/05/77 B-1 DATA F

INDEX     FREQ        REAL        IMAG     INDEX     FREQ        REAL        IMAG
    1  -3.678E-05 -8.250E-05   50.0        14  -1.668E-06 -3.530E-05   180.
    2   6.802E-05  6.031E-05   60.0        15   2.224E-05  1.190E-05   190.
    3  -9.607E-05 -9.642E-06   70.0        16  -3.182E-05 -1.479E-05   200.
    4   3.846E-05 -2.891E-05   80.0        17   2.008E-05 -9.433E-06   210.
    5  -4.280E-05  5.814E-05   90.0        18  -1.514E-05  1.970E-05   220.
    6  -1.010E-05 -3.205E-05  100.         19   6.654E-06 -1.644E-05   230.
    7   2.175E-05  2.195E-05  110.         20   1.329E-05  9.884E-06   240.
    8  -1.980E-05 -7.347E-06  120.         21  -4.078E-06 -6.843E-06   250.
    9  -4.157E-07 -2.421E-05  130.         22   4.965E-06 -1.024E-05   260.
   10  -1.132E-06  6.860E-06  140.         23   1.358E-06  7.064E-06   270.
   11  -3.048E-05 -7.405E-07  150.         24  -1.093E-05 -1.001E-06   280.
   12   1.804E-05 -1.083E-05  160.         25   1.543E-05 -4.144E-07   290.
   13  -5.585E-06  2.662E-05  170.         26  -4.886E-06  1.442E-05   300.

         DATA SET TITLE

    DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          FREQ           NONE                  26          2             0

LIST OF DATA SET       FREQ


NDEX     VALUE    NDEX     VALUE    NDEX     VALUE    NDEX     VALUE    NDEX     VALUE
   1 -3.678E-05     2  6.802E-05     3 -9.607E-05     4  3.846E-05     5 -4.280E-05
   6 -1.010E-05     7  2.175E-05     8 -1.980E-05     9 -4.157E-07    10 -1.132E-06
  11 -3.048E-05    12  1.804E-05    13 -5.585E-06    14 -1.668E-06    15  2.224E-05
  16 -3.182E-05    17  2.008E-05    18 -1.514E-05    19  6.654E-06    20  1.329E-05
  21 -4.078E-06    22  4.965E-06    23  1.358E-06    24 -1.093E-05    25  1.543E-05
  26 -4.886E-06
         DATA SET TITLE

!!!! GTDH RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. JP  PHI MODE  09.14.47. 01/05/77 B-1 DATA F
    DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTDF2          FREQUENCY             26          3             4

LIST OF DATA SET      GTDF2

 GTDH RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. JP  PHI MODE  09.14.47. 01/05/77 B-1 DATA F

INDEX     FREQ        REAL        IMAG     INDEX     FREQ        REAL        IMAG
    1  -5.916E-03 -7.251E-05   50.0        14  -5.928E-03  5.057E-04   180.
    2  -5.561E-03  5.146E-04   60.0        15  -5.225E-03  8.106E-04   190.
    3  -5.021E-03  4.790E-04   70.0        16  -4.598E-03  3.706E-04   200.
    4  -4.629E-03 -1.214E-04   80.0        17  -4.620E-03 -3.700E-04   210.
    5  -5.039E-03 -6.905E-04   90.0        18  -5.224E-03 -7.410E-04   220.
    6  -5.809E-03 -6.231E-04  100.         19  -5.854E-03 -4.868E-04   230.
    7  -6.101E-03  1.083E-04  110.         20  -6.000E-03  1.861E-04   240.
    8  -5.671E-03  7.023E-04  120.         21  -5.525E-03  6.591E-04   250.
    9  -4.930E-03  7.116E-04  130.         22  -4.899E-03  5.223E-04   260.
   10  -4.473E-03  8.784E-05  140.         23  -4.675E-03 -3.387E-05   270.
   11  -4.787E-03 -6.637E-04  150.         24  -4.987E-03 -5.213E-04   280.
   12  -5.560E-03 -7.811E-04  160.         25  -5.550E-03 -5.356E-04   290.
   13  -6.083E-03 -2.161E-04  170.         26  -5.860E-03 -8.014E-05   300.
```

Figure 14. LIST CURFIL Output

```
        DATA SET TITLE

    DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          FREQ            NONE               26            4            0

LIST OF DATA SET       FREQ


NDEX     VALUE   NDEX     VALUE   NDEX     VALUE   NDEX     VALUE   NDEX     VALUE
   1 -5.916E-03     2 -5.561E-03     3 -5.021E-03     4 -4.629E-03     5 -5.039E-03
   6 -5.809E-03     7 -6.101E-03     8 -5.671E-03     9 -4.930E-03    10 -4.473E-03
  11 -4.787E-03    12 -5.568E-03    13 -6.083E-03    14 -5.928E-03    15 -5.229E-03
  16 -4.598E-03    17 -4.620E-03    18 -5.224E-03    19 -5.854E-03    20 -6.000E-03
  21 -5.525E-03    22 -4.699E-03    23 -4.675E-03    24 -4.987E-03    25 -5.550E-03
  26 -5.860E-03
        DATA SET TITLE

:::: GTDM RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. PS  PHI MODE  09.14.47. 01/05/77 B-1 DATA F
    DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTDF3          FREQUENCY            26            5            6

LIST OF DATA SET       GTDF3

 GTDM RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. PS  PHI MODE  09.14.47. 01/05/77 B-1 DATA F

INDEX     FREQ        REAL        IMAG     INDEX     FREQ        REAL        IMAG
    1  -5.280E-13  1.076E-12   50.0        14  -1.394E-14 -3.933E-13   180.
    2   7.266E-14 -8.513E-13   60.0        15   1.436E-14 -1.110E-12   190.
    3  -3.288E-13 -1.516E-12   70.0        16  -1.243E-13 -7.020E-13   200.
    4   2.720E-13 -9.008E-13   80.0        17   3.500E-14  2.534E-13   210.
    5   3.522E-13  8.556E-13   90.0        18   1.477E-13  9.175E-13   220.
    6   1.575E-13  1.403E-12  100.         19   1.530E-13  7.656E-13   230.
    7   1.290E-13  4.758E-13  110.         20   1.866E-13 -9.689E-14   240.
    8  -1.820E-13 -7.787E-13  120.         21   4.708E-14 -7.528E-13   250.
    9  -1.544E-13 -1.276E-12  130.         22  -2.657E-14 -6.984E-13   260.
   10   2.028E-14 -6.015E-13  140.         23  -9.253E-15 -4.459E-14   270.
   11   1.089E-13  6.314E-13  150.         24   1.385E-14  6.056E-13   280.
   12   2.434E-13  1.213E-12  160.         25   1.203E-13  6.346E-13   290.
   13   2.148E-13  6.412E-13  170.         26   1.768E-13  8.985E-14   300.

        DATA SET TITLE

    DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          FREQ            NONE               26            6            0

LIST OF DATA SET       FREQ


NDEX     VALUE   NDEX     VALUE   NDEX     VALUE   NDEX     VALUE   NDEX     VALUE
   1 -5.280E-13     2  7.266E-14     3 -3.288E-13     4  2.720E-13     5  3.522E-13
   6  1.575E-13     7  1.290E-13     8 -1.820E-13     9 -1.544E-13    10  2.028E-14
  11  1.089E-13    12  2.434E-13    13  2.148E-13    14 -1.394E-14    15  1.436E-14
  16 -1.243E-13    17  3.500E-14    18  1.477E-13    19  1.530E-13    20  1.866E-13
  21  4.708E-14    22 -2.657E-14    23 -9.253E-15    24  1.385E-14    25  1.203E-13
  26  1.768E-13
```

Figure 14. LIST CURFIL Output (Continued)

DATA SET TITLE

1111 GTDM RUN-PHI= 0 THETA= 90 OBSERVATION POINT-Z= 225. ANGLE= 0. JZ THETA MODE 09.14.47. 01/05/77 B-1 DATA F

DATA SET NAME	RESPONSE TYPE	NUM OF PTS	INDEX	IND INDEX
GTDF4	FREQUENCY	26	7	8

LIST OF DATA SET GTDF4

GTDM RUN-PHI= 0 THETA= 90 OBSERVATION POINT-Z= 225. ANGLE= 0. JZ THETA MODE 09.14.47. 01/05/77 B-1 DATA F

INDEX	FREQ	REAL	IMAG	INDEX	FREQ	REAL	IMAG
1	4.866E-03	-3.141E-06	50.0	14	5.258E-03	-1.567E-05	180.
2	5.415E-03	9.151E-05	60.0	15	5.323E-03	-4.090E-05	190.
3	5.107E-03	1.373E-04	70.0	16	5.233E-03	2.961E-06	200.
4	5.240E-03	-1.550E-04	80.0	17	5.255E-03	-3.406E-05	210.
5	5.209E-03	1.106E-04	90.0	18	5.278E-03	4.535E-05	220.
6	5.141E-03	-3.934E-06	100.	19	5.273E-03	1.061E-05	230.
7	5.301E-03	2.920E-05	110.	20	5.326E-03	-1.372E-06	240.
8	5.253E-03	3.185E-05	120.	21	5.299E-03	-1.332E-05	250.
9	5.255E-03	-4.415E-05	130.	22	5.278E-03	-3.257E-05	260.
10	5.266E-03	-2.366E-05	140.	23	5.278E-03	-1.185E-05	270.
11	5.213E-03	3.453E-05	150.	24	5.265E-03	2.007E-05	280.
12	5.253E-03	-8.846E-06	160.	25	5.292E-03	7.272E-06	290.
13	5.311E-03	5.222E-05	170.	26	5.322E-03	2.288E-05	300.

DATA SET TITLE

DATA SET NAME	RESPONSE TYPE	NUM OF PTS	INDEX	IND INDEX
FREQ	NONE	26	8	0

LIST OF DATA SET FREQ

NDEX	VALUE	NDEX	VALUE	NDEX	VALUE	NDEX	VALUE	NDEX	VALUE
1	4.866E-03	2	5.415E-03	3	5.107E-03	4	5.240E-03	5	5.209E-03
6	5.141E-03	7	5.301E-03	8	5.253E-03	9	5.255E-03	10	5.266E-03
11	5.213E-03	12	5.253E-03	13	5.311E-03	14	5.258E-03	15	5.323E-03
16	5.233E-03	17	5.255E-03	18	5.278E-03	19	5.273E-03	20	5.326E-03
21	5.299E-03	22	5.278E-03	23	5.278E-03	24	5.265E-03	25	5.292E-03
26	5.322E-03								

DATA SET TITLE

1111 GTDM RUN-PHI= 0 THETA= 90 OBSERVATION POINT-Z= 225. ANGLE= 0. JP THETA MODE 09.14.47. 01/05/77 B-1 DATA F

DATA SET NAME	RESPONSE TYPE	NUM OF PTS	INDEX	IND INDEX
GTDF5	FREQUENCY	26	9	10

LIST OF DATA SET GTDF5

GTDM RUN-PHI= 0 THETA= 90 OBSERVATION POINT-Z= 225. ANGLE= 0. JP THETA MODE 09.14.47. 01/05/77 B-1 DATA F

INDEX	FREQ	REAL	IMAG	INDEX	FREQ	REAL	IMAG
1	-6.250E-07	1.436E-06	50.0	14	-5.565E-06	1.451E-05	180.
2	-3.972E-07	5.471E-08	60.0	15	5.884E-06	1.138E-05	190.
3	2.116E-06	6.292E-07	70.0	16	1.212E-05	2.179E-06	200.
4	-8.362E-05	-3.512E-05	80.0	17	7.650E-06	-8.279E-06	210.
5	-2.485E-05	-1.279E-05	90.0	18	-2.088E-06	-1.122E-05	220.
6	-2.043E-05	1.134E-05	100.	19	-1.016E-05	-5.682E-06	230.
7	-1.778E-05	2.034E-05	110.	20	-1.156E-05	3.025E-06	240.
8	1.667E-05	1.275E-05	120.	21	-5.373E-06	1.064E-05	250.
9	2.000E-05	-5.064E-06	130.	22	3.601E-06	1.070E-05	260.
10	9.155E-06	-1.873E-05	140.	23	9.284E-06	3.796E-06	270.
11	-6.289E-05	-1.901E-05	150.	24	8.469E-06	-3.743E-06	280.
12	-1.776E-05	-8.353E-06	160.	25	1.813E-06	-8.280E-06	290.
13	-1.615E-05	5.880E-06	170.	26	-4.991E-06	-6.573E-06	300.

Figure 14. LIST CURFIL Output (Continued)

```
        DATA SET TITLE

    DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          FREQ            NONE                26          10            0

LIST OF DATA SET       FREQ

NDEX     VALUE    NDEX     VALUE    NDEX     VALUE    NDEX     VALUE    NDEX     VALUE
   1 -6.250E-07     2 -3.972E-07     3  2.116E-06     4 -8.362E-06     5 -2.485E-05
   6 -2.043E-05     7 -1.778E-06     8  1.667E-05     9  2.000E-05    10  9.159E-06
  11 -6.289E-06    12 -1.776E-05    13 -1.615E-05    14 -5.565E-06    15  9.064E-06
  16  1.212E-05    17  7.650E-06    18 -2.088E-06    19 -1.010E-05    20 -1.196E-05
  21 -5.373E-06    22  3.601E-06    23  9.284E-06    24  8.469E-06    25  1.813E-06
  26 -4.991E-06
        DATA SET TITLE

!!!! GTOM RUN-PHI=    0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. PS THETA MODE 09.14.47. 01/05/77 B-1 DATA F
    DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTOF6           FREQUENCY            26          11            6

LIST OF DATA SET       GTOF6

 GTOM RUN-PHI=    0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. PS THETA MODE 09.14.47. 01/05/77 B-1 DATA F

INDEX     FREQ         REAL         IMAG     INDEX     FREQ         REAL         IMAG
    1  -5.280E-13 -4.419E-13   50.0        14  -1.334E-14 -1.781E-13   180.
    2   7.266E-14  1.007E-13   60.0        15   1.436E-14 -1.063E-13   190.
    3  -3.288E-13  5.040E-13   70.0        16  -1.243E-13  8.165E-14   200.
    4   2.720E-13  8.112E-14   80.0        17   3.500E-14  6.806E-14   210.
    5   3.522E-13  2.816E-13   90.0        18   1.477E-13  1.449E-13   220.
    6   1.575E-13 -2.162E-13  100.         19   1.530E-13 -7.940E-15   230.
    7   1.290E-13 -2.385E-13  110.         20   1.866E-13 -8.466E-14   240.
    8  -1.820E-13 -6.298E-14  120.         21   4.708E-14 -1.099E-13   250.
    9  -1.544E-13  3.118E-14  130.         22  -2.657E-14 -7.663E-14   260.
   10   2.028E-14  1.830E-13  140.         23  -9.253E-15  2.564E-14   270.
   11   1.089E-13  2.175E-13  150.         24   1.385E-14  9.928E-14   280.
   12   2.434E-13 -4.055E-14  160.         25   1.203E-13  6.012E-14   290.
   13   2.148E-13 -5.786E-14  170.         26   1.768E-13  3.506E-14   300.

********** END OF FILE LIST **********

********** END OF LIST **********
```

Figure 14. LIST CURFIL Output (Concluded)

```
••••    R A N D O M    T O    S E Q U E N T I A L    I N D E X    ••••


   FILE INDEX=   1     FILE NAME= STEMP<

••••••••••••••••••••••••••••••••• END OF FILE LIST •••••••••••••••••••••••••••••••••


   FILE INDEX=   2     FILE NAME= GTDB1
        DATA SET TITLE

:::: GTDM RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. JZ  PHI MODE  09.14.47. 01/05/77 B-1 DATA F
   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTDF1          FREQUENCY             26          1              2
        DATA SET TITLE

   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          FREQ            NONE                26          2              0
        DATA SET TITLE

:::: GTDM RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. JP  PHI MODE  09.14.47. 01/05/77 B-1 DATA F
   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTDF2          FREQUENCY             26          3              4
        DATA SET TITLE

   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          FREQ            NONE                26          4              0
        DATA SET TITLE

:::: GTDM RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. PS  PHI MODE  09.14.47. 01/05/77 B-1 DATA F
   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTDF3          FREQUENCY             26          5              6
        DATA SET TITLE

   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          FREQ            NONE                26          6              0
        DATA SET TITLE

:::: GTDM RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. JZ THETA MODE 09.14.47. 01/05/77 B-1 DATA F
   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTDF4          FREQUENCY             26          7              8
        DATA SET TITLE

   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          FREQ            NONE                26          8              0
        DATA SET TITLE

:::: GTDM RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. JP THETA MODE 09.14.47. 01/05/77 B-1 DATA F
   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTDF5          FREQUENCY             26          9             10
        DATA SET TITLE

   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          FREQ            NONE                26         10              0
        DATA SET TITLE

:::: GTDM RUN-PHI=   0 THETA=  90 OBSERVATION POINT-Z= 225. ANGLE=  0. PS THETA MODE 09.14.47. 01/05/77 B-1 DATA F
   DATA SET NAME      RESPONSE TYPE      NUM OF PTS      INDEX      IND INDEX
          GTDF6          FREQUENCY             26         11              5

••••••••••••••••••••••••••••••••• END OF FILE LIST •••••••••••••••••••••••••••••••••


••••••••••••••••••••••••••••••••• END OF LIST •••••••••••••••••••••••••••••••••
```

Figure 15. Index From SAVRAN Command

4. EXAMPLE 4 DESCRIPTION

Figure 16 is the job stream and ECL for linking a user-defined subroutine to EXEMPT. If the user wishes to use a SUBa, OPRn, or a MYCOMn then the job stream is exactly the same. The SCOPE COPYL utility modifies or appends any new routines to the present program. The output of this example is presented in Figures 17 to 19.

```
JOB CARD.
ACCOUNT CARD.
FTN.
ATTACH,OLD,PFNAME (PARAMETER LIST).          EXEMPT BINARY
COPYL,OLD,LGO,EXMBIN,,A.
LIBRARY(METALIB)
ATTACH,SEG,PFNAME (PARAMETER LIST).          SEGMENT DIRECTIVES
SEGLOAD(I=SEG)
LOAD(EXMBIN)
NOGO.
REQUEST(PLTFIL,*Q)
DISPOSE(PLTFIL,*NF=PCF)                      DISPOSE PLOTS TO CALCOMP
ABS.
7/8/9 (EOR)          INPUT FOR FTN CARD
      SUBROUTINE CNGSUB
      EXTERNAL PULSE1
      CALL SETUP(6HPULSE1,PULSE1)
      RETURN
      END
      SUBROUTINE PULSE1(RSP,M,N,T,FT)
      DIMENSION RSP(10),T(1),FT(1)
C
      A = RSP(2)
      ALPHA = RSP(3)
      F = RSP(4)
      Q = RSP(5)
      TSHIFT = RSP(6)
      NT = M
C
      DATA TWOPI/6.28318531/
      W = TWOPI*F
      IF(Q.EQ.0.0)  GO TO 1
      ALPHA = 0.5*W/Q
    1 CONTINUE
      IF(A.NE.0.0)  GO TO 2
      TP = ATAN(W/ALPHA)/W
      A = 1.0/(EXP(-ALPHA*TP)*SIN(W*TP))
    2 CONTINUE
      DO 10  I = 1,NT
      X = T(I) + TSHIFT
      FT(I) = A*EXP(-ALPHA*X)*SIN(W*X)
   10 CONTINUE
      X = - TSHIFT
      I = 0
   15 I = I + 1
      IF(T(I).GE.X)  GO TO 20
      FT(I) = 0.0
      GO TO 15
   20 CONTINUE
      RETURN
      END
7/8/9 (EOR)          EXEMPT ECL
$*************************************************************************$
$                                                                         $
$                  DEMONSTRATES USER SUBROUTINE CAPABILITIES              $
$                                                                         $
$*************************************************************************$
   FILE DMPSIN
$
$  USER DEFINED SUBROUTINE TO CALCULATE A DAMPED SIN BY
$
$     F(T)=R2*EXP(-R3*T)*SIN(2*PI*R4*T)*U(T)
$
   TIME = LIN128,0.5E-6          $ INITIALIZE GLOBAL TIME ARRAY

$ INITIALIZE GLOBAL REAL ARRAY FOR SUBROUTINE PULSE1

   R2 = 0. $  FOR R2=0. THEN F(T) CALCULATED SUCH THAT MAX F(T)=1.0
   R3= 9.366E6  //  R4 = 1.953E7
   R5=0.    $ SPECIFIES A Q RATHER THAN A DAMPING FACTOR FOR R2 NOT EQUAL TO 0.
   R6=0.    $ OPTIONAL TIME SHIFT FOR FUNCTION SHIFT FROM 0

   Y = PULSE1(TIME)              $ USER FUNCTION PULSE1 CALLED

   TITLE Y * DAMPED SIN FUNCTION CALCULATED 1-12-77 FROM EXEMPT WITH PULSE1 *
   LIST Y
   PRTPLT Y
PLOT Y
   END
6/7/8/9 (EOF)        END OF JOB
```

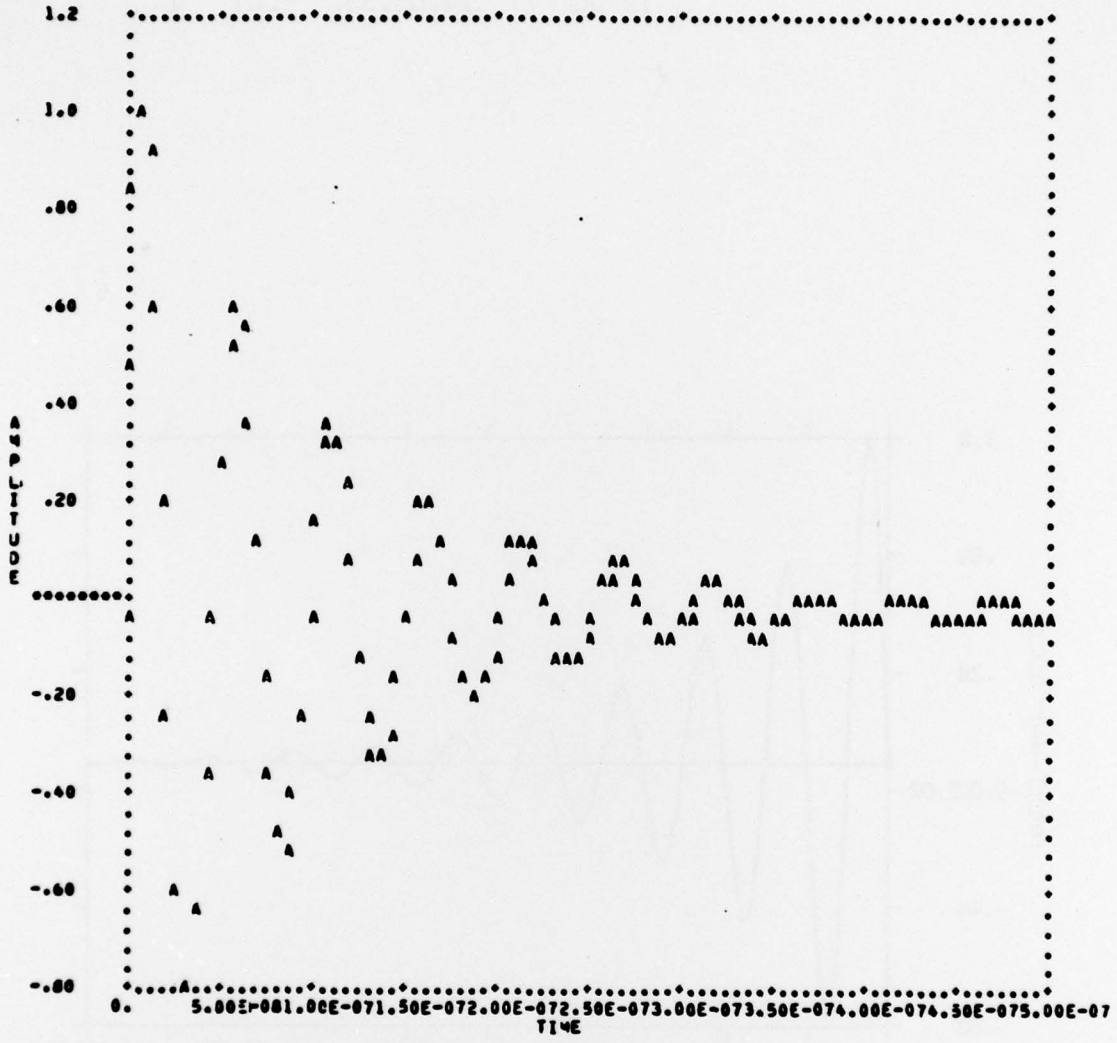
Figure 16. Job Stream and ECL for Example 4

```
LIST OF DATA SET          Y
 DAMPED SIN FUNCTION CALCULATED 1-12-77 FROM EXEMPT WITH PULSE1

NDEX     TIME        VALUE     NDEX     TIME        VALUE     NDEX     TIME        VALUE
   1     0.          0.          44   1.693E-07     .216       87   3.386E-07 -3.063E-02
   2   3.937E-09     .303        45   1.732E-07     .149       88   3.425E-07 -4.220E-02
   3   7.874E-09     .359        46   1.772E-07   5.314E-02    89   3.465E-07 -4.358E-02
   4   1.181E-08     .299        47   1.811E-07  -4.740E-02    90   3.504E-07 -3.519E-02
   5   1.575E-08     .307        48   1.850E-07   -.130       91   3.543E-07 -1.959E-02
   6   1.969E-08     .521        49   1.890E-07   -.178       92   3.583E-07 -7.470E-04
   7   2.362E-08     .217        50   1.929E-07   -.183       93   3.622E-07  1.692E-02
   8   2.756E-08    -.207        51   1.969E-07   -.147       94   3.661E-07  2.957E-02
   9   3.150E-08    -.554        52   2.009E-07  -8.128E-02   95   3.701E-07  3.477E-02
  10   3.543E-08    -.754        53   2.047E-07  -1.798E-03   96   3.740E-07  3.188E-02
  11   3.937E-08    -.772        54   2.087E-07   7.243E-02   97   3.780E-07  2.212E-02
  12   4.331E-08    -.518        55   2.126E-07     .125      98   3.819E-07  8.141E-03
  13   4.724E-08    -.337        56   2.165E-07     .147      99   3.858E-07 -6.646E-03
  14   5.118E-08 -1.894E-03      57   2.205E-07     .134     100   3.898E-07 -1.891E-02
  15   5.512E-08     .310        58   2.244E-07   9.234E-02  101   3.937E-07 -2.610E-02
  16   5.906E-08     .531        59   2.283E-07   3.325E-02  102   3.976E-07 -2.699E-02
  17   6.299E-08     .518        60   2.323E-07  -2.901E-02  103   4.016E-07 -2.183E-02
  18   6.693E-08     .362        61   2.362E-07  -8.041E-02  104   4.055E-07 -1.219E-02
  19   7.087E-08     .385        62   2.402E-07   -.110      105   4.094E-07 -5.286E-04
  20   7.480E-08     .136        63   2.441E-07   -.114      106   4.134E-07  1.042E-02
  21   7.874E-08    -.126        64   2.480E-07  -9.145E-02  107   4.173E-07  1.828E-02
  22   8.268E-08    -.342        65   2.520E-07  -5.058E-02  108   4.213E-07  2.152E-02
  23   8.661E-08    -.466        66   2.559E-07  -1.392E-03  109   4.252E-07  1.976E-02
  24   9.055E-08    -.478        67   2.599E-07   4.461E-02  110   4.291E-07  1.373E-02
  25   9.449E-08    -.383        68   2.638E-07   7.744E-02  111   4.331E-07  5.092E-03
  26   9.843E-08    -.210        69   2.677E-07   9.075E-02  112   4.370E-07 -4.065E-03
  27   1.024E-07 -2.345E-03      70   2.717E-07   8.298E-02  113   4.409E-07 -1.167E-02
  28   1.063E-07     .191        71   2.756E-07   5.734E-02  114   4.449E-07 -1.614E-02
  29   1.102E-07     .328        72   2.795E-07   2.080E-02  115   4.488E-07 -1.672E-02
  30   1.142E-07     .383        73   2.835E-07  -1.775E-02  116   4.528E-07 -1.354E-02
  31   1.181E-07     .349        74   2.874E-07  -4.963E-02  117   4.567E-07 -7.584E-03
  32   1.220E-07     .239        75   2.913E-07  -6.823E-02  118   4.606E-07 -3.682E-04
  33   1.260E-07   8.491E-02     76   2.953E-07  -7.036E-02  119   4.646E-07  6.416E-03
  34   1.299E-07 -7.744E-02      77   2.992E-07  -5.673E-02  120   4.685E-07  1.129E-02
  35   1.339E-07    -.211        78   3.031E-07  -3.148E-02  121   4.724E-07  1.332E-02
  36   1.378E-07    -.288        79   3.071E-07  -1.034E-03  122   4.764E-07  1.224E-02
  37   1.417E-07    -.296        80   3.110E-07   2.747E-02  123   4.803E-07  8.528E-03
  38   1.457E-07    -.238        81   3.150E-07   4.796E-02  124   4.843E-07  3.185E-03
  39   1.496E-07    -.131        82   3.189E-07   5.617E-02  125   4.882E-07 -2.486E-03
  40   1.535E-07 -2.178E-03      83   3.229E-07   5.143E-02  126   4.921E-07 -7.202E-03
  41   1.575E-07     .118        84   3.269E-07   3.561E-02  127   4.961E-07 -9.984E-03
  42   1.614E-07     .203        85   3.307E-07   1.302E-02  128   5.000E-07 -1.035E-02
  43   1.654E-07     .237        86   3.346E-07  -1.086E-02

********************************** END OF LIST **********************************
```

Figure 17. LIST Output of Example 4



DATA SET Y
TITLE = DAMPED SIN FUNCTION CALCULATED 1-12-77 FROM EXEMPT WITH PULSE1

A - MAG Y

Figure 18. PRTPLT Output of Example 4

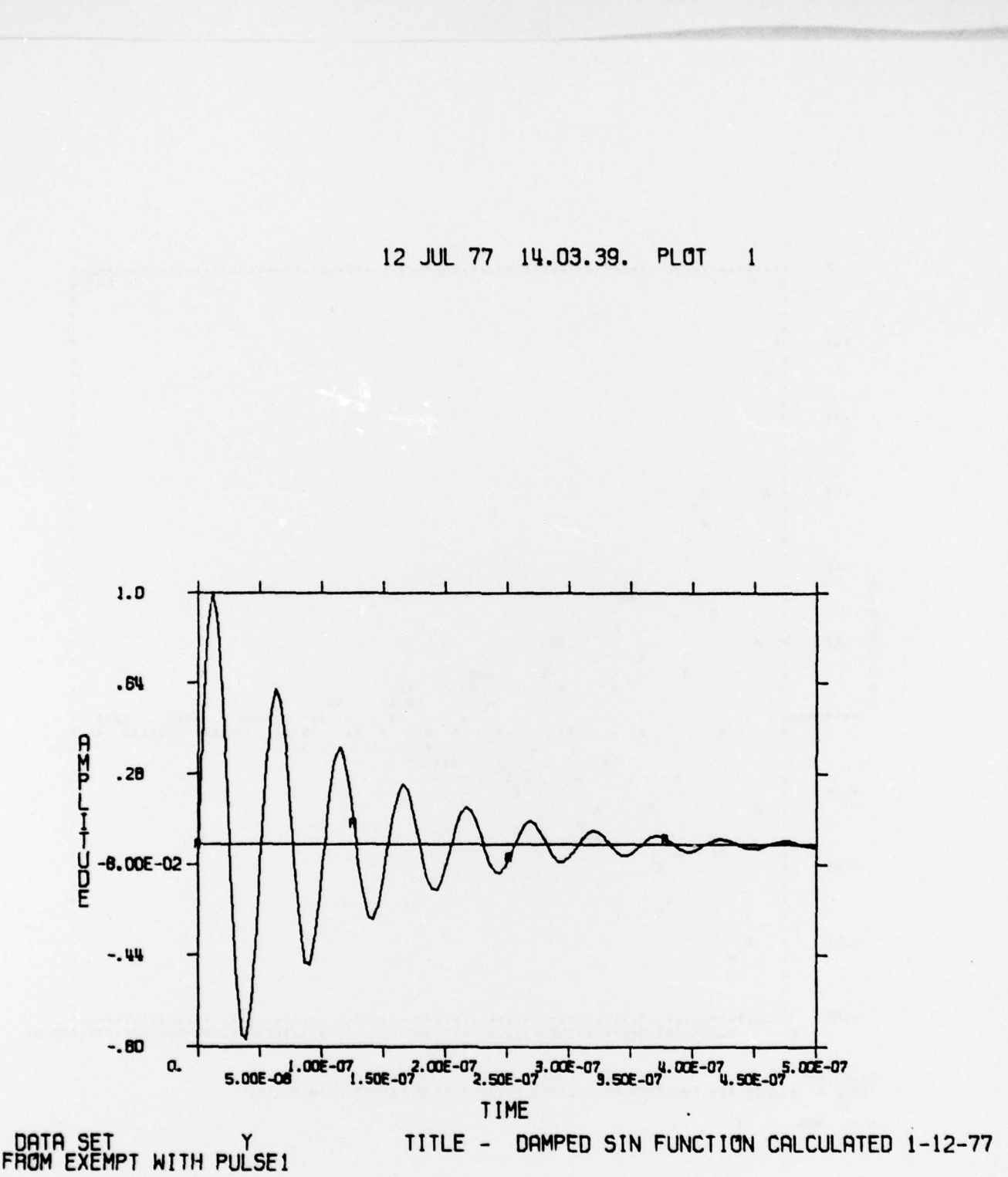


Figure 19. METAPLOT Output for Example 4