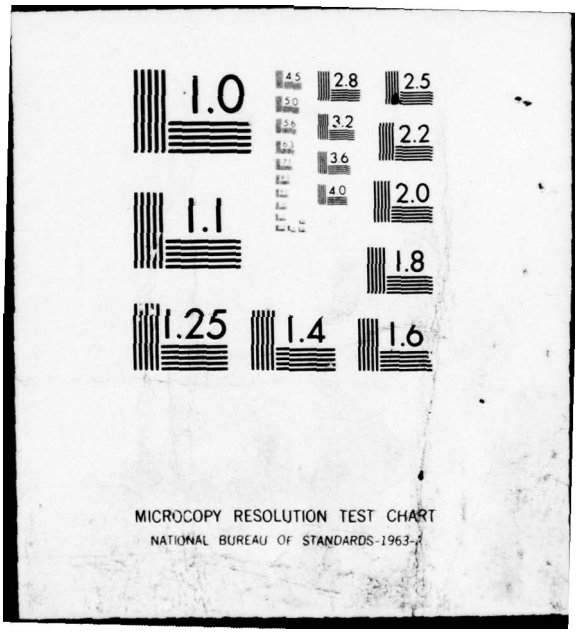
AD-A065 270 BELL HELICOPTER TEXTRON FORT WORTH TEX F/G 1/2
OPERATIONAL LOADS SURVEY - DATA MANAGEMENT SYSTEM. VOLUME II. S--ETC(U)
JAN 79 R B PHILBRICK, A L EUBANKS DAAJ02-77-C-0053
UNCLASSIFIED BHT-299-099-871-VOL-2 USARTL-TR-78-52B NL

1 OF 2
AD A065 270


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

USARTL-TR-78-52B

(12) LEVEL III

A065129

OPERATIONAL LOADS SURVEY - DATA MANAGEMENT SYSTEM
Volume II - Systems Manual

Richard B. Philbrick
Alfred L. Eubanks

Bell Helicopter Textron
P.O. Box 482
Fort Worth, Texas 76101

January 1979

Final Report

Prepared for

APPLIED TECHNOLOGY LABORATORY
U. S. ARMY RESEARCH AND TECHNOLOGY LABORATORIES (AVRADCOM)
Fort Eustis, Va. 23604

APPLIED TECHNOLOGY LABORATORY POSITION STATEMENT

This report has been reviewed by the Applied Technology Laboratory, US Army Research and Technology Laboratories (AVRADCOM), and is considered to be technically sound.

This program was initiated to design and implement a computer software system for data management of the Operational Loads Survey test data base, allowing the user to select the data to be retrieved, to specify certain processes by which the data may be reduced and/or analyzed, and to choose the mode by which the data will be presented. User options include interactive or batch processing with output options of printing and graphic displays using Tektronix and Calcomp devices.

This program was conducted under the technical management of D. J. Merkley of the Aeronautical Technology Division.

(9) Final rept.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(19) **REPORT DOCUMENTATION PAGE**

READ INSTRUCTIONS BEFORE COMPLETING FORM

(18) 1. REPORT NUMBER: USARTL TR-78-52B

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

(6) 4. TITLE (and Subtitle): OPERATIONAL LOADS SURVEY - DATA MANAGEMENT SYSTEM. Volume II. Systems Manual.

5. TYPE OF REPORT & PERIOD COVERED: Technical Report

(14) 6. PERFORMING ORG. REPORT NUMBER: BHT-299-099-871-VOL-2

(10) 7. AUTHOR(s): Richard B. Philbrick, Alfred L. Eubanks

(15) 8. CONTRACT OR GRANT NUMBER(s): DAAJ02-77-C-0053

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Bell Helicopter Textron, Post Office Box 482, Fort Worth, Texas 76101

(16) 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 62209A 1L262209AH76 00 211 EK

11. CONTROLLING OFFICE NAME AND ADDRESS: Applied Technology Laboratory, US Army Research and Technology Laboratories, Fort Eustis, Virginia 23604

(11) 12. REPORT DATE: January 1979

13. NUMBER OF PAGES: 126 (17) 00

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)

(12) 128 p.

15. SECURITY CLASS. (of this report): Unclassified

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)

AD65 129

18. SUPPLEMENTARY NOTES

Volume II of a two-volume report.

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

Helicopters, Data Bases, Data Reduction, Data Management, Computer Graphics, Interactive Graphics, Signal Processing, Mathematical Analysis.

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

The Operational Loads Survey/Data Management System (OLS/DMS) was designed and programmed as a computer software tool for data management and processing of the Operational Loads Survey (OLS) test data base. With limited modification, the OLS/DMS will accommodate other large, time based, test data bases. The system transfers selected test data to a large, direct access disc file and maintains the data on a semi-permanent → next page

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

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. Abstract (Continued)
basis. Data are retrieved from this file, processed, and displayed interactively or in batch. Plot output is generated on a Tektronix 4014 or an incremental plotter (e.g., Calcomp).

A small sample of available processing options includes amplitude spectrum, harmonic analysis, digital filtering, blade static pressure coefficient, and blade normal force coefficient. This program will accommodate data from multiple sensors simultaneously for processing of functions with two geometric independent variables (e.g., chord and radius). Output options include printout, single or multiple curve X-Y plots, contour plots, and pictorial representation of 3-dimensional surfaces. User input is free field with errors diagnosed where possible. Prompting for available command input is optional.

This report is in two volumes. Volume I is a user's manual and Volume II is a systems manual for assistance in program maintenance, modification, and/or installation.

PREFACE

The AH-1G Helicopter Aerodynamic and Structural Loads Survey conducted under Contract DAAJ02-73-C-0105 was awarded in June 1974 by the Applied Technology Laboratory, U.S. Army Research and Technology Laboratories (AVRADCOM) to produce a comprehensive base of helicopter test data. In particular, measurements were taken of parameters such as airfoil surface pressure, leading-edge stagnation point, local flow magnitude and direction, blade accelerations, bending moments, and the attendant responses in the control system and airframe. The output of 367 transducers was recorded continuously and simultaneously. Over 72,000 separate functions of time were digitized, recorded on digital tapes, and delivered to the Applied Technology Laboratory. The results of the above-mentioned contracted effort are documented in Report USAAMRDL-TR-76-39[1].

The Operational Load Survey Data Management System was developed under Contract DAAJ02-77-C-0053 awarded in September 1977 by the Applied Technology Laboratory (ATL). The software developed under this contract is primarily designed to process data taken during the AH-1G Helicopter Aerodynamics and Structural Loads Survey and other similar test programs. Documentation prepared under this contract consists of two volumes. Volume I provides user instructions and information on the analytical methods used in the software. Volume II, Systems Manual, details the various programming considerations.

Technical program direction was provided by Mr. D. Merkley of ATL. Principal Bell Helicopter Textron personnel associated with the current contract were Messrs. R. B. Philbrick, A. L. Eubanks, and W. R. Dodds.

---

[1]Gerald A. Shockey, Joe W. Williamson, and Charles R. Cox, AH-1G HELICOPTER AERODYNAMIC AND STRUCTURAL LOADS SURVEY, Bell Helicopter Co., USAAMRDL Technical Report 76-39, Eustis Directorate, U.S. Army Air Mobility Research and Development Laboratory, Fort Eustis, Va., February 1977, AD A036910.

TABLE OF CONTENTS

TABLE OF CONTENTS (Concluded)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Concluded)

LIST OF TABLES

1. INTRODUCTION

This volume documents the source code for the Operational Loads Survey - Data Management System (OLS/DMS) and should assist the programmer/analyst in modification, maintenance, and installation of the system. However, the reader must be familiar with Volume I of this report before reading Volume II, since many structural features, concepts, and terms for the system are introduced and defined in Volume I. Volume I describes the purpose, capabilities and analytical techniques of the system, and provides instructions for system commands.

The OLS/DMS source code is organized, written, and commented so as to minimize the difficulties of software maintenance and modification. This document was written both to further clarify the flow and structure of the system and to provide specific assistance for certain kinds of system modifications. Section 2 of this volume documents the File Creation Program and also explains the detailed structure of the Master File. Section 3 describes the Processing Program, including interpretation of command steps, processing, and graphics. Various utility routines that are used throughout the programs are described in Section 4. Appendixes A and B list the meanings for each of the common variables in the File Creation Program and the Processing Program. This information is essential to understand and maintain the code. Appendix C gives the Job Control Language and/or Time-Sharing Option commands to compile, edit, and execute the code.

Two specific kinds of system modifications are documented with particular detail. Modification of the File Creation Program to accept tape formats other than the BHT Ground Data Center (GDC) standard tape format is discussed in Section 2.3. Interface requirements for replacement of a processing module are discussed in Paragraph 3.4.4. When a new processing module is to be interfaced or when a replacement module requires new user instruction specifications, a modification of the user interface tree structure is required. This structure is discussed in detail in Paragraph 3.3.2.

Transportation of a large software system from the computer system it was designed for (in this case an IBM 370/168 or an IBM 360/65) to a different computer system can be a difficult and time-consuming process. In coding the OLS/DMS, a careful attempt has been made to minimize these difficulties and assure the transportability of the software. The code is written entirely in FORTRAN and use of IBM extensions to American National Standard (ANS) FORTRAN have been limited.

However, requirements for various system capabilities have made necessary the use of certain system, hardware, and installation dependent code. All such code is identified and explained in Section 5. In addition, system, hardware, and installation dependent code is identified in the program source statements with rows of stars, '*', above and below the nonstandard code.

2. FILE CREATION PROGRAM

2.1 MASTER FILE STRUCTURE

The Master File is a large direct access file containing records that are individually addressable by number and are 1024 bytes long. A numerically contiguous sequence of these records forms a partition and is referenced by an offset specified in the master file directory, which is always absolute record 1 (Figure 1).

The first four bytes of this directory are four characters, which when set to '$$$$' indicate that the entire file is initialized so that any record may be referenced directly. The next entry is an integer giving the total available size of the Master File in records. The third entry is eight bytes long and is a string called the superword, which is the key for the Master File Utility Program to list or delete partitions without individual passwords or to restore the whole Master File from tape. Following the superword in the directory record is a sequence of 63 possible 16-byte partition specifications. The first eight bytes of a partition specification form a string containing the partition name. The next four bytes form an integer giving the offset for the partition. The final four bytes give the length of the partition in records.

The initial record for a partition is specified by adding one to the partition offset (Figure 2). This record contains information about the partition as a whole. The user's name is contained as a string in bytes 1-16. In bytes 17-32 is the password which the user must have to modify or replace the partition. The third entry is the directory offset, which when added to the partition offset gives new offset for relative addressing of the partition directory. Entries four and five specify the partition directory size and partition data area size respectively in records.

The next sequential record contains the date and time, in string form as indicated by Figure 3, that the partition was last accessed.

The first directory record comes after the data records in the partition and always contains the initial record of the counter directory (frequently the only record in the counter directory). Figure 4 illustrates what this directory might contain if there were more than 127 counters in the partition. Each counter entry includes a counter followed by relative

ENTRY	BYTES	CONTENTS	PARTITION ENTRY
1	4	=4H$$$$ IF INITIALIZED	
2	4	TOTAL RECORDS	
3	8	SUPERWORD	
4	8	PARTITION NAME	1
5	4	PARTITION OFFSET	
6	4	PARTITION LENGTH	
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
190	8	PARTITION NAME	63
191	4	PARTITION OFFSET	
192	4	PARTITION LENGTH	

Figure 1. Absolute record #1 master directory record.

ENTRY	BYTES	CONTENTS
1	16	USER NAME
2	16	PASSWORD
3	4	DIRECTORY OFFSET
4	4	DIRECTORY SIZE
5	4	DATA AREA SIZE
6	8	DATE CREATED e.g. 12/15/77
7	8	DATE CHANGED e.g. 12/19/77

Figure 2. Relative record #1 (partition offset +1) partition initial record.

ENTRY	BYTES	CONTENTS
1	8	DATE LAST ACCESSED e.g. 12/20/77
2	8	TIME LAST ACCESSED e.g. 10.32.30

Figure 3. Relative record #2 (partition offset+2) partition access record.

ENTRY	BYTES	CONTENTS	COUNTER ENTRY
1	4	COUNTER	1
2	4	ITEM CODE DIRECTORY RELATIVE LOCATION	
3	4	COUNTER (=C)	2
4	4	ITEM CODE DIRECTORY RELATIVE LOCATION (=I)	
.	.	.	.
.	.	.	.
.	.	.	.
.	.	.	.
253	4	COUNTER	127
254	4	ITEM CODE DIRECTORY RELATIVE LOCATION	
255	4	0 => CONTINUATION	-
256	4	COUNTER DIRECTORY CONTINUE RELATIVE LOCATION = L	

Figure 4. Directory relative record #1 (partition offset + directory offset +1), counter directory initial record (more than 127 counters assumed).

ENTRY	BYTES	CONTENTS	COUNTER ENTRY
1	4	COUNTER	
2	4	ITEM CODE DIRECTORY RELATIVE LOCATION	128
3	4	COUNTER	
4	4	ITEM CODE DIRECTORY RELATIVE LOCATION	129
5	4	-1 => END OF COUNTERS	-
.	.	.	
.	.	.	
.	.	.	
.	.	.	
255	4	-1 => ENDED	
256	4	0	

Figure 5. Directory relative record #L (partition offset + directory offset +L), counter directory continuation record with termination.

location for the first (possibly only) record of the item code directory for that counter. A negative counter signals the end of the counter directory as shown in Figure 5.

The structure of each item code directory is identical to the counter directory as shown in Figure 6. In the example, the directory contains only three item codes (and thus uses only one record), but an item code directory could contain multiple records and hundreds of item codes as shown for the counter directory. Each item code entry includes a relative location, which points to an information record in the partition data area. Thus, only the partition offset is added to this pointer to obtain the information record location.

The information record for an item code/counter data stream contains information about that data stream and marks the beginning of data. The first data record follows the information record sequentially and all data records for that item code/ counter pair follow sequentially and contiguously. Figure 7 lists the contents of the information record. Some of the values in this record are necessary for processing the data stream and others are only present for information purposes.

The first four information record entries are available for future use if a program is written to condense partitions which contain unused areas where time histories have been deleted. These entries refer back to the corresponding item code directory record and position within the record. Entry five, the data stream length, must be divided by the number of points in a record (adding one for a non-zero remainder) to arrive at the number of records in the data stream.

The number of data values in a record is obtained from the number of bytes in a record, currently 1024, and the data word length, which depends on whether the data are calibrated or uncalibrated as stored (entry 6). A calibrated value is stored as a four-byte floating number, while an uncalibrated value is stored as a two-byte integer, giving 512 uncalibrated data values per record or 256 calibrated data values per record. Uncalibrated data can be calibrated using entries 27 and 28.

The sample rate (data points/second) of the data stored is obtained by dividing entry 23, the initial sample rate on digital tape, by entry 24, the sample rate reduction factor. The data type, entry 25, indicates whether the data are time history or min/max data.

ENTRY	BYTES	CONTENTS	ITEM CODE ENTRY
1	4	ITEM CODE	
2	4	DATA INFO RECORD RELATIVE LOCATION	1
3	4	ITEM CODE	
4	4	DATA INFO RECORD RELATIVE LOCATION	2
5	4	ITEM CODE (P)	
6	4	DATA INFO RECORD RELATIVE LOCATION (=K).	3
7	4	-1 => END OF ITEM CODES	-
.	.	.	
.	.	.	
.	.	.	
255	4	-1 => ENDED	
256	4	0	

Figure 6. Directory relative record #I (partition offset + directory offset + I), item code directory for counter 'C' (counter entry #2, Figure 4).

ENTRY	BYTES	CONTENTS
1	4	ITEM CODE
2	4	COUNTER
3	4	DIRECTORY RECORD LOCATION
4	4	SEQUENCE POSITION IN RECORD
5	4	DATA STREAM LENGTH (DATA POINTS)
6	4	1=CALIBRATED, 0=UNCALIBRATED
7	4	START TIME
8	4	STOP TIME
9	4	ALIGNMENT OFFSET (SEC)
10	4	ADDITIONAL OFFSET (SEC)
11	2	TRACK-BAND WORD
12	2	ANALOG FILTER RATE CODE
13	30	ITEM CODE DISCRIPTION
14	6	ITEM CODE UNITS
15	4	REFERENCE VALUE (REF)
16	4	DELTA CAL VALUE (ΔCAL)
17	4	CAL COMMAND (VCC)
18	4	CAL SHIFT VALUE (VCS)
19	4	REFERENCE VOLTAGE (VREF)
20	4	INTERCEPT VALUE (B)
21	4	DIGITAL FILTER CUTOFF (HZ)
22	4	DATA RATE REDUCTION (SKIP) FACTOR
23	4	INITIAL DATA RATE
24	48	ASSIGNMENT RECORD FIELDS 6 THRU 13
25	4	DATA TYPE: 1=MIN/MAX, 2=HISTORY
26	4	ANALOG PLAYBACK SPEEDUP FACTOR
27	4	CAL SLOPE (XM)
28	4	CAL INTERCEPT (XB)

Figure 7. Partition relative record #K (partition offset +K) information record for item 'P', counter 'C'.

Entries 9 and 10 indicate the time offsets in seconds applied to the data stream in transfer from digital tape to the Master File. Entry 9 is for information purposes and indicates the amount of data discarded in an effort to line up the starting data point in time with all other starting data points from the same counter. Entry 10 shows the amount of additional data discarded before a subsequent data point was saved on disc. A negative value for entry 10 indicates no time alignment was done even though data from other item codes for the same counter may be aligned. The additional offset is then the absolute value of entry 10.

The other entries are present largely for information and display purposes and are all explained in Reference 2, except for the digital filter cutoff, entry 21. This entry gives the cutoff of the low-pass digital convolution filter (in Hz) applied to the data during transfer from tape to disc. A value less than or equal to zero indicates no filter was applied.

Now that the Master File and partition record structure have been examined in detail, the overall structure of the Master File can be considered by looking at Figure 8. The first record of the Master File is the Master File directory record which, for an existing partition, supplies an offset pointing to the initial record of that partition. This initial record contains a second offset pointing to the partition directory. The partition directory first record is the initial record of the counter directory which, for a given counter, points to the initial item code directory record for that counter. The item code directory points, for a given item code, to the information record in the data area for that item code/ counter pair. The data stream follows the information record contiguously.

Some advantages of the Master File structure are now evident. First, a partition as a whole is easily portable since every record in the partition is located with relative addressing. Second, a partition directory is easily portable separate from the partition since records within the directory are located using a second order of relative addressing. Third, there is no theoretical limit to the number of item codes or counters

---

[2]L. J. Tieman, 'GROUND DATA CENTER STANDARD DIGITAL TAPE FORMAT,' Bell Helicopter Textron Report 699-099-020, Fort Worth, Texas, 21 April 1976.

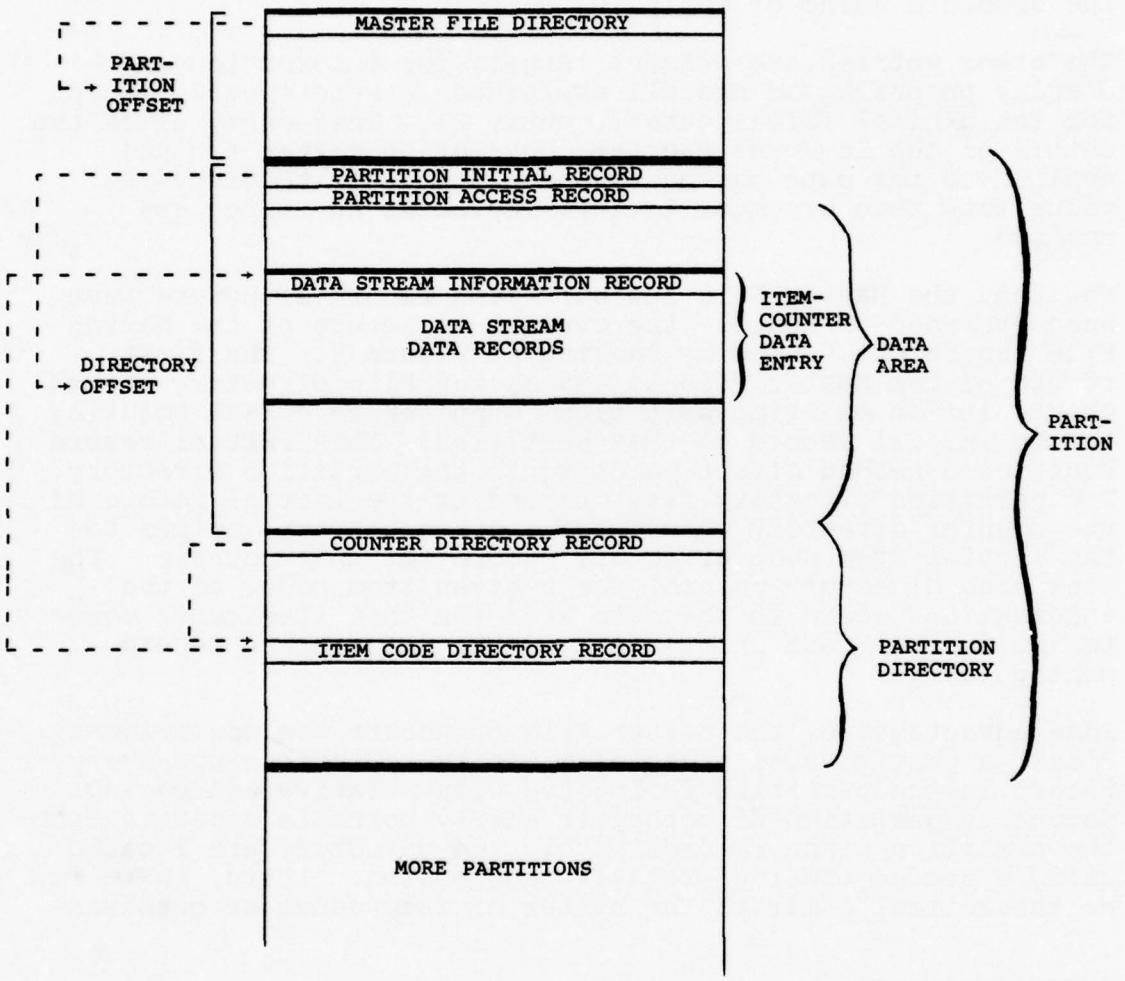


Figure 8. Master File structure.

stored or to the amount of data from an item code/counter data stream which can be saved. Practically, physical disc space limitations will limit these quantities.

2.2 FILE CREATION PROGRAM FLOW

The flow sequence of the File Creation Program is described here with close reference to the flowchart on Figures 9, 10 and 11 using the numeric label just outside each block.

Block 1 MAIN calls INLIST to read input commands according to the free field format described in Volume I. READF is used to interpret numeric input and group strings. MATCHR is called to recognize keywords. PACK is used to transfer four characters to one four-byte word.
Block 2 MAIN lists the number of errors detected by INLIST and then calls LISTCM to sort and list the data requests. Any duplicate item codes or counters are noted and the duplicates rejected.
Block 3 MAIN checks the number of errors detected by INLIST and goes to an error termination point if one or more occurred or if no input was requested; otherwise, the program goes to Block 5.
Block 4 is an error termination in MAIN. The Master File has not been disturbed at this point.
Block 5 If there are no input errors, MAIN calls SETUP1 to read the first record of the Master File and check that the Master File is initialized. If the Master File is not initialized properly ('$$$$'), then the routine returns an error and MAIN goes to Block 4. With proper initialization, SETUP1 double checks the initialization by attempting to read the numerically highest record in the file. Failure on this read attempt abnormally terminates the job.
Block 6 Assuming that the previously mentioned read attempt succeeds, SETUP1 initializes the direct access scratch disc file using the sequential alias for that file.
Block 7 SETUP1 also provides WMS, RMS and FMS (routines which do intermediate checking and apply the relative offsets before performing direct access WRITE, READ and FIND calls respectively) with preliminary offset and check values for the Master File and the scratch random access file. Control is then returned to MAIN.
Block 8 - MAIN then calls MAKRUM which sorts the partition names by ascending location in the file and attempts to match the name of the specified ADD/NEW/REPLACE partition.
Block 9 - (Still in MAKRUM) If a match for the requested partition name is found, the program goes to block 10; if not, the program goes to block 13.

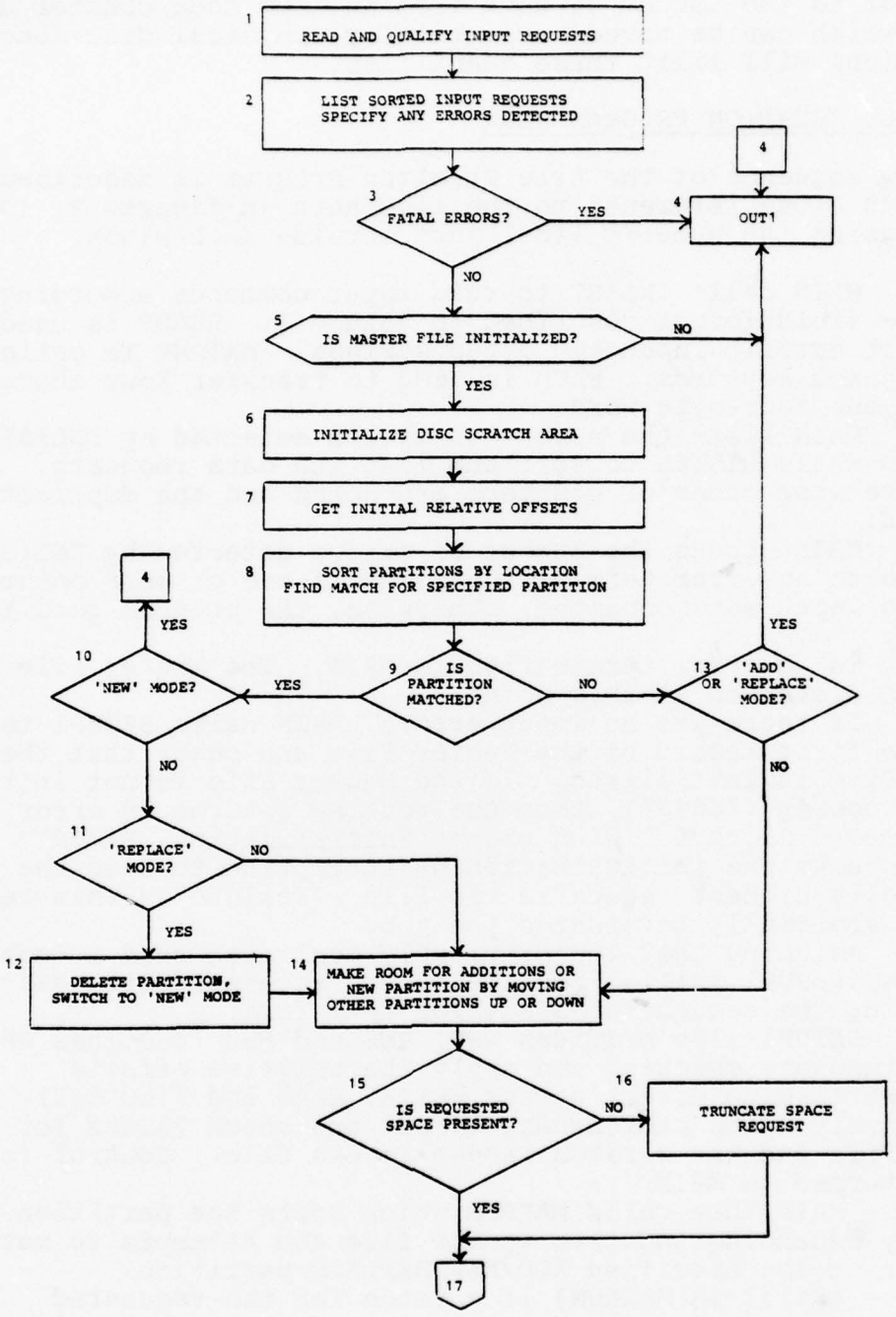


Figure 9. File Creation Program flow diagram (first part).

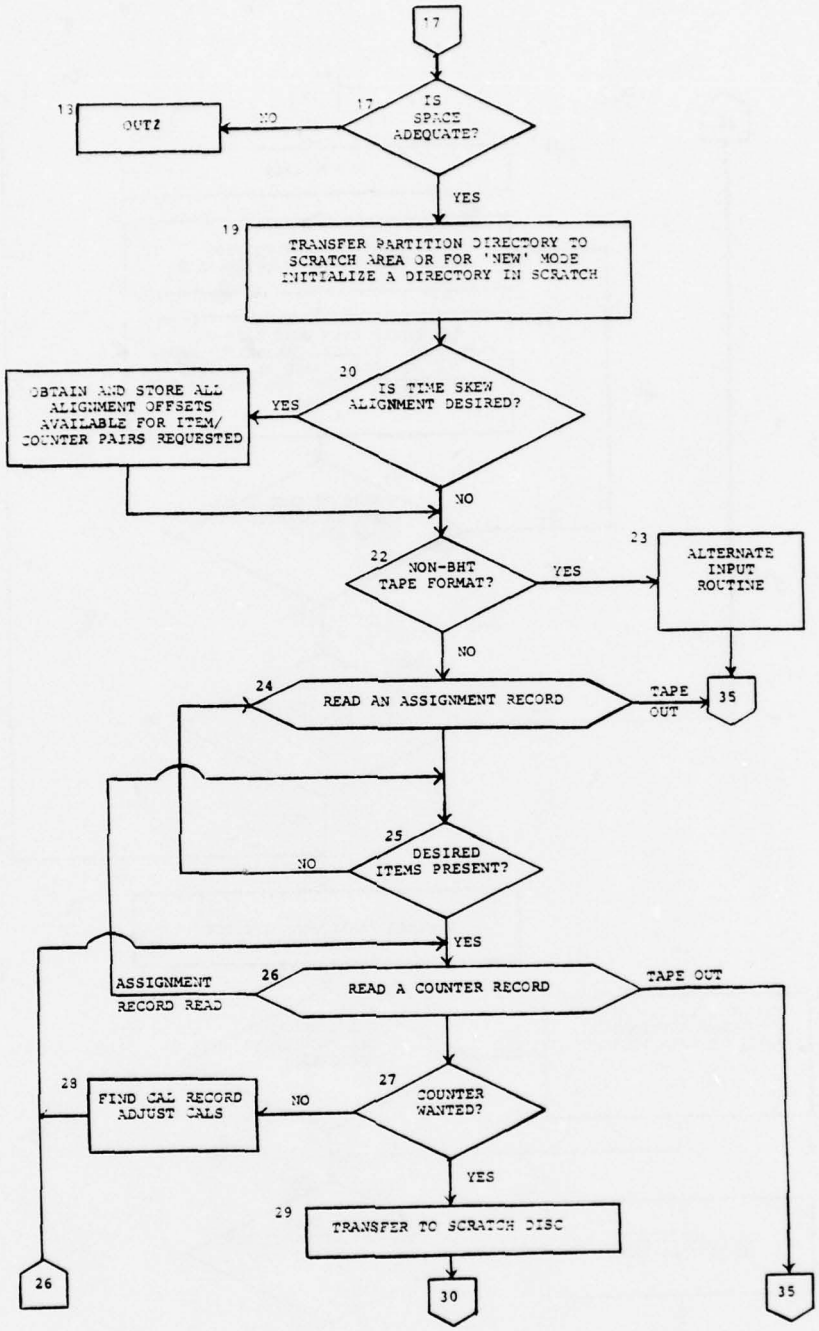


Figure 10. File Creation Program flow diagram (second part).

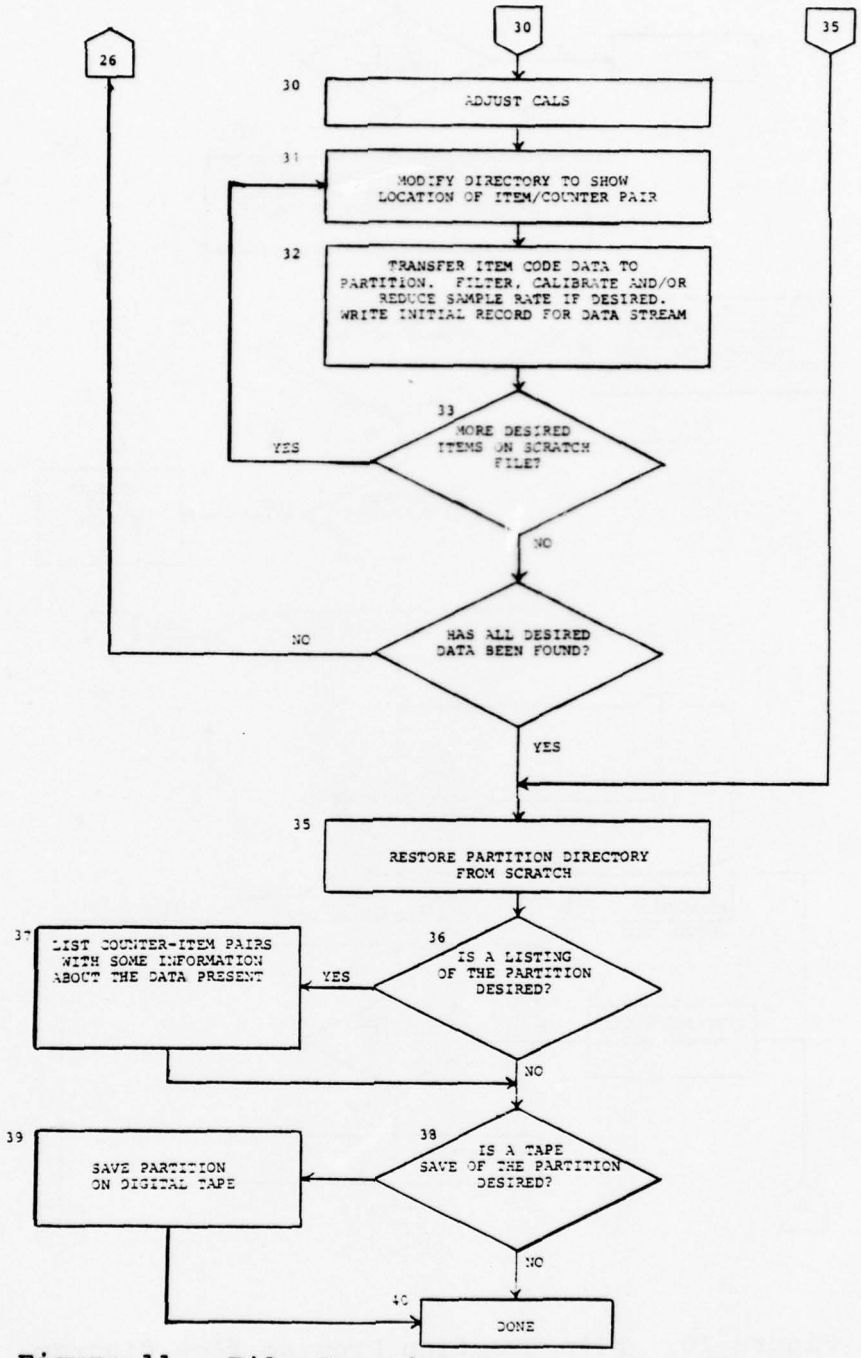


Figure 11. File Creation Program flow diagram (third part).

Block 10 - (MAKRUM) A check is made on whether the requested partition name was supposed to be 'NEW'. If so, an error has occurred since a partition by that name already exists and the program goes to block 4. If not, the program goes to block 11.

Block 11 - (MAKRUM) A check is made on whether the requested partition name was supposed to be replaced. If so, the program goes to block 12; if not, it goes to block 14.

Block 12 - (MAKRUM) The partition matched is removed from the Master File directory and then the mode is changed from 'REPLACE' to 'NEW'. Thus, the partition name will be retained but different data will be added to the partition. Then the program goes to block 14.

Block 13 - (MAKRUM) No match for the requested data set name has been found so the program checks whether an 'ADD' or 'REPLACE' has been specified. If so, an error has occurred so the program goes to block 4. If not, the program goes to block 14.

Block 14 - All gaps between the last record of a partition and the first record of the next sequential partition are eliminated. Any gap between the first sequential partition and the Master File directory record is eliminated. The record space following the partition to be modified is maximized. This process of moving partitions up and down in the Master File uses a scratch disc file so that a number of records are read from the Master File to scratch and then from scratch to a new location in the Master File. When this process is complete, the total number of Master File records available for the partition to be modified is computed. Then, the program goes to block 15. As mentioned in Volume I, Section 3.2.1, the entire Master File could be destroyed if too short a time limit were specified for a run of the File Creation Program. In particular, destruction of the Master File would occur if block 14 of the File Creation Program were executing when the time limit was encountered.

Block 15 - (MAKRUM) A comparison is made of the number of Master File records available and the total number of records requested for the partition. If fewer Master File records are available than requested, the program goes to block 16; otherwise, the program goes to block 17.

Block 16 - (MAKRUM) The space request is truncated to the amount of space available in the Master File. Then, the program goes to block 17.

Block 17 - (MAKRUM) The space request (original or truncated) is checked to assure that it provides a basic amount of space for a partition or a partition increment. If the space is inadequate, the program goes to block 18; otherwise, the program goes to block 19.

Block 18 - Is an error return from MAKRUM to MAIN and a termination message indicating the problem is generated. At this point, the directory has been reset excluding the partition of interest.
Block 19 - MAKRUM returns to MAIN, which calls SETUP2 to prepare for the partition creation/addition process. If in 'ADD' mode, the existing directory is transferred to the scratch random access file where it will be added to and modified in the data addition process. If in 'NEW' mode, the directory is initialized in the scratch random access file. Control returns to MAIN and the program goes to block 20.
Block 20 - MAIN checks whether ALIGN has been specified and if so, the program goes to block 21; otherwise, the program goes to block 27.
Block 21 - MAIN calls a routine to provide alignment correction offsets for each item code desired for all counters. These offsets will correct for time skew misalignment in the data. The routine takes the lists of item-codes and counters and provides a number of data points to be discarded (at the original data rate) at the beginning of each item code/counter pair data stream. An invalid offset is indicated with a -1. Offsets are stored on disc by routine EXCORE. From here the program proceeds to block 22.
Block 22 - MAIN checks to see if a tape format other than the standard BHT format has been specified. If so, control is transferred to block 23. If not, control is transferred to block 24.
Block 23 - MAIN calls STRNGF to input data in a format other than the BHT standard tape format. See Section 2.4 for more information on the required format for STRNGF. When STRNGF returns, control is transferred to block 35.
Block 24 - MAIN calls READD to read an assignment record from the tape. READD will continue reading blocks until an assignment record is found or all tapes have been read. If no assignment record is located, control is transferred to block 35. Otherwise, an assignment record is returned to MAIN and control is transferred to block 25.
Block 25 - MAIN calls FITEM to scan the assignment record to determine whether any requested item codes are present. If not, control is transferred to MAIN with zero items found (NMATCH=0) and control is transferred to block 24. If some desired items have been found (NMATCH>0), selected assignment record information is saved for each item and control is returned to MAIN and then to block 26.
Block 26 - MAIN calls READD to input the first block of data for a counter. Three possibilities exist on return from READD. If READD has indeed found an initial data block for a counter, the program goes to block 27. If READD ran out of tapes, the program goes to block 35. If a new assignment block has been read, the program goes back to block 25.

Block 27 - MAIN calls FCNTR to check whether this counter was requested. If not, the program returns to MAIN (MCNTR=-1) and then to block 28. If the counter is wanted, the program saves some information particular to this counter, returns to MAIN, and goes on to block 29.
Block 28 - This counter is not wanted so MAIN calls READD to page forward to the CAL record and calls CALUPD to update the calibration values. Then the program returns to block 26. An outside possibility exists, if there is an error on the tape, that READD could return with an assignment record or an out of tape indication. In these cases, the program would go to blocks 25 or 35, respectively. These lines were left off the flowchart for simplification.
Block 29 - MAIN calls TRANSC to transfer data for the counter from tape to scratch disc. TRANSC attempts to transfer only the data blocks required for storage. Other blocks are left off (any offset for the counter as a whole is adjusted). TRANSC pages forward to the CAL record and then returns to MAIN and the program goes on to block 30.
Block 30 - MAIN calls CALUPD to update the calibration factors based on the CAL record present in scratch common. Then the program goes on to block 31.
Block 31 - MAIN enters a DO loop (through block 33) to transfer the data from scratch to the partition. MAIN calls LOCFIX to modify the directory to reflect the counter/item code data stream location for the next item code data stream to be transferred. LOCFIX returns to MAIN and the program goes on to block 32.
Block 32 - This is a large block covering a great deal of detailed processing and two options. If no digital filtering is desired, MAIN will call SAVD to strip the data for the item code of interest out of the scratch file and write these data to the Master File. Data can be calibrated if this mode is selected and the data rate can be reduced by using only every n'th data point as specified by the sample rate reduction factor. If more data is requested than available, then the data request is truncated to the amount available.

If digital filtering is desired, MAIN will call SAVF instead of SAVD. SAVF will perform the same functions as SAVD except that the data are digitally filtered and calibration must be performed.

Either routine returns to MAIN which then calls INIDAT to write the information record on the Master File at the beginning of the data stream. INIDAT returns to MAIN and the program goes on to block 33.

Block 33 - This block represents the test at the end of the MAIN 'DO' loop, which checks to see if all the items requested and present on scratch have been found. If the 'DO' is complete, the program goes on to block 34.
Block 34 - MAIN compares the number of item code/counter pairs found so far with the number requested. If the number of pairs found so far is complete, the program stops searching tape and goes on to block 35. If all requested data have not been found, the program returns to block 26.
Block 35 - MAIN calls RESTRD to copy the partition directory from the scratch random access file to the top of the partition. RESTRD returns to MAIN which then annotates the Master File directory record to reflect the size and location of the partition. Then, the program goes on to block 36.
Block 36 - MAIN checks to see whether a listing of the modified partition was requested. If so, the program goes to block 37; if not, the program goes on to block 38.
Block 37 - MAIN calls MAP to list the item code/counter pair data streams present in the partition along with some information on each data stream. Then control returns to MAIN and the program goes on to block 38.
Block 38 - MAIN checks whether a digital tape save of the revised Master File is wanted. If not, the program goes on to a normal exit at block 40. If so, the program goes to block 39.
Block 39 - MAIN calls SAVALL to save the partition on digital tape. SAVALL then returns to MAIN and the program goes to block 40.
Block 40 - Done.

2.3 NON-BHT DATA FORMATS

The File Creation Program can be modified to accommodate data tape formats other than the standard BHT-GDC format through generation of an appropriate replacement for the program stub, STRNGF. The rest of the File Creation Program will continue to provide the following functions, read user instructions, manage Master File Space, manage partition directory, write data to Master File. The subroutine STRNGF must handle all the details of reading the data from digital tape, consult with the common block /LIST/ containing the user instructions (see Appendix A), provide appropriate information for each item code/counter pair, and provide the data for transfer to the Master File in record size blocks.

Table 1 lists a prototype version of STRNGF showing the sequence which must be followed to store data. The routine ADDAT is appended to actually write data to the Master File. However, code must be provided to satisfy the required function of STRNGF as listed in the program comments. Data must

TABLE 1. PROTOTYPE EXAMPLE FOR SUBROUTINE STRNGF

```
      SUBROUTINE STRNGF
C
C   PROTOTYPE 'STRNGF' ROUTINE FOR READING DATA FROM
C   NON-BHT-GDC STANDARD TAPE FORMATS FOR THE OLS/DMS
C   FILE CREATION PROGRAM.
C
      DIMENSION INREC(256)
      COMMON/LOCOM/ITEMN,ICNTRN,IDROFF,
     $  IDASIZ,IDRSIZ,ITEMRC,ITEMSQ
      LOGICAL LCAL
      COMMON/INFO/IRSIZ,MLOC,LOCO,IPCLES,HIGH,LCAL,IRAT,
     $ ISKIP,NPS,NPP,NOFF,ISEQ,LSTRT,IADD,INSIZ,INSIZD,
     $ IRDATS,IRSAVS,ICNTR,XALIGN
      COMMON/KARD/ITEMTP(26),ITEMW(26),CALSH(26),
     $ CALCM(26),CXM(26),CXB(26),NMATCH,DCAL(26)
      LOGICAL LALIN,MAPIT,SAVIT,STRANG
      COMMON/LIST/NCTR(400),NOFFST(400),NPWANT(400),
     $ ITEM(400),FILT(400),ICAL(400),ISKP(400),
     $ NITEMS,NCNTRS,ISPAC,ITAPES,IADNU,LALIN,
     $ NAME(2),NPWD(4),NUSER(4),MAPIT,SAVIT,STRANG
      COMMON/FILES/NRPS,NRSC,NSSC,NITT,NDIR,NREA,NWRI,
     $ NSAV,IALI
C
C
      IEND = 0
      LW = IRSIZ/2
C
C   SET UP INPUT FROM NON-STANDARD TAPE.
C   INSERT CODE AS APPROPRIATE.
C
C   LOOP OVER SUBSETS OF THE DATA ON TAPE.
      DO 500 I = 1,10000
C
C       DETECT THE PRESENCE OF A SUBSET OF THE REQUESTED
C       DATA CORRESPONDING TO ONE COUNTER AND ONE OR MORE
C       ITEM CODES.  ASSIGN A NUMBER BETWEEN ONE AND THE
C       DIMENSION OF THE ARRAY 'ITEMW' TO EACH ITEM CODE
C       IN THE SUBSET (ADJUST THE DIMENSIONS OF THE ARRAYS
C       IN THE COMMON BLOCK /KARD/ AS NECESSARY).  SET
C       SET ITEMW(N) FOR EACH ITEM CODE NUMBER, 'N', TO
C       THE CORRESPONDING ARRAY POSITION OF THE ITEM CODE
C       IN THE ARRAY 'ITEM'.  IF DATA IS TO BE STORED IN
C       INTEGER FORM, SET THE CORRESPONDING 'CXM' AND 'CXB'
C       ARRAY VALUES FOR CALIBRATION ON RETRIEVAL FROM THE
C       MASTER FILE.  SET 'NMATCH' TO THE NUMBER OF ITEM
C       CODES IN THE SUBSET.  INSERT CODE AS NECESSARY TO
C       PERFORM THESE FUNCTIONS
C
C       LOOP OVER ITEMS IN THE SUBSET
          DO 400 J = 1, NMATCH
C
C             SET THE VARIABLES 'ICNTRN' AND 'ITEMRC' TO THE
C             COUNTER AND ITEM CODE RESPECTIVELY.  SET THE
C             ARRAY 'INREC' WITH SOME OF THE REQUIRED VALUES
C             FOR THE CORRESPONDING INFORMATION RECORD.
C               INREC(1) = ITEM CODE = ITEMRC
C               INREC(6) = 1 IF CALIBRATED DATA IS TO BE STORED,
C                          0 IF INTEGER DATA
C               INREC(12-20) = ITEM CODE DISCRIPTION/UNITS WITH
C                              UNITS IN LAST SIX BYTES
```

TABLE 1. PROTOTYPE EXAMPLE FOR SUBROUTINE STRNGF (Continued)

```
C               INREC(27) = DIGITAL FILTER CUTOFF, -1.0 IF NO
C                           FILTER APPLIED
C               INREC(28) = SAMPLE RATE REDUCTION FACTOR.
C                         = ISKP(ITEMW(J))
C               INREC(29) = SAMPLE RATE OF DATA ON TAPE BEFORE
C                           SAMPLE RATE REDUCTION FACTOR IS
C                           APPLIED
C               INREC(42) = 2  (TIME HISTORY DATA, NOT MIN/MAX)
C
                CALL WMS(2,INREC,LW,J,IERR)
                IF(IERR .NE. 0)GO TO 580
C
C             SET MCNTR TO THE ARRAY POSITION IN 'NCTR' ARRAY
C             FOR THE CURRENT COUNTER.  I.E.
C             SET     ICNTRN = NCTR(MCNTR)
C             INSERT CODE AS NECESSARY
C
                CALL LOCFIX(NERR,INFO,IERR)
                IF(NERR .NE. 0)GO TO 550
C
C             LOOP OVER RECORDS OF OUTPUT FOR ITEM CODE
                DO 300 K = 1,10000
C
C                 READ THE DATA FOR THE NEXT RECORD.  IF OUT OF
C                 DATA, BRANCH TO 350.  GET DATA (256 CALIBRATED
C                 OR 512 INTEGER VALUES) INTO THE ARRAY 'INREC'.
C                 SET NUM TO NUMBER OF POINTS IN THE RECORD.
                    CALL ADDAT(INREC,NUM,ICHK)
                    IF(ICHK .EQ. 0)GO TO 300
                        IEND = 1
                        GO TO 350
 300            CONTINUE
C
C             ADD THE INFORMATION RECORD FOR THE TIME HISTORY
C             JUST TRANSFERRED TO THE PARTITION.
 350            CALL INIDAT(J,MCNTR,NERR,INFO,IERR)
                IF(NERR .NE. 0)GO TO 560
                IF(IEND .NE. 0)GO TO 570
 400        CONTINUE
 500    CONTINUE
        GO TO 1000
C
C   DIRECT ACCESS WRITE ERROR STORING INFOMATION FILE
C   ON SCRATCH DIRECT ACCESS DISC.
 550    WRITE(NWRI,9000)IERR
        GO TO 1000
C
C   ERROR ADDING INFORMATION RECORD TO THE PARTITON.
 560    WRITE(NWRI,9001)NERR,INFO,IERR
        GO TO 1000
C
C   OUT OF SPACE ON THE PARTITION
 570    WRITE(NWRI,9002)IERR
        GO TO 1000
C
C   ERROR ANNOTATING DIRECTORY FOR START OF DATA STREAM
 580    WRITE(NWRI,9003)NERR,INFO,IERR
C
C
```

TABLE 1. PROTOTYPE EXAMPLE FOR SUBROUTINE STRNGF (Concluded)

```
 1000 RETURN
C
C
 9000 FORMAT(3X,39H***ERROR STORING INFO RECORD ON SCRATCH,I10//)
 9001 FORMAT(3X,42H***ERROR STORING INFO RECORD ON PARTITION ,
     $    3I10//)
 9002 FORMAT(3X,36H***RAN OUT OF SPACE ON THE PARTITION //)
 9003 FORMAT(3X,26H***ERROR SETTING DIRECTORY //)
      END

      SUBROUTINE ADDAT(IDAT,NUM,ICHK)
C
C  ROUTINE FOR USE BY ROUTINE 'STRNGF' TO WRITE DATA
C  TO THE MASTER FILE.
C       IDAT = DATA ARRAY
C       NUM = NUMBER OF VALUES IN DATA ARRAY
C             (SHOULD EQUAL 'LIM' UNLESS LAST RECORD)
C       ICHK = PROBLEM RETURN
C                0 - NO PROBLEM
C                1 - DIRECT ACCESS WRITE ERROR
C                2 - OUT OF SPACE FOR MORE WRITES.
C
      DIMENSION IDAT(1)
      LOGICAL LCAL
      LOGICAL LCAL
      COMMON/INFO/IRSIZ,MLOC,LOCO,IPOLES,HIGH,LCAL,IRAT,
     $ ISKIP,NPS,NPP,NOFF,ISEQ,LSTRT,IADD,INSIZ,INSIZD,
     $ IRDATS,IRSAVS,ICNTR,XALIGN
C
C
      ICHK = 0
      IF(NUM .LE. 0)GO TO 1000
C
C
      LW = IRSIZ/2
      LIM = IRSIZ
      IF(LCAL)LIM = LIM/2
      CALL WMS(1,IDAT,LW,MLOC,IERR)
      IF(IERR .GT. 0)GO TO 500
      NPS = (MLOC-IDASIZ-1)*LIM*ISKIP + NUM*ISKIP
      MLOC = MLOC + 1
      IF(MLOC .GT. MLEN(1)+4)GO TO 510
      GO TO 1000
C
C  DIRECT ACCESS WRITE ERROR
 500  ICHK = 1
      GO TO 1000
C
C  OUT OF DATA SPACE IN PARTITION
 510  ICHK = 2
C
C
 1000 RETURN
      END
```

be selected from digital tape using the user instructions in /LIST/. Appropriate data must be extracted from the tape or provided in some other manner for the information record for each item code/counter pair. In some instances, STRNGF may need to translate the identifiers on the input digital tape to four-character item codes and integer counters with values between 1 and 32767.

STRNGF will most likely provide calibrated, REAL data for transfer to the Master File. However, the option is available when the program is executed on a system with INTEGER*2 capability to store the data in integer format. In this case, STRNGF must call ADDAT with twice as many records containing INTEGER*2 values as would be provided if the values were REAL. In addition, the appropriate calibration factors must be provided, and the information record value INREC(6) must be set to zero to indicate that integer values are present.

When the sample rate is to be reduced, STRNGF must perform this function before supplying the data to ADDAT. The sample rate reduction factor must be inserted in the array location INREC (28). Notice that the array location INREC(29) must contain the original sample rate on tape before the sample rate reduction factor is applied.

The subroutines LOCFIX, ADDAT and INIDAT are called by STRNGF. The routines will appropriately manage storage of the data on the Master File and annotation of the directory. The routines also monitor error conditions so that the error returns must be appropriately handled by STRNGF as shown in the prototype.

3. PROCESSING PROGRAM

3.1 STRUCTURE AND FLOW

The OLS/DMS Processing Program was designed to be broken into overlays corresponding to various functions of the program. Figure 12 shows a diagram of program flow from block to block with the main block excluded. The Main program is not shown in this figure and serves only to transfer control from block to block and store certain utility routines used by more than one block.

The Startup or Program Initialization block extracts setting commands from the user which should be valid throughout the program run, and initializes and/or validates certain files including the Master File. The User Command Interface reads and checks the user commands and produces an instruction matrix (common block /DIRECT/) which can be interpreted by the other overlay blocks. The Processing block performs all the data retrieval, data processing, and data display functions of the program according to the instruction matrix. The Command Sequence block initializes all command sequencing functions and performs the actual editing of command sequence blocks. The Menu block generates non-data displays to assist the user in generating processing commands. The Terminate function is accomplished in program MAIN.

3.2 PROGRAM INITIALIZATION

Subroutine STRTUP is the control routine for this block. The required and optional user inputs for this phase are described in Section 5.6 of Volume I. The entries are read and interpreted using the READF and MATCHR utilities, as well as the READ1 and READOP routines and code within STRTUP.

In addition to extracting user control options for the program run, the Program Initialization block performs several other setup functions. The first function performed in STRTUP is to call the CPU timer initialization routine, SETIME. SETIME is installation dependent and may be replaced by another routine or entry name which starts the CPU timer (see Section 5.4).

After the user options have been specified, STRTUP calls ALLSCR to initialize each of the direct access disc scratch areas, including SCF1, SCF2, and the temporary scratch area. There are two possible configurations for these scratch areas. For the first possible configuration, each scratch area is associated with a different I/O file number and each I/O file

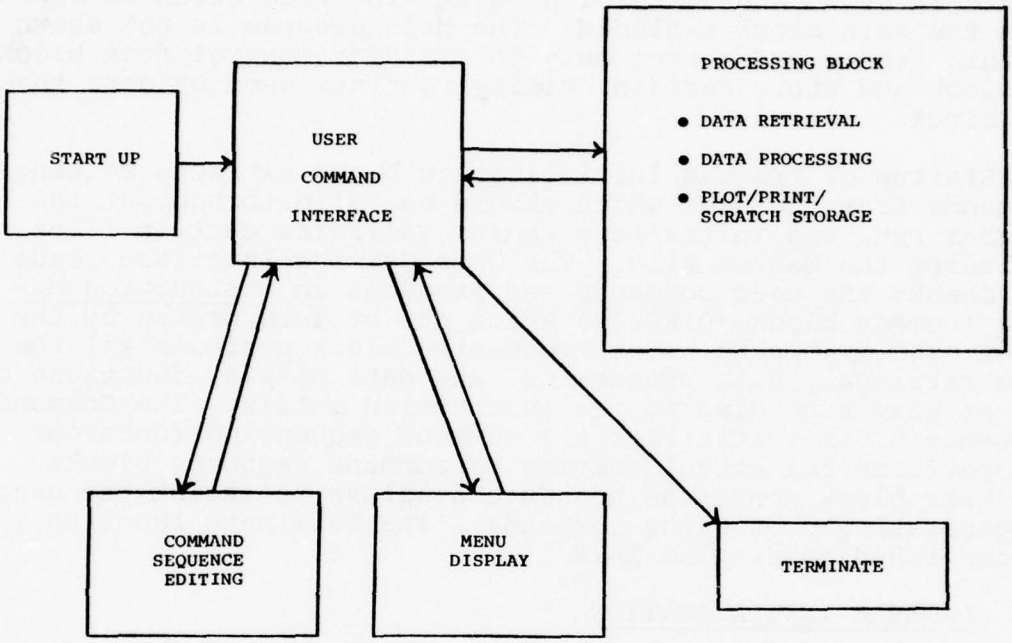


Figure 12. General flow of processing program.

number is presumably associated with a different area on the disc. When this configuration is used, the optimum results are obtained when each I/O file corresponds to a different physical disc drive so that head movement is minimized during transfer of data between files.

The second possible configuration for the scratch files places all these files on one contiguous area of disc addressed by one I/O file number. The different scratch files are addressed as different pseudo-devices using the RMS, WMS and FMS calls. The pseudo-device numbers are listed in Section 4.1. For either configuration, the files are initialized by writing dummy records on every record position of the sequential alias(es) for the direct access file(s).

After ALLSCR returns, STRTUP calls INFOST. This routine reads the initial group of the Info file and stores the keywords, item codes, and associated numeric values in common block /SINGIF/. Then STRTUP calls EDINIT to read the initial record of the direct access Command Sequence (Edit) file, to check the size of the file, and to set certain variables based on this size.

Following the EDINIT call, STRTUP extracts the name of the Master File partition which is to be accessed during the program run. Then DASTRT is called to find the partition and to set up the retrieval routines to address the partition data. If the partition name is successfully found and the Master File is properly initialized, STRTUP transfers the current date into the system output label, DEFCOM (in common block /DEFLT/), and exits. If the partition name is not found, the user is requested to enter a corrected partition name.

3.3 USER INTERFACE

The User Interface generates an instruction matrix for each command step. This matrix is generated by extracting from the user a sequence of entries which specify option selections for the matrix values. Relatively few of the instruction matrix values are specified for each command step, since a small subset of the total number of command specifications is required for execution of each different command. For example, a MENU command will not require specification of the static pressure or outside air temperature instructions.

A pseudo tree structure directs the program in specifying a sequence which includes all entries required for execution of the command step that is being generated. Each element of a sequence depends upon the options selected for the previous

elements of the same sequence. For convenience in specifying defaults, generating HELP messages, and explaining the entry sequences, each sequence is broken into one or more substeps as explained in Volume I. This tree structure, together with allowed options and HELP message strings, is stored as data in common blocks. The user interface code interprets the stored tree structure and maintains the syntax for user input. Thus, a change in user commands which does not conflict with the current command syntax should require only a change in the block data statements and array sizes and no change in the executable user interface code. Paragraph 3.3.1 discusses this code, while Paragraph 3.3.2 covers the requirements for the block data tree structure.

3.3.1 User Interface Routines

USER is the main routine for the user interface block. Figures 13 and 14 depict the flow for subroutine USER, which encompasses most of the general logic for the user interface block. The other routines for this block are briefly described below.

INISTP is the first routine called by USER to set the default values for the step, to initialize certain pointers, to calculate the CPU time for the previous step execution, and to print the 'NEW STEP' message that prompts the user for the next command step.

LININP is used to obtain a scanned, valid line of user input. LININP will obtain the line from system input or the command sequence file (using EDINP) according to the edit mode indicated by the variable LED (in the common block /LEDIT/). The line is scanned by READF to check for line errors, to evaluate numeric entrys, and to delimit string entries.

MATCHR is used to match individual strings of characters with an array of four-character keywords stored in common block/ WLIST/.

INTERP is called to interpret each individual user entry. The various categories of entries are numbers, nulls (defaults), keyword strings, non-keyword strings (e.g., an item code), specified defaults (i.e., defaults specified by a slash which terminates a substep), and comment entries. INTERP assures that the entry conforms to the allowed values for the current tree position and codes the entry in the instruction matrix.

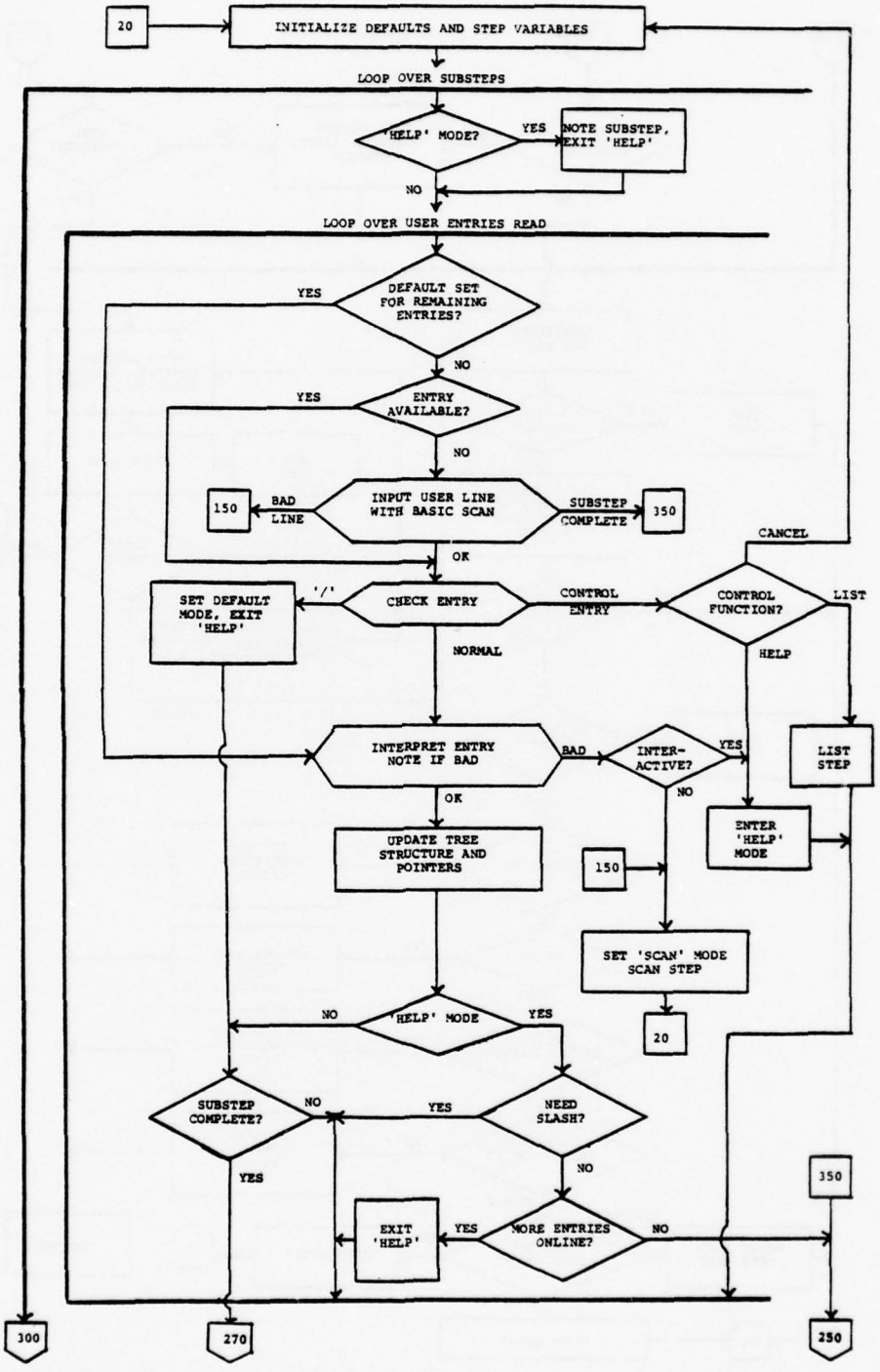


Figure 13. User interface flow diagram (first part).

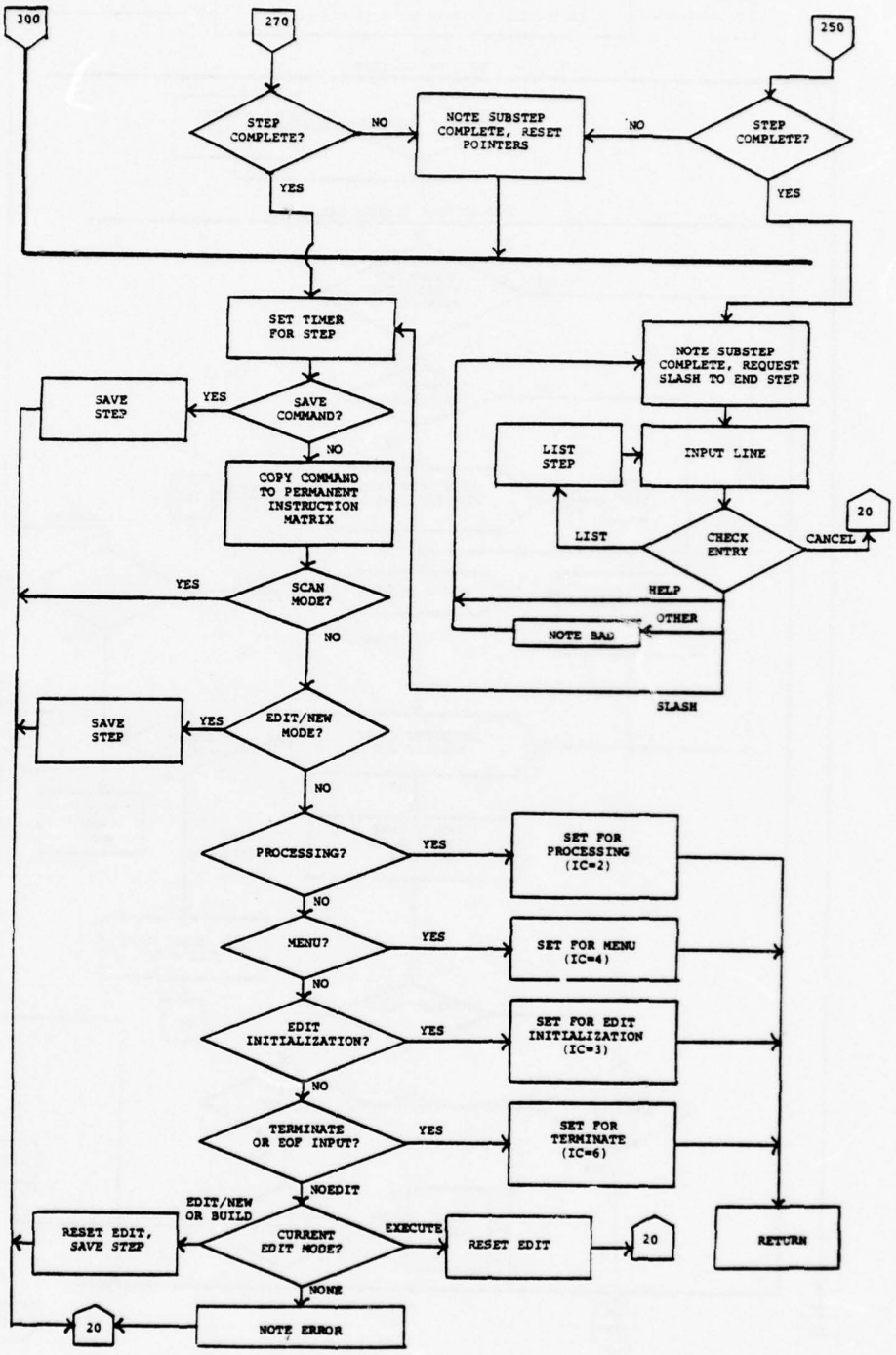


Figure 14. User interface flow diagram (second part).

The HELP mode prompting message generation routine, HELPR, is called by LININP when the HELP indicator, IHELP, is set to one. HELPR prints a prompting message for the current entry and looks ahead in the tree structure to print prompting messages for subsequent entries.

TREEUP updates the tree structure position and the substep number, ISBSTP, as necessary.

EDSAVE (see Paragraph 3.5.1) is used to save a command step on the command sequencing file.

LISTAD maintains the listing of the current command step including default entries. NTOSTR (see Section 4.2) is used to convert numeric values to string form.

3.3.2 User Input Encoding

The basic tree structure for the user interface is contained in the two-dimensional array NPOINT. The second array index for NPOINT corresponds to the tree position index. Thus, each tree position is mapped to a unique, positive integer (e.g., 6), which specifies three words in NPOINT (e.g., NPOINT (1,6), NPOINT (2,6), NPOINT (3,6)). The first index is dimensioned to three and these three allowed values correspond to three kinds of information stored in the array. Table 2 lists the present tree structure as defined by NPOINT. Figure 15 shows a general example of part of the NPOINT structure.

NPOINT (1,N) gives the index location in LWORDS (in common block /HLPWDS/) for the appropriate HELP message for the entry options corresponding to the tree location N. The actual index location given will be for the LWORDS value specifying the number of characters in the HELP message. The actual message is contained in the subsequent LWORDS words in A4 format. Figure 16 shows the structure of a typical HELP message in LWORDS.

NPOINT (2,N) specifies the subsequent tree position for the entry sequence in one of three ways. If the value is zero, the entry sequence (command) is complete. If the value is positive, the three lowest order decimal digits point to the tree location for the next entry. For example, NPOINT(2,N) = 3009, specifies that NPOINT(1,9), NPOINT(2,9) and NPOINT (3,9) provide pointers for the next entry. The thousands digit gives the substep number for the next entry. A negative NPOINT(2,N) value implies a branch in the tree structure at

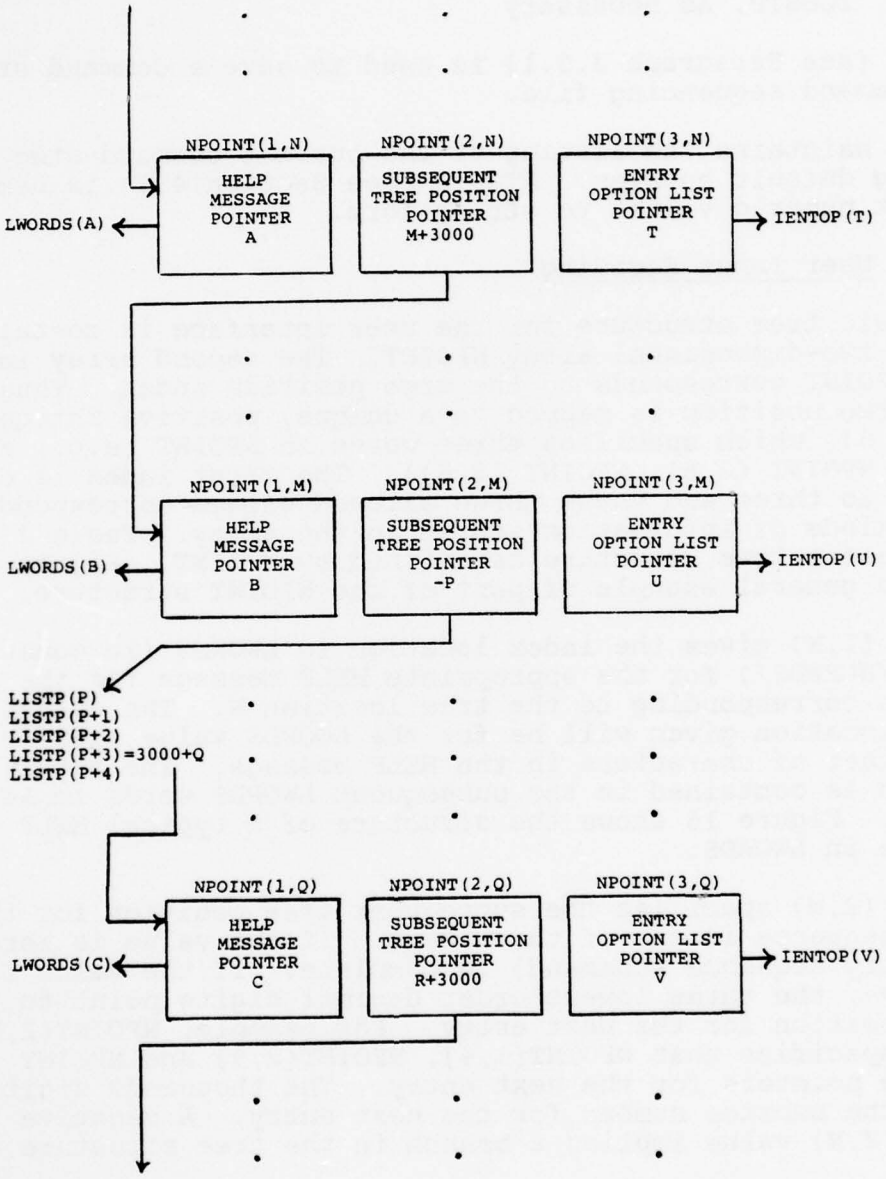


Figure 15. Example of part of the command entry tree structure.

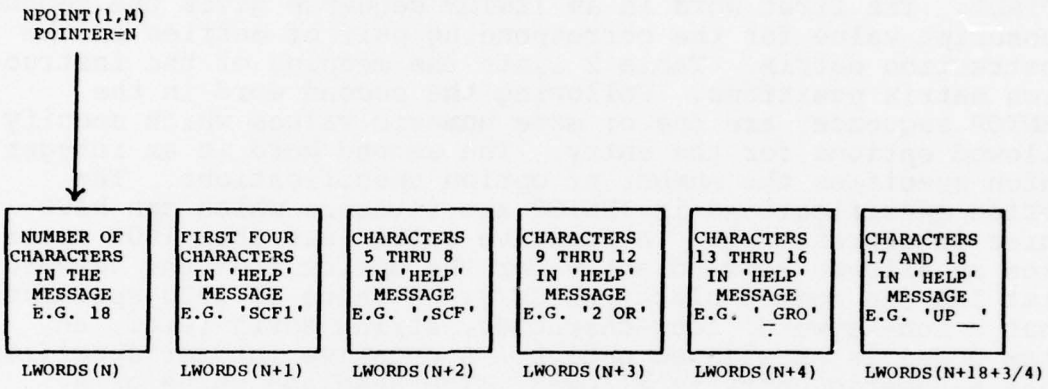


Figure 16. Structure of typical "HELP" message.

the current position and points to a sequence of pointer entrys in the array LISTP (in common block /ENTOPT/). The sequence of values contained in LISTP in turn point to possible next tree positions in a manner identical to that described for positive or zero values of NPOINT(2,N). The manner of choosing the appropriate value from LISTP is described in the next paragraph.

NPOINT(3,N) is an integer value which points to the first position of a sequence in the array IENTOP (in common block /ENTOPT/). Each sequence in the IENTOP array corresponds to one value for the second subscript of the instruction matrix, IDIRCT. The first word in an IENTOP sequence gives the second subscript value for the corresponding pair of entries in the instruction matrix. Table 2 lists the meaning of the instruction matrix positions. Following the second word in the IENTOP sequence, are one or more numeric values which specify allowed options for the entry. The second word is an integer which specifies the number of option specifications. The option specifications in IENTOP are integers which can have three interpretations. A positive value less than 1000 specifies an allowed keyword entry for the option from the keyword list IAA (in common block /WLIST/). A value of 1000 specifies that a non-keyword, four-character, string entry (e.g., an item code) is an allowed option. A negative integer specifies that a numeric entry is allowed. The absolute value of a negative integer points to the first of two floating entries in the RANGOP array (in common block /ENTOPT/) which give the lower and upper bounds for the numeric entry. Numeric or string specifiers must always be the last entry in an IENTOP sequence. Figure 17 shows the structure of a typical IENTOP sequence. If the current NPOINT(2,M) value is negative so as to point to the first element of a sequence in LISTP, each element of this sequence corresponds by sequence position to one of the options from IENTOP. For example, if the LISTP sequence contained the values 3021, 3045, 3064, 3082, the option sequence in IENTOP is 48, 24, 39, 40 and the keyword number 24 is selected, then the next tree position would be 45.

The instruction matrix, IDIRCT, is a two-dimensional array with the first subscript dimensioned to two. The second subscript corresponds to the instruction matrix number as listed in Table 3. IDIRCT(1,N) indicates the selected option for the instruction and IDIRCT(2,N) contains the number or non-keyword string if such an option was selected. An IDIRCT(1,N) value which is positive and less than 1000 indicates the position in the IENTOP option sequence for the option selected. For

TABLE 2. USER INTERFACE TREE STRUCTURE FOR ENTRY SPECIFICATION

TREE LOC	SUB-STEP	ORI-GEN	ENTRY MEANING	NPOINT(1,N) HELP LOC	NPOINT(2,N) NEXT TREE LOC	NPOINT(3,N) OPTIONS LOC
1	1	-	SPECIFICATION ENTRY (1)	1	-1 (2,3,4,5,0, 7,9,0,7,0,13)	1
2	2	1	ANALYZE ACTION SELECTION (2)	31	-16 (14,15,16, 17,19,19)	21
3	2	1	DERIVE ACTION SELECTION (3)	49	-25 (20,21,21, 21,21,10,26,26, 26,29,29,31,31, 33,33,35,38,37, 21,21)	33
4	3,46, 47,	1	DATA SOURCE (SCF1,SCF2, GROUP OR ITEM CODE) (29)	217	-49, (39,39,40, 41)	151
5	3	1	COMMAND SEQUENCE FUNCTION (26)	189	3044	134
6	-	-	-	-	-	-
7	3	1	COMMAND SEQUENCE NAME (25)	183	0	131
8	-	-	-	-	-	-
9	3	1	MENU TYPE (28)	205	0	142
10	2	3	ROTOR RADIUS (17)	146	2025	107
11	3	72	TIME OFFSET (21)	164	3081	119
12	2	14	HARMONIC NUMBER (18)	151	3019	110
13	3	1	COMMENT STRING (27)	196	0	139

TABLE 2. USER INTERFACE TREE STRUCTURE FOR ENTRY SPECIFICATION (CONTINUED)

TREE LOC	SUB-STEP	ORI-GEN	ENTRY MEANING	NPOINT(1,N) HELP LOC	NPOINT(2,N) NEXT TREE LOC	NPOINT(3,N) OPTIONS LOC
14	2	2	NUMBER OF HARMONICS (4)	79	2012	59
15	2	2	UPPER BREAK FREQ (5)	85	2045	62
16	2	2	MAXIMUM FREQ (8)	102	2047	71
17	2	2	DAMPING FREQ (10)	114	3004	80
18	-	-	-	-	-	-
19	3	2, 12	DATA SOURCE (SCF1,SCF2, GROUP OR ITEM CODE) (29)	217	-81 (39,39,27, 72)	151
20	2	3	OUTSIDE AIR TEMP (12)	360	2048	88
21	3	3	COUNTER (19)	156	3022	113
22	3	21	TIME OFFSET (21)	164	3023	119
23	3	22	NUMBER OF CYCLES (24)	179	3024	128
24	3	23	AZIMUTH ANGLE (23)	173	4049	125
25	2	10	OUTSIDE AIR TEMP (12)	360	2050	88
26	3	3	SCF1 OR SCF2 (29)	297	3028	203
27	3	19	GROUP NAME (34)	259	3080	177
28	3	26	ALL OR SCALE VALUE (30)	229	-95 (51,108)	159
29	2	3	OUTSIDE AIR TEMP (12)	360	2030	88

TABLE 2. USER INTERFACE TREE STRUCTURE FOR ENTRY SPECIFICATION (CONTINUED)

TREE LOC	SUB-STEP	ORI-GEN	ENTRY MEANING	NPOINT(1,N) HELP LOC	NPOINT(2,N) NEXT TREE LOC	NPOINT(3,N) OPTIONS LOC
30	2	29	STATIC PRESSURE (13)	369	2052	93
31	2	3	OUTSIDE AIR TEMP (12)	360	2032	88
32	2	31	STATIC PRESSURE (13)	369	2053	93
33	2	3	OUTSIDE AIR TEMP (12)	360	2034	88
34	2	33	STATIC PRESSURE (13)	369	2054	93
35	2	3	HARMONIC NUMBER (18)	151	3061	18
36	3	38	SCF1 OR SCF2 (29)	297	3069	203
37	2	3	OUTSIDE AIR TEMP (12)	360	3021	88
38	2	3	BLADE RADIUS (17)	146	3036	107
39	3	4, 19	ALL OR SCALE VALUE (30)	229	-98 (70,109)	159
40	3	4	GROUP NAME (34)	259	3092	177
41	3	4	COUNTER (19)	159	3042	113
42	3	41	TIME OFFSET (21)	164	3043	119
43	3	42	DURATION (22)	168	4049	122
44	3	5	COMMAND SEQUENCE NAME (25)	183	0	131
45	2	15	LOWER BREAK FREQ (6)	91	2046	65
46	2	45	NUMBER OF POLES (7)	97	3004	68

TABLE 2. USER INTERFACE TREE STRUCTURE FOR ENTRY SPECIFICATION (CONTINUED)

TREE LOC	SUB-STEP	ORI-GEN	ENTRY MEANING	NPOINT(1,N) HELP LOC	NPOINT(2,N) NEXT TREE LOC	NPOINT(3,N) OPTIONS LOC
47	2	16	WINDOW FUNCTION (9)	106	3004	74
48	2	20	STATIC PRESSURE (13)	369	2055	93
49	4	ALL WITH OUTPUT	OUTPUT METHOD (37)	278	-59 (56,56,57 57,58,59,60,60)	189
50	2	25	STATIC PRESSURE (13)	369	3061	93
51	3	28	ALL OR ROW ELEMENT (32)	244	3062	168
52	2	30	VEHICLE WEIGHT (16)	138	2063	104
53	2	32	ROTOR RADIUS (17)	146	3021	107
54	2	34	DETECTOR ANGLE (59)	437	3064	348
55	2	48	TAS CALIB SLOPE (14)	124	2065	98
56	4	49	X-AXIS SCALE (39)	302	4066	210
57	4	49	X-AXIS SCALE (39)	302	0	210
58	4	49	CONTOUR PLOT FORMAT (61)	447	4067	355
59	4	49	SURFACE PLOT FORMAT (61)	447	4068	355
60	4	49	SCF1 OR SCF2 (60)	442	4096	351
61	3	35 50	DATA SOURCE (SCF1, SCF2 GROUP OR ITEM CODE) (29)	217	-69 (39,39,71, 72)	151

TABLE 2. USER INTERFACE TREE STRUCTURE FOR ENTRY SPECIFICATION (CONTINUED)

TREE LOC	SUB-STEP	ORI-GEN	ENTRY MEANING	NPOINT(1,N) HELP LOC	NPOINT(2,N) NEXT TREE LOC	NPOINT(3,N) OPTIONS LOC
62	3	51	ALL OR COLUMN ELEMENT(33)	251	4049	173
63	2	52	ROTOR RADIUS (17)	146	3021	107
64	3	54	DATA SOURCE (SCF1,SCF2 OR GROUP) (29)	455	-75 (73,73,74)	360
65	2	55	TAS CALIB INTERCEPT (15)	130	3021	101
66	4	56	CURSOR STATE (40)	325	4075	227
67	4	58	1ST 3-D PLOT AXIS (39)	462	4076	366
68	4	59	1ST 3-D PLOT AXIS (39)	462	4077	366
69	3	36	ALL OR SCALE VALUE (30)	229	-101 (78,110)	159
70	3	39	ALL OR ROW ELEMENT (32)	244	3079	168
71	3	61	GROUP NAME (34)	259	3088	177
72	3	61	COUNTER (19)	156	3011	113
73	3	64	ALL OR SCALE VALUE (30)	229	-104 (82,111)	159
74	3	64	GROUP NAME (34)	259	3099	177
75	4	66	Y-INTERVAL (47)	379	-87 (84,85,84)	267
76	4	67	2ND 3-D PLOT AXIS (41)	331	4086	232
77	4	68	2ND 3-D PLOT AXIS (41)	331	4087	232

TABLE 2. USER INTERFACE TREE STRUCTURE FOR ENTRY SPECIFICATION (CONTINUED)

TREE LOC	SUB-STEP	ORI-GEN	ENTRY MEANING	NPOINT(1,N) HELP LOC	NPOINT(2,N) NEXT TREE LOC	NPOINT(3,N) OPTIONS LOC
78	3	69	ALL OR ROW ELEMENT (32)	224	3083	168
79	3	70	ALL OR COLUMN ELEMENT (33)	251	3107	173
80	3	27	DOUBLEROW SELECTION (65)	487	3097	381
81	3	72	NUMBER OF CYCLES (24)	179	3090	128
82	3	73	ALL OR ROW ELEMENT (32)	244	3091	168
83	3	78	ALL OR COLUMN ELEMENT (33)	251	4049	173
84	4	75	MIN Y SCALE VALUE (48)	389	4093	274
85	4	75	DECADES IN Y LOG SCALE (56)	422	4093	339
86	4	76	CONTOUR INTERVAL (42)	343	4094	240
87	4	77	OBSERVER LOC IN X (44)	389	4095	249
88	3	71	DOUBLEROW SELECTION (65)	487	3097	381
89	-	-	-	-	-	-
90	3	81	AZIMUTH	173	4049	125
91	3	82	ALL OR COLUMN ELEMENT (33)	251	4049	173
92	3	40	DOUBLEROW SELECTION (65)	487	3105	381
93	4	84, 85	X-INTERVAL (49)	396	-91 (100,101, 100)	280

TABLE 2. USER INTERFACE TREE STRUCTURE FOR ENTRY SPECIFICATION (CONTINUED)

TREE LOC	SUB-STEP	ORI-GEN	ENTRY MEANING	NPOINT(1,N) HELP LOC	NPOINT(2,N) NEXT TREE LOC	NPOINT(3,N) OPTIONS LOC
94	4	86	CONTOUR VALUE (43)	352	0	245
95	4	87	OBSERVER LOC IN Y (45)	517	4102	254
96	4	88	USER SCALE VALUE (55)	413	0	333
97	3	80	ALL OR COLUMN ELEMENT (35)	265	3103	180
98	-	-	-	-	-	-
99	3	92	ALL OR COLUMN ELEMENT	265	3104	180
100	4	93	MIN X SCALE VALUE (35)	406	0	287
101	4	93	DECADES IN X LOG SCALE (57)	427	0	342
102	4	95	OBSERVER LOC IN Z (46)	527	0	259
103	3	97	ALL OR ROW ELEMENT (36)	272	3072	184
104	3	99	ALL OR ROW ELEMENT (36)	272	3072	184
105	3	92	ALL OR COLUMN ELEMENT (35)	265	3106	180
106	3	105	ALL OR ROW ELEMENT (36)	272	3041	184
107	3	79	DOUBLEROW SELECTION (65)	487	4049	381
108	3	28	X-AXIS SCALE (51)	497	3051	293
109	3	39	X-AXIS SCALE (51)	497	3070	293

TABLE 2. USER INTERFACE TREE STRUCTURE FOR ENTRY SPECIFICATION (CONCLUDED)

TREE LOC	SUB-STEP	ORI-GEN	ENTRY MEANING	NPOINT(1,N) HELP LOC	NPOINT(2,N) NEXT TREE LOC	NPOINT(3,N) OPTIONS LOC
110	3	69	X-AXIS SCALE (51)	497	3083	293
111	3	73	X-AXIS SCALE (51)	497	3082	293
112	3	71	DOUBLEROW SELECTION (65)	487	3097	381

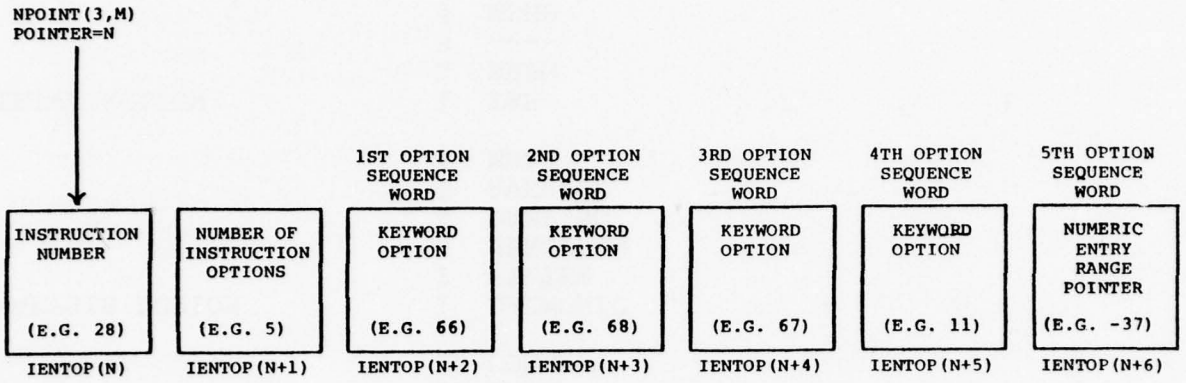


Figure 17. Typical IENTOP instruction option sequence.

TABLE 3. USER INTERFACE INSTRUCTION MATRIX

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS	INITIAL DEFAULT	DEFAULT CONTROL**
1	COMMAND SPECIFICATION	1 ANALYZE 2 DERIVE 3 DISPLAY 4 EDIT 5 NOEDIT 6 EXECUTE 7 ------- 8 MENU 9 TERMINATE 10 BUILD 11 SAVE 12 COMMENT	-	1
2	ANALYSIS ACTION	1 HARMONIC 2 FILTER 3 SPECTRUM 4 DAMPING 5 AVERAGE 6 MMAX	-	1
3	DERIVE ACTION	1 TAS 2 MRPM 3 ---- 4 MSHP 5 ---- 6 CP 7 CN 8 CC 9 CM 10 MCT 11 --- 12 MCQ 13 --- 14 MFLO	-	1

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
		15	DFLO		
		16	BLDIS		
		17	SLOPE		
		18	DENALT		
		19	MRAZ		
		20	----		
4	NUMBER OF HARMONICS	1	1 THRU 24	12	3
5	UPPER BREAK FREQUENCY	1	.1 THRU 1.E+5	-	1
6	LOWER BREAK FREQUENCY	1	0. THRU 1.E+5	0.0	3
7	NUMBER OF POLES	1	2 THRU 7	4	3
8	MAXIMUM FREQUENCY	1	0. THRU 1.E+5	NYQUIST FREQ	3
9	WINDOW FUNCTION	1	COS TAPER	COSINE TAPER	2
		2	HANNING		
		3	NONE		
10	DAMPING FREQUENCY	1	0. THRU 1.E+5	-	1
11		-	-	-	-
12	OUTSIDE AIR TEMPERATURE	1	CALCULATE	CALC	4
		2	-60 THRU +60		
13	STATIC PRESSURE	1	CALCULATE	CALC	4
		2	1.0 THRU 18.0		
14	CONVERSION SLOPE FOR TRUE AIRSPEED CALCULATION	1	.5 THRU 2.0	1.0	5

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
15	CONVERSION INTERCEPT FOR TRUE AIRSPEED CALCULATION	1	-40 THRU +40	0.0	5
16	SHIP GROSS WEIGHT OR COUNTERROTATING FORCE	1	10 THRU 1.E+5	-	6
17	ROTOR RADIUS	1	10 THRU 500	-	6
18	HARMONIC NUMBER	1	1 THRU 24	1	3
19	COUNTER	1	1 THRU 32767	-	6
20	ITEM CODE	1	4 CHARACTER STRING	-	1
21	OFFSET TIME	1	0.0 THRU 1000	-	6
22	DURATION	1	0.0 THRU 1000	-	6
23	AZIMUTH ANGLE	1	0.0 THRU 360	-.01	3
24	NUMBER OF CYCLES	1	-1.1 THRU 1000	1	3
25	COMMAND SEQUENCE BLOCK NAME	1	4 CHARACTER STRING	-	1
26	COMMAND SEQUENCE FUNCTION	1 2 3	NEW CHANGE DELETE	-	1
27	COMMENT	(NOT SAVED, SPECIAL CASE)			-
28	MENU SELECTION	1 2	DATA INFO	-	1

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING		INSTRUCTION OPTIONS	INITIAL DEFAULT	DEFAULT CONTROL**
		3	SCRATCH		
		4	EDIT		
		5	1 THRU 32767		
29	INPUT SOURCE	1	SCF1	-	1
		2	SCF2		
		3	GROUP		
		4	4 CHARACTER STRING (ITEM)		
30	SCF1 OR SCF2 INPUT FIRST DIMENSION	1	ALL	ALL	2
		2	0.0 THRU 1000		
31	-	-	-	-	-
32	SCF1 OR SCF2 INPUT, 2ND DIMENSION EXTENT	1	ALL	ALL	2
		2	1 THRU 64		
33	SCF1 OR SCF2 INPUT, 3RD DIMENSION EXTENT	1	ALL	ALL	2
		2	1 THRU 64		
34	INFO FILE GROUP NAME	1	4 CHARACTER STRING	-	1
35	INFO FILE GROUP INPUT, COLUMN ELEMENT SPECIFICATION	1	ALL	ALL	2
		2	1 THRU 64		
36	INFO FILE GROUP INPUT, ROW ELEMENT SPECIFICATION	1	ALL	ALL	2
		2	1 THRU 64		
37	OUTPUT SELECTION	1	PLOT	NONE	1
		2	MPLOT		
		3	APLOT		

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING		INSTRUCTION OPTIONS	INITIAL DEFAULT	DEFAULT CONTROL**
		4	PRINT		
		5	CONTOUR		
		6	SURFACE		
		7	KEEP		
		8	ADD		
38	-	-	-	-	-
39	OUTPUT SCALE FOR FIRST INDEPENDENT VARIABLE	1	TIME	IMPL	2
		2	FREQ		
		3	HARM		
		4	ROW		
		5	COLUMN		
		6	MRAZ		
		7	TAS		
		8	MRPM		
		9	IMPL		
40	GRAPHICS CURSOR CONTROL	1	CURSOR	CLOSE	2
		2	CLOSE		
41	OUTPUT SCALE FOR SECOND INDEPENDENT VARIABLE	1	ROW	IMPL	2
		2	COLUMN		
		3	MRAZ		
		4	TAS		
		5	MRPM		
		6	IMPL		
42	CONTOUR INTERVAL	1	AUTO	AUTO	2
		2	0 THRU 1.E+6		
43	MINIMUM CONTOUR LEVEL	1	AUTO	AUTO	2
		2	-1.E+6 THRU +1.E+6		

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING		INSTRUCTION OPTIONS	INITIAL DEFAULT	DEFAULT CONTROL**
44	OBSERVER'S EYE POSITION FOR SURFACE PLOT ON THE FIRST INDEPENDENT VARIABLE AXIS (FOR RECT FORMAT) OR IN THE ZERO DEGREE DIRECTION (FOR CYLINDRICAL FORMAT)	1	-1000 THRU +1000	0.0*	3
45	OBSERVER'S EYE POSITION FOR SURFACE PLOT ON THE 2ND INDEPENDENT VARIABLE AXIS (FOR RECT FORMAT) OR IN THE NINETY DEGREE DIRECTION (FOR CYLINDRICAL FORMAT)	1	-1000 THRU +1000	0.0*	3
46	OBSERVER'S EYE POSITION FOR SURFACE PLOT ON THE DEPENDENT VARIABLE AXIS	1	-1000 THRU +1000	0.0*	3
47	INTERVAL BETWEEN LABELED VALUES ON THE 'Y' AXIS (DEPENDENT VARIABLE AXIS) FOR AN X-Y PLOT. LOG SCALE MAY BE SELECTED	1 2 3	AUTO LOG 0 THRU 1.E+7	AUTO	2
48	MINIMUM LABELED VALUE ON THE 'Y' AXIS FOR AN X-Y PLOT	1 2	AUTO -1.E+7 THRU +1.E+7	AUTO	2
49	INTERVAL BETWEEN LABELED VALUES ON THE 'X' AXIS (INDEPENDENT VARIABLE AXIS) FOR AN X-Y PLOT. LOG SCALE MAY BE SPECIFIED.	1 2 3	AUTO LOG 0 THRU 1.E+7	AUTO	2

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING		INSTRUCTION OPTIONS	INITIAL DEFAULT	DEFAULT CONTROL**
50	MINIMUM LABELED VALUE ON THE 'X' AXIS FOR AN X-Y PLOT	1 2	AUTO -1.E+7 THRU +1.E+7	AUTO	2
51	SCF1 OR SCF2 INPUT, SCALE FOR FIRST INDEPENDENT VARIABLE VALUE SPECIFIED AS INSTRUCTION 30	1 2 3 4	IMPL MRAZ TAS MRPM	IMPL	2
52	-	-	-	-	-
53	-	-	-	-	-
54	-	-	-	-	-
55	USER SUPPLIED VALUE FOR COLUMN POSITION	1 2	NONE -1.E+7 THRU +1.E+7	NONE	2
56	NUMBER OF DECADES TO INCLUDE ON THE 'Y' AXIS WHEN LOG SCALING IS CHOSEN FOR THAT AXIS	1	1 THRU 6	3	3
57	NUMBER OF DECADES TO INCLUDE ON THE 'X' AXIS WHEN LOG SCALING IS CHOSEN FOR THAT AXIS	1	1 THRU 6	4	3
58	-	-	-	-	-
59	ANGLE BETWEEN THE INBOARD POINTING DETECTOR OF THE BOUNDARY LAYER BUTTONS AND THE CHORDLINE	1	-90 THRU +90	45	5

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONCLUDED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
60	SCRATCH FILE SELECTION FOR OUTPUT	1	SCF1	-	1
		2	SCF2		
61	FORMAT FOR SURFACE OR CONTOUR PLOT	1	CYLI	RECT	2
		2	RECT		
62	-	-	-	-	-
63	-	-	-	-	-
64	-	-	-	-	-
65	DOUBLEROW ELEMENT SELECTION	1	BOTH	BOTH	2
		2	TOP		
		3	BOTTOM		

* When all three observer eye positions are set to 0.0, then the subroutines which generate the plots supply default values.
**See array IDVAL (common block /DEFLT/) in Appendix B.

example, if the corresponding IENTOP sequence held five allowed options and the third option position was selected, then IDIRCT(1,N) would equal the integer three. An IDIRCT(1,N) value of 1000 indicates that the entry is a non-keyword string held in IDIRCT(2,N). An IDIRCT(1,N) value of -1 indicates that the instruction is a floating number which is held in IDIRCT(2,N). This floating number must be accessed with an equivalenced array which is normally DIRECD.

Default values are coded in the arrays IDVAL and PVAL (in common block /DEFLT/). IDVAL must be dimensioned the same as IDIRCT and the second subscript corresponds to the option number for the options listed in Table 3. IDVAL(1,N) indicates a default control number, and IDVAL(2,N) indicates a default value as described in Appendix B.

3.4 PROCESSING

3.4.1 Processing Flow

The control routine for the processing block is PROCES and the flow for this routine is shown in Figures 18 and 19. PROCES first calls the three routines, PROSET, INPSET and OUTSET, which interrogate the instruction matrix and, as necessary, set up input functions and set control values. The routines also check for errors in the instruction matrix. For example, a reference to a nonexistent Info file group will be detected in routine *COMPGP* which is called by *INPSET*.

PROCES then enters a DO loop which covers the column positions (radial stations). Inside this loop, PROCES calls ATTGET to retrieve and/or calculate the attached parameters for the appropriate counter and time span. ATTGET is called once for each column position since the counter could change with column number if the input is from a scratch file. ATTGET will not recalculate or retrieve the attached parameters if the currently stored attached parameter data are appropriate. In addition, ATTGET will not calculate the attached parameters if these parameters are not required for processing or display in the current step and the output is not to a scratch file.

Following the ATTGET call, PROCES enters a second, nested DO loop which covers the row positions (chord positions). The flow inside this loop is quite straightforward. GETDAT retrieves the appropriate data stream(s) for a row/column intersection, PRO1 calls the appropriate routine to process the data, and TSAV1 stores the data either on the temporary scratch file or on SCF1 or SCF2. GETDAT may retrieve the data stream from the Master File according to the user item code or Info

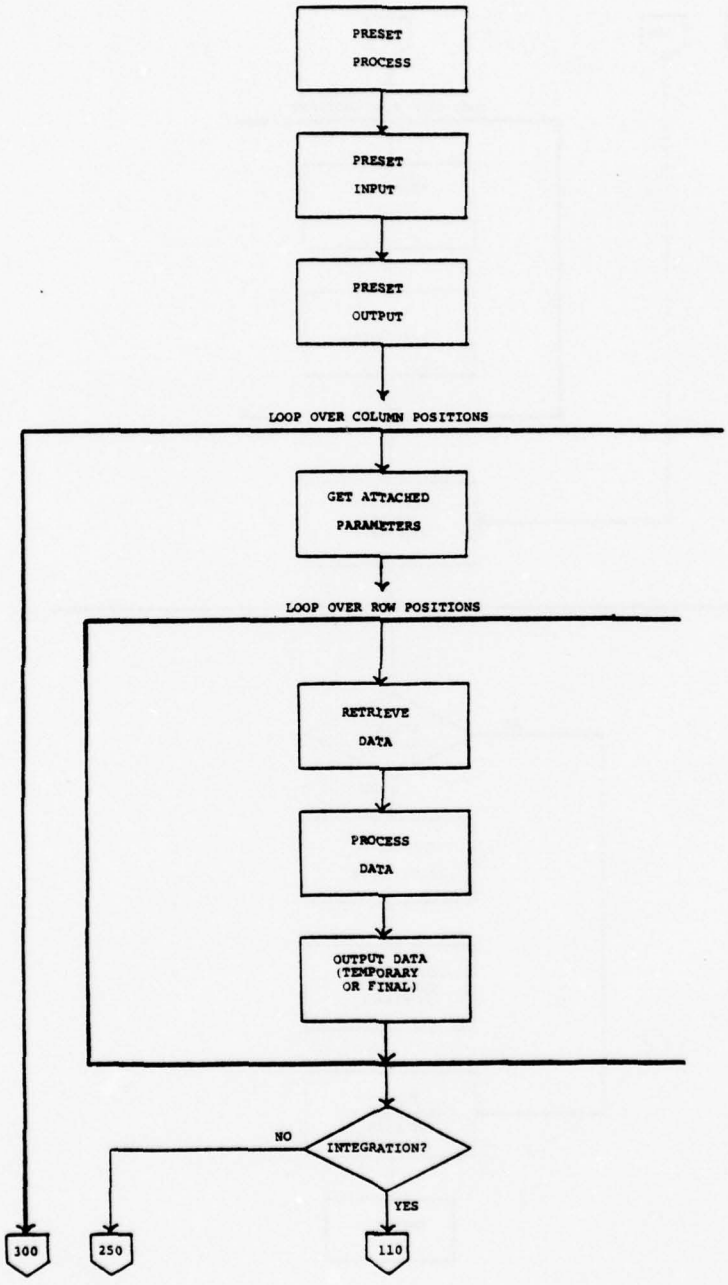


Figure 18. Processing flow diagram (first part).

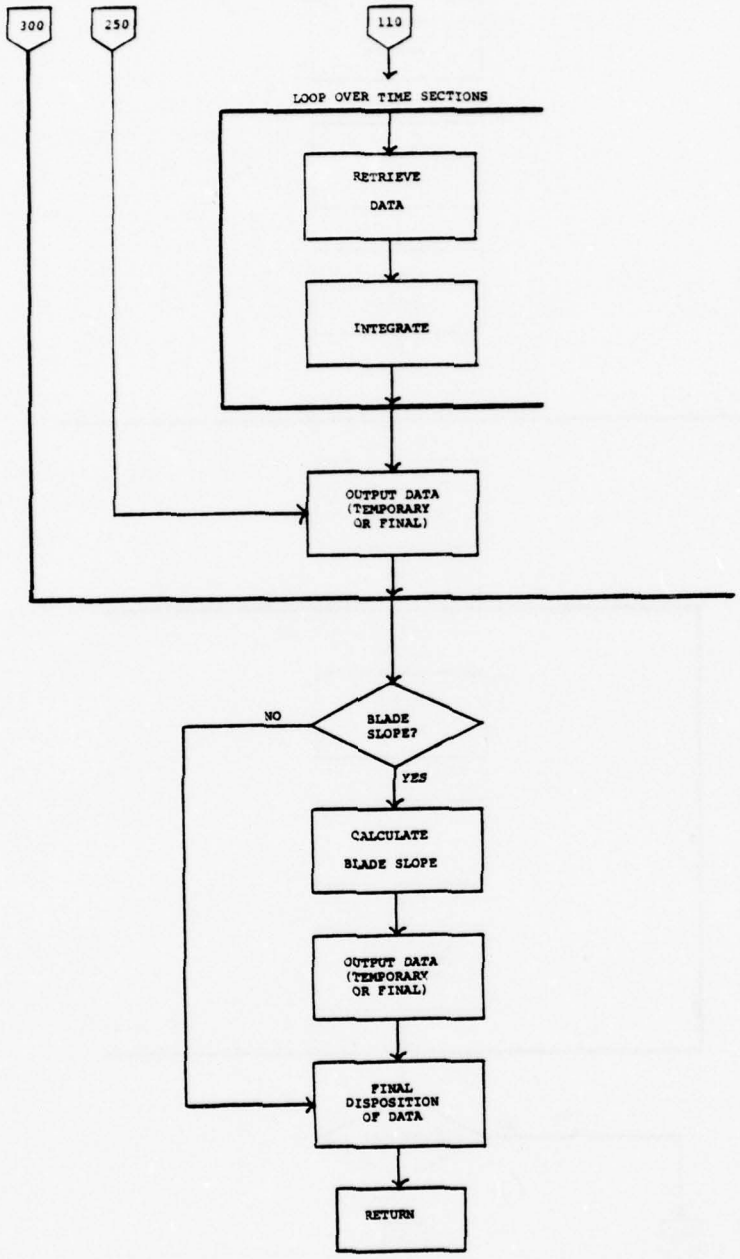


Figure 19. Processing flow diagram (second part).

file specification, or GETDAT may call RTRVSC to retrieve the data from SCF1 or SCF2.

PRO1 addresses most of the available processing routines. PRO1 does not address processes which must treat data from more than one row/column position simultaneously (e.g., C_n integration). For such processes, PRO1 passes the input data straight through to output, treating the data in the same way that data is handled for a DISPLAY command. When the output of a process is to be stored on SCF1 or SCF2 and the processing has been completed by PRO1, TSAV1 calls SCADD to save the data for the current row and column. Otherwise, TSAV1 saves the data in one of three ways. Data streams which contain a single data point are saved in a portion of the array XBUFF. Multiple point data streams are written to the temporary scratch file. However, if a single row position is being processed in the command step, the output data is not written to the temporary scratch file. If the data is to be printed, one of the printout routines (XYPRNT or XYPRN2) is called to print the data stream immediately).

When the row position DO loop completes, PROCES checks whether the specified process is an integration over multiple chord positions (i.e., C_n, C_c, C_p integrations). If not, PROCES jumps ahead to a call to TSAV2. If so, PROCES enters a DO loop which covers data stream sections. Possibly every data point for every row (chord) position will not fit into the program scratch storage array. Thus, each data stream is broken into 128 point (one-half of a scratch file record) sections and all the data for each section is processed simultaneously. GETEMP retrieves the data stream sections from the temporary scratch file and PRO2 selects the appropriate integration. When the loop has covered all the data stream sections, PROCES calls INTEMP to supply the appropriate labels for the process output.

After the call to INTEMP, PROCES calls TSAV2 to store the results of the integration. If the output is to be stored on SCF1 or SCF2, TSAV2 calls SCADD to save the data for the current column. In addition, attached parameter data are stored using more calls to SCADD if those data have not already been stored for the current counter. If the output is printout, TSAV2 calls XYPRNT or XYPRN2 to print the output data stream immediately. When neither of the above output methods is selected and a single-column position is to be processed in the command step, the processed data are left in the scratch storage array, XBUFF. Otherwise, the data are saved by one of two methods. If the output is a single data point for the column, (i.e., one azimuth position), this point is stored in

the XBUFF array. If the output is multiple data points for the column, the data are written to the temporary scratch file.

The same call to TSAV2 may be executed after a jump around the DO loop which performs the integrations. In this case, required storage or printout of the data may have already been performed by TSAV1. If the output is to SCF1 or SCF2 and the column position represents a new counter, TSAV2 calls SCADD to save the attached parameters. If the output is printout, this printout has already been performed in TSAV1. For graphic output, the output data are stored in XBUFF or on temporary scratch unless only one column is to be processed in the command step.

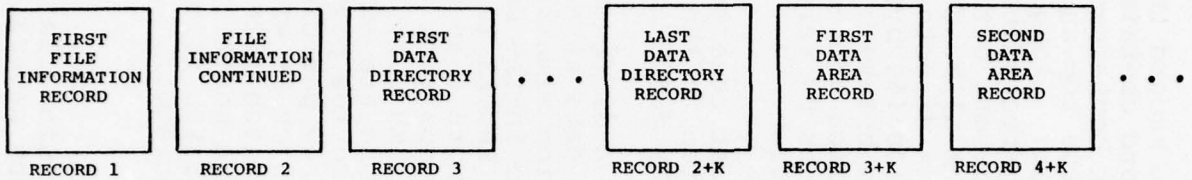
After the loop over the column positions is complete, PROCES checks whether the specified process is a differentiation over the column positions (radial stations). If not, PROCES jumps ahead to call DISPOS. If so, PROCES calls SLOPST to retrieve the appropriate data from the temporary scratch file and to execute SLOPE to calculate the blade slope for each radial position. Then TSAV3 is called to store the output from SLOPST.

The final routine called by PROCES is DISPOS. DISPOS selects the proper routine to perform the output. If the output is to SCF1 or SCF2 or to printout, then the output process has been completed in TSAV1, TSAV2 or TSAV3 and DISPOS simply returns. Otherwise, DISPOS calls the appropriate routine for the graphic output selected: MULTPL for multiple curve X-Y plots, SINGPL for single curve X-Y plots or to add a curve to an X-Y plot, CONSET for a contour plot, and SURSET for a surface plot.

When DISPOS returns, PROCES sets the subroutine argument, IC, to one and returns. MAIN transfers program control back to USER.

3.4.2 Scratch Files

Scratch files SCF1 and SCF2 are written by subroutine SCADD and data are retrieved from these files by subroutine RTRVSC. Subroutine INFSCR is used to obtain information about the contents of a scratch file. The scratch files are direct access and Figure 20 shows the assigned purpose for the scratch file records. The first scratch file record contains labels, information on the data stored, row positions, and column positions. Along with each column position stored, there is a directory for the associated attached parameter



K=((MAXIMUM ALLOWED ROWS) x (MAXIMUM ALLOWED COLUMNS) + 15)/16

Figure 20. Scratch file record assignments.

data and other information including the counter for the column. The column position and attached parameter directions may continue into the second record of the scratch file. Figure 21 shows the contents of the first record of the scratch file.

Data directory records begin at record three of the scratch file. Each data directory record contains several data directory blocks and each block contains the record locations for the data corresponding to the top and bottom double-row elements for one row/column pair. Along with the data location pointers, some information regarding each data stream is included. Space is provided for one data directory block for each matrix intersection for the allowed number of rows and columns. Figure 22 shows a data directory record. Data directory blocks require 16 words (64 bytes) and scratch file records contain 256 words (1024 bytes) so that 16 blocks are written to each record. The block address for a given row/column intersection is determined by varying the row position first and then the column position.

Data records begin after the last reserved data directory record. Data streams are written to the lowest available data records in the order received by SCADD.

The temporary scratch file has a different format from SCF1 or SCF2. This file will not hold data streams corresponding to every row and column intersection simultaneously. The directory for this file is contained in the common block /GENSCR/ (see Appendix B). The flow of PROCES is such that this file should be required to hold no more than one data stream for one row element of each column position and one data stream for each row element of one column position. Data streams corresponding to column positions are entered first followed by data streams corresponding to row positions. When an integration is performed, the row position data streams are condensed to one data stream which is written as a column position data stream on the scratch file. The row position data streams for the next column position must then be written to a higher location on the scratch file to avoid overwriting the new column position data stream.

3.4.3 Info File Retrieval

The information stored on the Info file is retrieved and provided to the Processing Block by several different routines. INFOST is called in the Program Initialization Block to read and transfer the information from the initial group into the

START WORD	LENGTH (WORDS)	ENTRY CONTENTS
1	1	Independent Variable 0=No Data, 1=Time, 2=Freq, 3=Harm
2	2	Overall Units (2A4)
4	4	Row Position Scale Variable (4A4)
8	2	Row Position Scale Variable (Shortened) (2A4)
10	4	Row Position Topographic Feature (4A4)
14	4	Column Position Scale Variable (4A4)
18	2	Column Position Scale Variable (Shortened) (2A4)
20	4	Column Position Topographic Feature (4A4)
24	1	First Dimension Sampling Interval
25	1	First Data Record
26	1	Attached Parameters: 0=Universal, 1=By Column
27	1	LTYPE (from /PRCOM/)
28	1	LXAX (from /PRCOM/)
29	2	Ship Model (A4,A2)
31	2	Ship Number (A4,A2)
33	2	Ship Gross Weight (A4,A2)
35	2	Ship Longitudinal CG (A4,A2)
37	1	Number of Columns Present
38	1	Number of Rows Present
39	1	Top Doublerow Element Keyword
40	1	Bottom Doublerow Element Keyword
41	8	General Label For Data (7A4,A2)
IRPOFF+1	1	Row Position Number 1 Unoccupied=-1.E+35
.	.	.
.	.	.
.	.	.
.	.	Last Available Row Position Unoccupied=-1.E+35
ICPOFF+1	1	Column Position Number 1
" +2	1	First Dimension Value For First Data Point
" +3	1	Data Stream Number of Entries
" +4	1	Record For Azimuth Data
" +5	1	Number of Points of Azimuth Data
" +6	1	Record For Airspeed Data
" +7	1	Number of Points of Airspeed Data
" +8	1	Record For RPM Data
" +9	1	Number of Points of RPM Data
" +10	1	Record For Static Pressure Data
" +11	1	Number of Points of Static Pressure Data
" +12	1	Record For Outside Air Temperature Data
" +13	1	Number of Points of Outside Air Temperature Data
" +14	2	Counter Label (A4,A1)
" +16	2	Flight Date (2A4)
" +18	2	Flight Time (2A4)
" +20	1	Column Position Number 2
.	.	.
.	.	.
.	.	.
.	.	.
.	.	Column Position Number 3
.	.	.
.	.	.

Figure 21. First scratch file record.

START WORD	LENGTH (WORDS)	ENTRY CONTENTS
.	.	.
.	.	.
.	.	.
L+1	1	Top Doublerow Element Start Location in File
L+2	1	Top Doublerow Element Minor Geometric Position
L+3	3	Short Data Stream Label (3A4)
L+6	1	Top Doublerow Element Item Code (A4)
L+7	1	Top Doublerow Element Number of Points
L+8	1	Not Used
L+9	1	Bottom Doublerow Element Start Location in File
L+10	1	Bottom Doublerow Element Minor Geometric Position
L+11	3	Short Data Stream Label (3A4)
L+14	1	Bottom Doublerow Element Item Code (A4)
L+15	1	Bottom Doublerow Element Number of Points
L+16	1	Not Used
.	.	.
.	.	.
.	.	.

L = (MOD(((Column Element - 1)*18 + (Row Element)),16) - 1)*16

Figure 22. Structure of a data directory block.

common block /SINGIF/ (see Appendix B). This common block is interrogated by SINGGP to extract the appropriate item code for a particular keyword.

For other Info file groups, COMPGP is called to read and transfer the group information into the common block /INFGRP/. Then INFO2 is called to provide the proper item code(s) for a specified row/column intersection.

3.4.4 Replacement/Addition of Analysis or Derivation Routines

Most routines which execute specific analyses or derivations on input data are accessed by PRO1 through an interface routine. For example, to calculate blade displacement, PRO1 calls the interface routine DSPSET and DSPSET calls BLDISP to perform the actual calculations. The interface must take the input data as stored in the program and provide these data to the processing subroutine in the required format. The main stream of input data is contained in the array XBUFF. Data for the top double-row element always begin at array element one and data for the bottom double row element begin in the second quarter of XBUFF at location IBFSIZ/4 + 1 where IBFSIZ (in common block /SIZES/) is the array size of XBUFF.

The presence of top and/or bottom double-row elements is indicated by the value M12INP (in common block /CNTLIP/) where the allowed values are:

0 = both double-row elements present
1 = top double-row element present only
2 = bottom double-row element present only

The number of data points in the data stream(s) is given by the two-element array IDATPR (in common block /CNTLIP/) where IDATPR(1) is the number of data points for the top double-row element and IDATPR(2) is the number for the bottom double-row element. Attached parameter data are contained in the common block /ATTPAR/ as explained in Appendix B. Array XSPARE (in common block /BSPARE/) is available for intermediate storage of data.

After the process is completed, the interface routine must assure that the output data streams are stored in XBUFF with the top double-row element data stream starting at XBUFF(1) and the bottom double-row element data stream, if present, starting at XBUFF (IBFSIZ/4 = 1). M12OUT should be set to indicate the presence of the top and/or bottom double-row elements using the same allowed values as M12INP. IDATPR should be set to give the amount of data for each double-row

element. The output keywords, KEYWD1 and KEYWD2, should be set to indicate the type of data present. If a double-row element is not present, the corresponding keywords should be set to zero. If the double-row element is present, there are three cases. The keywords for output from analysis should be identical to the corresponding input keyword, KEYQ1 or KEYQ2. The keyword for a derivation output, which could in turn become the input to a second derivation, should be set by reference to the KWDAT array in subroutine PROSET. The keyword for other derivation output could be set to any non-zero value.

Then the labels and label pointers should be set. When the process is a derivation, the dependent variable description, ITEMDS (in common block /LABELS/) should be changed as necessary along with the dependent variable units, IUNITS. LTYPE (in common block /PRCOM/) should be set to eight. When the process is an analysis, ITEMDS should not be changed but LTYPE should be set to indicate the type of analysis as listed by HLABLS (in common block /PLABLS/). In either case, LXAX (in common block /PRCOM/) should be set to indicate the independent variable as listed by XLABLS (in common block /PLABLS).

3.5 COMMAND SEQUENCING

3.5.1 Command Sequencing File

The Command Sequence File is a direct access file with a structure as shown in Figure 23. Each record contains 16 command lines with 64 characters (16, 4-byte words) per line. Each block requires 7 records for 112 available lines.

The first word of the directory record is an integer which specifies the total number of records in the file. Following the first word is a sequence of two-word entries, corresponding to the command sequence blocks, which gives the four-character block name in the first word and the record location in the second. An empty block is indicated by a blank block name.

3.5.2 Command Sequencing Routines

Access to the Command Sequence file is initialized by the routine EDINIT in the Program Initialization block. EDINIT first reads the directory record for the file and sets certain control values based on the size of the file. Then EDINIT checks the location pointers in the directory record for reasonableness. Finally, EDINIT reads the last file record to check that the command sequence file is properly initialized.

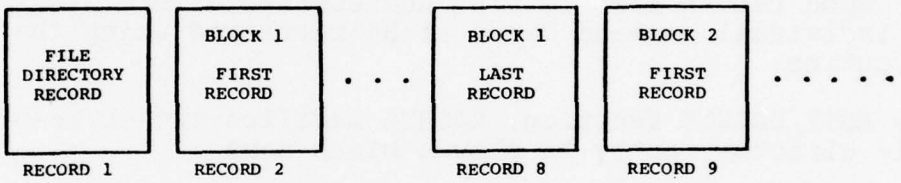


Figure 23. Structure of command sequence file.

The main routine for the Command Sequencing block, EDCNTL, is called to perform any of the functions: EDIT/NEW, EDIT/CHANGE, EDIT/DELETE, BUILD or EXECUTE. EDCNTL first searches the Command Sequence directory record for the block name entered. If the name is found and the function is EDIT/NEW or BUILD, an error message is generated. An error message is also generated when the name is not found and the function is EDIT/CHANGE, EDIT/DELETE or EXECUTE.

For the EDIT/NEW or BUILD function, EDCNTL sets the variable LED (in common block /LEDIT/) to the appropriate value (EDIT/NEW = 1, BUILD = 2) and searches for an unused command sequence block. If a blank block name is found, the name is set to the specified name and the command line storage area is preset with dollar signs for every line. Upon return from EDCNTL, the User Interface block causes the individual command steps to be saved using the EDSAVE routine.

For the EXECUTE function, EDCNTL sets the variable LED to three and sets appropriate pointers for retrieval of the named block. Upon return from EDCNTL, the User Interface block causes individual command steps to be retrieved using the EDINP routine.

For the EDIT/DELETE function, EDCNTL modifies the corresponding file directory entry to show a blank name.

For the EDIT/CHANGE function, EDCNTL reads the indicated command sequence block into the array LINE (in common block /CNGBLK/). Then EDITCH is called to allow the user to modify the sequence. Upon return from EDITCH, the argument ISAVE can have a value of one to indicate the modified sequence should replace the original sequence or zero to indicate that the original sequence should be left unchanged on the command sequence file.

3.6 MENUS

Menu displays are controlled by the routine MENU. This routine simply calls the appropriate routine to create the specified menu display. MCOUNT is called to list the counters present on the Master File partition currently accessed by the program. MITEMS lists the item codes present in the partition for a given counter. INFRED lists the groups present on the Info file. LSCRAT lists the contents of the scratch files. Finally, EDITLS lists the Command Sequence blocks present on the Command Sequence file.

3.7 GRAPHICS

3.7.1 Tektronix/Calcomp Plotting Interface

Plots generated on the Tektronix 4014 screen are nearly identical to corresponding plots generated on a Calcomp or Houston Instruments DP-1. In addition, differences in the source code required for the Batch mode load module and the Interactive Graphics load module are held to a minimum. These features of the software have been implemented through the generation of a group of plotting interface routines and through the use of a modified version of the Calcomp Preview routines. Calcomp Preview is a set of routines provided by Tektronix in the PLOT-10 software. The PLOTS routine supplied by Tektronix has been replaced by a BHT modified version for this specific application.

The plotting interface routines replace the functions of the LINE and AXIS routines. In addition, the plotting interface routines perform five functions required by the Processing Program. First, certain residual differences between calls to the Calcomp routines and calls to the Tektronix and Calcomp Preview routines are handled by this interface. For example, the interface routine STPLT handles the difference between clearing the screen on the Tektronix and moving the plot origin to start a new frame on the Calcomp. Second, the interface routines generate the Tektronix screen format for plotting and also handle positioning of the cursor on the left-hand side of the screen for printed user input and computer messages. Third, data plot curves which exceed the allowed plotting area are clipped by these routines. Fourth, the facility to generate dashed curves is provided by these routines. Finally, access to the graphic cursor and evaluation of cursor-specified locations in user coordinates is provided by the plotting interface.

The plotting interface calls only four Calcomp routines: PLOT, PLOTS, NUMBER and SYMBOL. LINE and AXIS are not used. In addition, three Tektronix PLOT-10 routines are accessed by the plotting interface and three additional PLOT-10 routines are called by the modified PLOTS routine. The modified PLOTS calls the routines INITT, VWINDO and SWINDO. The plotting interface calls the routines MOVABS, ANMODE and VCURSR. Dummy versions of these last three routines are provided for the Batch mode load module.

To begin each plot frame, either STALL or STPLT must be called. STALL should be called for the first plot frame to be

generated by the current program run. STPLT is called otherwise. Following the call to STALL or STPLT, AXES or AREA must be called to define the allowed plotting area. AXES will generate a box around the area, annotate X and Y AXES and, depending on the IGRID and NOTICS settings (in common block /DRW/), draw tic marks inside the box and/or a grid inside the box. AREA will simply define the allowed plotting area without generating any axes.

LYNX is called to draw data curves. LYNX will generate continuous curves or dashed curves and/or curves with characters centered on every N'th point. LYNX cannot draw lines outside the allowed plotting area. DRAWN is called to draw a line outside the allowed plotting area according to the dash code used by the last call to LYNX (see Appendix B).

INSET relocates the cursor on the left-hand side of the screen for printed input or output. The cursor is located on the number of raster points down from the top specified by LNCNT.

PLOC activates the graphic cursor and evaluates the user specified location in units of the current plot frame. The resultant values and the user typed character are returned to the calling routine for processing or output. One position is evaluated for each call to PLOC.

ENPLT ends all plotting by the Processing Program.

3.7.2 X-Y Plots

Both simple and multiple curve X-Y plots are generated through the routine XYPLOT. XYPLOT calls STPLT or STALL as necessary to initialize the plot frame. These routines are not called if a curve is being added to an existing plot frame. X and Y scaling values are determined using SCALEV and the axes are drawn using AXES. This portion of the code is also skipped if a curve is being added to an existing plot frame.

LYNX is then called to draw the curve on the plot. Following the call to LYNX, labels are drawn for the plot if the curve is the first for the current frame. For a multiple curve plot, a sample of the type of dashed line used by LYNX is drawn using the routine DRAWN. This line is then annotated appropriately.

PLOC is then called if graphics cursor activation was specified by the user. INSET is called and the returned arguments from PLOC are printed. If the returned character from PLOC is

a 'C', then the program loops back to call PLOC again. Otherwise the program proceeds to call INSET and return.

3.7.3 Contour Plots

For contour plot generation, subroutine DISPOS calls subroutine CONSET. Based on the two independent variables for the output function, CONSET calls NOFRST or YSFRST. NOFRST is selected when the first or time-related dimension is not one of the two independent variables. YSFRST is called when the first dimension is one of the two independent variables. Both of these routines retrieve the output data and interpolate the data matrix to obtain a data matrix with the prescribed number of rows and columns for the plot format selected.

CONSET then calls either CONCYL or CONREC for a cylindrical or rectangular format, respectively. These two routines follow the same general flow. After STPLT or STALL, and AREA are called, a box or circle is drawn around the allowed plotting area. Then the interval between contour levels is set using SCALEV and/or user supplied values.

When the vertical of Z scale is set, CONTUR is called to draw the contour plot. CONTUR finds the sequences of X-Y positions which form the individual contours. However, CONNEC is called by CONTUR to actually draw the contours using LYNX. In addition, CONNEC uses DRAWN to draw line samples with level annotation in the label area.

Upon return from CONTOUR, additional labels are drawn under the plot and then INSET is called to reposition the cursor for printed I/O for the next command step.

3.7.4 Surface Plots

Surface plot generation follows the same general flow as contour plot generation. Subroutine DISPOS calls subroutine SURSET. Based on the same criterion used by CONSET, SURSET calls NOFRST or YSFRST. Upon return from the selected routine, SURSET calls SURCYL or SURREC to draw a surface plot using respectively a cylindrical or rectangular format.

As with CONCYL and CONREC, SURCYL and SURREC follow the same general flow pattern. Either STPLT or STALL and then AREA are called and a box is drawn around the allowed plotting area. PLSURD is then called to draw the surface. GTFORM is used by PLSURD to generate the perspective transformation from three-dimensional point locations to point locations on a viewing plane.

Upon return from PLSURD, SRRCRF is called to draw annotation around the allowed plotting area.

Upon return from SRRCRF, labels are drawn below the plotting area, INSET is called, and control is transferred from SURCYL or SURREC to SURSET. Next, control is returned to DISPOS.

3.8 DATA RETRIEVAL

Measured data are retrieved from the Master File with the routines DATAIN and FINDIT. FINDIT locates the appropriate data in the Master File. Two separate FINDIT calls are required to locate an item code/counter pair. The first call locates the specified counter in the counter directory and transfers part or all of the corresponding item code directory into the ITEMD array (in common block /DATSET/). The second call locates the specified item code in the item code directory and transfers the information record for the data stream into the ITMINF array (in common block /DATSET/). Both of these calls to FINDIT are performed by DATAIN so that a single call to DATAIN is required to input data for a specified item code/counter pair. Based on the requested time offset and the time history length specified, DATAIN calculates the appropriate first record and reads the requested data. Calibration is performed if the data are stored on the Master File in integer format.

Part or all of the counter directory and the most recently used item code directory are kept in the arrays ICTRD and ITEMD so as to minimize reads of directory records. Thus, if there are fewer than 128 counters, the counter directory need not be read more than once. Similarly, the item code directory need not be read more than once if that directory has fewer than 128 entries and if the counter does not change. In addition, the information record need not be re-read until a different counter/ item code pair is required.

FINDIT checks the required data against the data present to prevent unnecessary reads of directory records and information records.

4. UTILITY ROUTINES

Certain subroutines are used in both the File Creation Program and the Processing Program. These routines have been written to be general in nature.

4.1 DIRECT ACCESS

All direct access READ, WRITE and FIND operations are processed by the routines RMS, WMS and FMS, respectively. For example, instead of a direct access read statement using the IBM format,

```
READ(NRI'IXXX)IARRAY
```

the File Creation Program and Processing Program make the call,

```
CALL RMS(1,IARRAY,ISIZE,IXXX,IERR)
```

Of course, the normal IBM format, or some equivalent format for a different computer system, is used in the RMS, WMS and FMS routines.

The routines use the common block /MASS/ to retain device numbers, offsets and sizes. Calls to RMS, WMS and FMS specify a pseudo-device number which is an index for the arrays in /MASS/. In the example, the integer '1' is the pseudo-device number. The array MDEV contains the actual I/O file numbers (data set reference numbers) for the direct access files. The array MOFF contains offsets to be used in addressing records in the direct access file. Thus, an MOFF value provides relative addressing to a group of contiguous records which form a subset of all the records present in the direct access file. These subsets are called pseudo-devices. Thus, if

```
MDEV (2) = 8
MOFF (2) = 5248
```

then pseudo-device '2' is direct access file number eight and the first record of pseudo-device two is actually record 5249 on direct access file number eight.

The array MLEN gives the number of records assigned to each pseudo-device. The array MTOT gives the total number of records and MSIZ gives the record size in four-byte words for the direct access file which contains the corresponding pseudo-device.

The routines RMS, WMS and FMS check that the requested relative record number is within the assigned pseudo-device area and that the resultant absolute record number does not exceed the boundaries of the corresponding direct access file. These routines also check that the requested record size is less than or equal to the record size for the direct access file.

Initialization of the direct access files and setup of the /MASS/ common block are performed in routines other than RMS, WMS and FMS. The programs and routines which perform initialization on the direct access files are listed in Section 5.1. Setup of the /MASS/ common block is performed for the File Creation Program in routines SETUP1 and SETUP2. For the Processing Program, /MASS/ is set up in the routines INITSC, DASTRT and EDINIT.

IBM OS and MVS system direct access files can be initialized in one of two ways: a write can be specified as the first file operation of the program run or every available record in the file can be written on using a sequential alias for the direct access file number. The former method is not used because the File Creation Program and Processing Program DEFINE FILE statements specify more records for a direct access file than would ever likely be physically provided for the file. Thus, the normal system initialization of the file would always result in an error.

The pseudo-device numbers in the File Creation Program are

1 = Initially is all of Master File and then during the data transfer is the partition of the Master File.
2 = Scratch file temporarily containing the partition directory.

The pseudo-device numbers in the Processing Program are

1 = Initially is set to read all of the Master File and after the Startup Block is executed, is set to access the specified partition of the Master File
2 = Directory for accessed partition
3 = SCF1
4 = SCF2
5 = Temporary Scratch
6 = Command Sequence File

4.2 STRING HANDLING

Several routines are used by the system to process strings. Subroutine PACK transfers the leftmost character (highest

order byte) from each of four sequential words to the leftmost four bytes of a single word. The sequence of the characters is maintained. Thus the sequence of string words 'AWWW', 'BXXX', 'CYYY', 'DZZZ' becomes 'ABCD'.

Subroutine SHFSTR transfers a contiguous sub-string from a string containing four characters per four-byte word to a set of contiguous character locations in a second string containing four characters per four-byte word. Both SHFSTR and PACK use 'LOGICAL*1' variables, which is IBM-dependent code (see Section 5.2).

NTOSTR is a routine which converts floating numeric values to strings. PACK is used to create four-character words. NTOSTR is used by INTERP to convert numeric command entries to string form for storage of command lines.

READF performs scanning and some interpretation of free-field user input lines. The calling routine must read the command line into the array ICHAR (in common block /KARD/) storing one character per four byte word. READF evaluates numeric entries as floating numbers, calculates the starting character position and number of characters for string entries, and notes the position in the entry sequence of null entries. This information is returned in the common block /KARD/.

Subroutine MATCHR is frequently called following a call to READF to find a match between a character string and one element of an array of four-character strings. The first character of the test string is compared with each of the first characters from the keywords stored in the array IAA (in common block /WLIST/). Subsequently, the following character from the test string is compared with the corresponding characters for all the keywords which matched for the previous character. If, after every character of the input string has been processed, there is more than one keyword which compares character for character, the entry is considered ambiguous and the return argument IOUT is given the value zero.

If no keywords match the test string, then IOUT is set to minus one. When a single keyword matches the test string, the corresponding index for the keyword is returned.

A maximum of four characters from the test string are examined by MATCHR. Additional characters are ignored. Fewer than four characters may be provided and then only the characters supplied are processed. If an unambiguous match is found for the test string before all the test characters have been processed, the remainder of the test characters are still compared and a mismatch will result in a return with IOUT= -1.

4.3 SORTING

Several routines are used by the File Creation Program and the Processing Program to sort arrays in ascending order of floating or integer value. These routines are SORTM, SORT1, SORT2, SORT3, and SORTMF. These routines all use the binary sort algorithm and retain the same flow pattern. The routines differ in the number of associated arrays carried along with the array to be sorted and whether the array to be sorted contains floating or integer values.

SORTMF sorts an index array corresponding to the array to be sorted. Then the routine SORTID, which calls SORTMF, carries through the sort using the location pointers in the index array. The sort is carried through on a matrix of array values with the column elements corresponding to the index pointers.

5. TRANSPORTABILITY CONSIDERATIONS

The OLS/DMS has been written so as to make conversion of the software to another computer system as simple as possible. However, certain installation and system dependent code has been required in the programs to achieve the requirements for the system. Such code is always flagged in the source listings and a corresponding process which is valid for the local installation can be inserted in the place of the invalid code. The various types of nontransportable codes will be discussed here.

5.1 DIRECT ACCESS

The OLS/DMS uses the IBM direct access capability extensively. All of the READ, WRITE and FIND calls are restricted to the routines RMS, WMS and FMS. Thus, conversion of the actual reads and writes for direct access files should be reasonably simple if there is a corresponding process at the new installation. In addition, the file definition statements (DEFINE FILE) are always grouped near the beginning of the main routine for each program. The files are always set up with 256 four-byte words per record, the records being unformatted.

The initialization code for direct access files is distributed to four different routines. The Master File itself is initialized in the Master File Initialization Program. The direct access scratch file for the File Creation Program is initialized in the routine SETUP1. The Command Sequence file is initialized by the Command Sequence File Initialization Program. The Processing Program scratch files are initialized by the routine INITSC.

5.2 CODING VARIATIONS

The OLS/DMS uses certain nonstandard IBM FORTRAN features. LOGICAL *1 variables are used in the routines SHFSTR and PACK to address individual bytes of four-byte words for character manipulation.

INTEGER *2 variables are used extensively in the data handling portions of the File Creation Program to process the standard BHT-Ground Data Center (GDC) tape format. The routines which use INTEGER *2 variables are READD, FITEM, FCNTR, TRANSC, CALUPD, SAVD and SAVF. All of these routines would probably need to be replaced in the STRNGF routine to handle non-BHT-GDC tapes.

INTEGER *2 variables are also used in the Processing Program in the data retrieval routine DATAIN.

The following form of input statement is used in the OLS/DMS to detect end files or errors on input.

```
READ(NREA,9000,ERR=500,END=500)list
```

This form is used in the File Creation Program in the subroutine INLIST, in the Processing Program in the subroutines CMPGRP, INFOST, LININP and READ1, in the Master File Initialization Program, in the Command Sequence File Initialization Program, in the Question and Answer Program, and in the Master File Utility Program.

5.3 COMPUTER WORD SIZE PROBLEMS

Strings are stored either with one left justified character per word or with four left justified characters per word. Strings are read or printed in A1 or A4 format. Thus, conversion for string processing should not create much of a problem when at least four characters can be stored per word (installation on mini-computers with 16-bit (2-byte) integers would present significant problems). However, for systems where more than four characters are stored in a word, the calls to the Calcomp SYMBOL routine present a problem since SYMBOL expects a continuous sequence of characters. SYMBOL is called by routines XYPLOT, CONREC, CONCYL, SURREC, SURCYL, ANNOT, and MCHAR.

The READF routine has two integer values set in a data statement, IBITS and NBYT. IBITS must be set to the number of bits in a character byte and NBYT must be set to the number of character bytes which can be stored in a word.

5.4 SPECIAL ROUTINES

Certain installation provided routines are used in the OLS/DMS. Most installations have corresponding routines or, alternatively, the routine functions are not critical to program operation. The function of each of these routines is described here.

Subroutine FASTIO is used to avoid FORTRAN conversion routines for input and output using fixed length records. A second reason for using FASTIO is to read blocks which have no byte count appended. The first argument for FASTIO is one of the character strings 'READ' or 'WRITE' to indicate a sequential input or output operation, respectively. The second argument

is the I/O file number (data set reference number) for the operation. The third argument is the array which contains the data for output or which will receive the data for input. The fourth argument is the number of bytes to be transferred. The fifth and sixth arguments use the IBM system dependent coding technique. The arguments are FORTRAN statement labels for a jump on return from the subroutine. The character '&' is appended to the front of the label in the subroutine argument. The fifth and sixth arguments give the return locations for an end-of-file condition or an error condition, respectively.

Subroutine FASTIO could be replaced, if necessary, by 'A' format READ and WRITE statements (i.e., (3(255A8),45A8) for GDC tapes) or an appropriate system routine (e.g., BUFFER IN for Control Data machines). The detection of end file and error conditions provided by FASTIO is critical only in subroutine READD of the File Creation Program. Subroutine READD is used only for standard BHT-GDC tape input.

Subroutine FASTIO is used in the Processing Program only in subroutine INITSC. In the File Creation Program, FASTIO is used in the following routines: MAKRUM, READD, SAVALL, SAVD, SAVF, SETUP1, and TRANSC. However, the functions of READD, SAVD, SAVF and TRANSC would have to be replaced for non-BHT-GDC data tape formats. FASTIO is also used in the Master File Initialization Program, the Command Sequence File Initialization Program and in the Master File Utility Program.

Subroutine DATE returns the current Gregorian date into eight sequential character locations of the argument array. The format for the returned date is a character string 'mm/dd/yy'. DATE is used in the Processing Program in subroutines DASTRT and LISTER. In the File Creation Program, DATE is used in subroutine RESTRD.

Subroutine TIMOD returns the current time of day into twelve sequential character locations of the argument array. The format for the returned time is a character string 'hh.mm.ss.fr'. The right-most character is set to a blank. TIMOD is called in the Processing Program from subroutine LISTER.

TIMEX and SETIME are entries to DATE which must be used in concert to monitor CPU execution time for the calling program. SETIME is called to initialize the CPU timing process. The argument to SETIME is a REAL value specifying a time limit in minutes. This number must be greater than zero and less than 1440. This argument is not critical to the OLS/DMS application, except that a reasonably large number is defined. TIMEX is called to obtain the CPU time consumed. All arguments are

returned as REAL values. The first argument is the CPU time used since the last call to SETIME. The second argument is the CPU time used since the last call to TIMEX. The third argument is not used by OLS/DMS. This argument gives the time not yet consumed from the interval specified in the call to SETIME. SETIME is called in the Processing Program in subroutine STRTUP. TIMEX is called in the Processing Program in subroutines USER and INISTP.

Subroutine PLTIME is a special routine which estimates the required plotting time for a Calcomp plot and outputs this time to the computer operator. Subroutine PLTIME is called in subroutine ENPLT for Calcomp plots only. For Tektronix plots, the PLOTS subroutine has a dummy entry for PLTIME.

5.5 GRAPHICS

The graphic software was discussed extensively in Section 3.7. However, the graphics features related specifically to transportability of the code are discussed here. For incremental pen output, the software assumes that the Calcomp routines PLOTS, PLOT, SYMBOL, and NUMBER are provided by some system library. These routines must be either the actual Calcomp routines or simulations of these routines for plotting on another device. The plotting interface assumes a plotting area for a plot frame of 8.5 inches horizontal and 11 inches vertical. Approximately 9.7 inches vertical and 7.7 inches horizontal are actually used for a frame. The plotting interface moves to a new plot frame position by incrementing the basic pen origin horizontally to the right at least 8.5 inches. A larger increment may be specified by changing the default value for the variable PLTWID (in common block /MDEP/) or by the user specifying a larger value for PLTWID in the Initialization Phase of a Processing Program run.

Plotting on a graphics terminal assumes that a Tektronix 4014, the PLOT-10 software, and, specifically, the Calcomp Preview package are available. SYMBOL and NUMBER from a Calcomp package are also required if not present in the Calcomp Preview software.

The plotting interface should generate plots without modification on a Tektronix 4010 (the reduction in screen size may make the plots harder to read). However, the different hardware character size could cause printed computer messages and user command input lines to overlap the plotting area. In addition, when the APLOT plotting option is used, INSET will not relocate the cursor to the correct vertical position on the left-hand side of the screen so that printed input and

output lines could overlap each other. This problem can be eliminated by resetting the value of the integer INCTEK (in common block /SIZES/) from -13 to -22. Some of the problems of printed messages overlapping the plot area can be eliminated by resetting the allowed number of characters in an input line to 30. (See common block SIZES in Appendix B.)

Conversion of the program to run on some other graphics terminal would depend on the presence of several software items. A Calcomp emulation package would be required which provided the routines PLOT, PLOTS, NUMBER and SYMBOL. The PLOTS routine, called once for each plot frame for the interactive graphics mode, must set up a simulated plot area of at least 9.51 vertical inches and 7.51 horizontal inches. The origin must be set initially at the lower left-hand corner.

The Tektronix PLOT-10 routine ANMODE is always called when the program changes from drawing plot lines to reading or writing characters. Some corresponding function may be required for other plot devices.

The Tektronix PLOT-10 routine MOVABS is used to reposition the cursor for character input or output after graphics lines and/or characters have been drawn. After a fresh frame has been created, the cursor is moved to the upper left-hand corner of the screen. When a curve is added to an existing frame, the cursor is moved to a raster position at the left-hand side of the screen which corresponds to the next line of character printout after the last line printed or entered. The program keeps track of this position with the variable LNCNT (in common block /STATUS/). When the screen is cleared for a new plot, LNCNT is set to a raster number corresponding to the top line on the screen for alphanumeric I/O. For every alphanumeric line which is read or written, LNCNT is modified by adding INCTEK (in common block /SIZES/). Both the ANMODE and MOVABS calls occur in the subroutine INSET. For application to a different graphics terminal, the ANMODE and MOVABS calls could be replaced and the LNCNT information might or might not be useful.

Subroutine PLOC accesses the virtual cursor using subroutine VCURSR. VCURSR returns the cross-hair location in terms of the simulated 9.51 inches high by 7.51 inches wide plotting area established in PLOTS with a call to VWINDO. For a different graphics device, the graphics cursor function might be eliminated or some substitute for the cursor position evaluation might be found.

6. REFERENCES

1. Shockey, Gerald A., Williamson, Joe W., and Cox, Charles R., AH-1G HELICOPTER AERODYNAMIC AND STRUCTURAL LOADS SURVEY, Bell Helicopter Co., USAAMRDL Technical Report 76-39, Eustis Directorate, U.S. Army Air Mobility Research and Development Laboratory, Fort Eustis, Va., February 1977, AD A036910.

2. Tieman, L. J., 'GROUND DATA CENTER STANDARD DIGITAL TAPE FORMAT,' Bell Helicopter Textron Report 699-099-020, Fort Worth, Texas, 21 April 1976.

APPENDIX A - FILE CREATION PROGRAM COMMON

/CALASC/ Stored initial calibration factors for the item codes present in an assignment record as supplied by that assignment record.

- CM - Array which holds the slope values for calibration. CM(N) corresponds to the N'th requested item code present in the current assignment record.
- CB - Array which holds the intercept values for calibration.

/DATAPE/ Information on the data tape being read.

- IRCNTR - Current record number on the data tape being read.
- NTPERR - Number of tape errors encountered so far in reading the data tapes.

/EXCORB/ Common block used by routines EXSET and EXCORE for extended core simulation. These routines set up a two-dimensional matrix of simulated memory while actually storing and retrieving the values from a direct access disc file.

- NROWSX - Number of rows (most rapidly varying index) in the simultated array.
- NCOLSX - Number of columns (less rapidly varying index) in the simulated array.
- NRECOF - Offset to the first storage record available to EXSET/EXCORE in the direct access file addressed by these routines.
- NRECPR - Direct access record number currently held in the array EXTREC. If NRECPR = -1, no record is present.
- NPRMOD - Indicator of whether the record currently stored in the array EXTREC has been changed without storing the changed version on the disc.

NEXDEV - RMS, WMS, FMS pseudo-device number for direct access storage.

NRSIZE - Size of a direct access record in four-byte words.

EXTREC - Storage array for records from direct access disc.

/FILC/ Convolution filter multipliers.

F1 - Central value for the convolution function.

FILTM - Array holding values for the convolution function.

NFILT - Number of convolution function values held in FILTM.

/FILES/ I/O file reference numbers.

NRPS - Master File (=1)

NRSC - Direct access scratch file (=12)

NSSC - Sequential scratch file (=13)

NITT - Data Tape (=21 for first tape)

NDIR - Time skew alignment tape (=20)

NREA - System input (=5)

NWRI - System output (=6)

NSAV - Digital tape for copy of a partition (=15)

IALI - Sequential alias for the direct access scratch file (=14)

/INFO/ This block contains a set of control and information values for processing and transferring the data.

IRSIZ - Number of two-byte words in a Master File record (=512)

MLOC - Next available partition record number in the data storage area.

IPOLES - Set to zero if no filtering is required for at least one of the requested items present on the current assignment record.

HIGH - Initially set to the lowest digital filtering breakpoint requested for any of the item codes present on the current assignment record. For transfer to the Master File, HIGH is set to the requested digital filter breakpoint for the item to be transferred.

LCAL - Set FALSE if item code data about to be transferred to the Master File is not to be calibrated, set TRUE otherwise.

IRAT - Sample rate for the data on digital tape corresponding to the current assignment record.

ISKIP - Sample rate reduction factor to be applied in transferring data for the currently specified item code to the Master File.

NPS - Initially, the number of points of data wanted for a particular counter given the sample rate of the data on tape. NPS must be divided by ISKIP to calculate the proper number of points to transfer to the Master File. After transfer of a data stream, NPS is the number of data values transferred multiplied by ISKIP.

NPP - This value indicates the number of values for each item code present on the scratch disc file before transfer to the Master File.

NOFF - This value indicates the total number of data samples which should be skipped before data from a particular item code is transferred to the Master File. NOFF should reflect both the alignment and absolute offsets.

ISEQ - This value gives the word position for the first data value on a data record which corresponds to the item code to be transferred to the Master File.

LSTRT - Equivalent to ISEQ.

IADD - Increment which must be applied to a location on an input data record to reach the next data value for the same item code. Equivalent to the number of item codes present in the assignment record.

INSIZ - Number of two-byte integer words in a data record on tape (=3240).

INSIZD - Number of two-byte integer word in a data record stored on the scratch file after the information bytes have been stripped from the front (=3200).

ICNTR - Current counter for the data transferred from tape.

XALIGN - Time skew alignment offset in seconds to be applied to the current item code/counter.

/KARD/ Common block to keep track of data from the current assignment record.

ITEMTP - ITEMTP(N) gives the position in the sequence of item codes present on the current assignment record of the N'th requested item of the items on the assignment record.

ITEMW - ITEMW(N) points to the word in the ITEM array which contains the item code which corresponds to the N'th requested item of the items on the assignment record.

CALSH - The CALSH array gives Calibration Shift values from the assignment or calibration record which correspond to the same item codes as the ITEMTP and ITEMW arrays.

CALCM - The CALCM array gives Calibration Command values from the assignment or calibration record which correspond to the same item codes as the ITEMTP and ITEMW arrays.

CXM - The CXM array gives calibration slope values from the assignment or calibration record which correspond to the same item codes as the ITEMTP and ITEMW arrays.

CXB - The CXB array gives calibration intercept values from the assignment or calibration record which correspond to the same item codes as the ITEMTP and ITEMW arrays.

NMATCH - Number of requested item codes present on the current assignment record.

DCAL - Delta cal values from the assignment or calibration record which correspond to the same item codes as the ITEMTP and ITEMW arrays.

/KARD1/ This common block corresponds identically to common block /KARD/ as described in Appendix B.

/LIST/ This block contains the interpreted user instructions for transfer of data from tape to the Master File.

NCTR - Array of requested counters.

NOFFST - Offsets to be applied to time histories from the counters in NCTR which correspond by index. Offsets are stored in seconds as floating numbers.

NPWANT - Length of time history to be transferred for the counters in NCTR which correspond by index. Times are stored in seconds as floating numbers.

ITEM - Array of requested item codes.

FILT - Break frequencies for low pass digital filters to be applied to the time history from the item codes in ITEM which correspond by index. Break frequencies are stored in Hz. Negative or zero values indicate no filtering should be applied.

ICAL - Indications of whether to store time histories in calibrated or integer format. The indications correspond by index to the item codes in ITEM. ICAL(N) = 0 means no calibration and ICAL(N) = 1 means calibration.

ISKP - Sample rate reduction factors for the time histories from the item codes in ITEM which correspond by index. Values are stored as integers (e.g., a value of four means every fourth sample will be transferred to the Master File).

NITEMS - Number of item codes stored in the ITEM array.

NCNTRS - Number of counters stored in the NCTR array.

ISPAC - Requested number of direct access records for the Master File partition to be created, replaced, or modified.

ITAPES - Number of input data tapes to be read.

IADNU - Mode with regard to Master File partition.
IADNU= 1, Add to existing partition
IADNU= 0, New partition
IADNU=-1, Replace partition

LALIN - Logical variable
LALIN = TRUE., Use time skew alignment
LALIN = FALSE., Do not use time skew alignment

NAME - Array containing partition name.

NPWD - Array containing partition password.

NUSER - Array containing user name.

MAPIT - Logical variable
MAPIT= .TRUE., Generate partition listing after completion of all data transfers.
MAPIT= .FALSE., Do not generate a partition listing.

SAVIT - Logical variable
SAVIT= .TRUE., Save partition on digital tape after completion of all data transfers.
SAVIT= .FALSE., Do not save partition on digital tape.

STRANG - Logical variable.
STRANG=.TRUE., Input data are not in standard BHT-GDC format.

STRANG=.FALSE., Input data tapes are in standard BHT-GDC format.

/LOCOM/ Information for data transfer process.

ITEMN - Current item code to be transferred.

ICNTRM - Current counter to be transferred.

IDROFF - Offset for the partition directory in records.

IDASIZ - Number of records in the partition data area.

IDRSIZ - Number of records in the directory.

ITEMRC - Record number for portion of item code directory which contains current item code.

ITEMSQ - Sequence position in directory record for current item code.

/MASS/ Identical to /MASS/ common block in Appendix B except that arrays are dimensioned to five instead of ten.

/SCRAT/ This common block specifies a general scratch area. Variable arrays are different in name and size for each routine.

/SIZES/ Single word common block.

MAXCI - Maximum number of counters or item codes which may be specified for any one run of the File Creation Program. Corresponds to the array size for the arrays in the /LIST/ common block.

/WLIST/ List of keywords to decode user input instructions.

N - Number of keywords present in the IAA array.

IAA - Two-dimensional array of keywords. Second array index corresponds to the keyword number. The four-character keywords are stored with one left justified character per four-byte word.

/WLIST1/ Coded start and stop times for current counter and item code.

ISTART - Coded start time as described in the BHT-GDC Standard Digital Tape Format.

ISTOP - Currently set to zero.

APPENDIX B

PROCESSING PROGRAM COMMON VARIABLES

/ATTPAR/ Area for storage of processed attached parameter information. The time base for the data stored is normally the sequence of zero degrees azimuth instants. When appropriate azimuth data are not available, this time base is synthesized with an interval between instants of two-tenths of a second.

NVAL - Total number of time instants represented in the time base, TMAZMO. These instants may be either synthesized or real azimuth equal zero degrees time instants as explained above.

NCNTR - Counter which corresponds to the current data stored in ATTPAR.

T1 - First time instant in time base.

T2 - Last time instant in time base.

LTMAZM - Total number of time instants in the time base TMAZMO, which are real azimuth values. The real azimuth values must form a contiguous sequence beginning with TMAZMO(1).

LTASVA - Total number of true airspeed values present in the TASVAL array. If LTASVA is greater than zero, the first TASVAL value must correspond to the first TMAZMO time.

LRPMVA - Total number of rotor speed values present in the RPMVAL array. If LRPMVA is greater than zero, the first RPMVAL value must correspond to the first TMAZMO time.

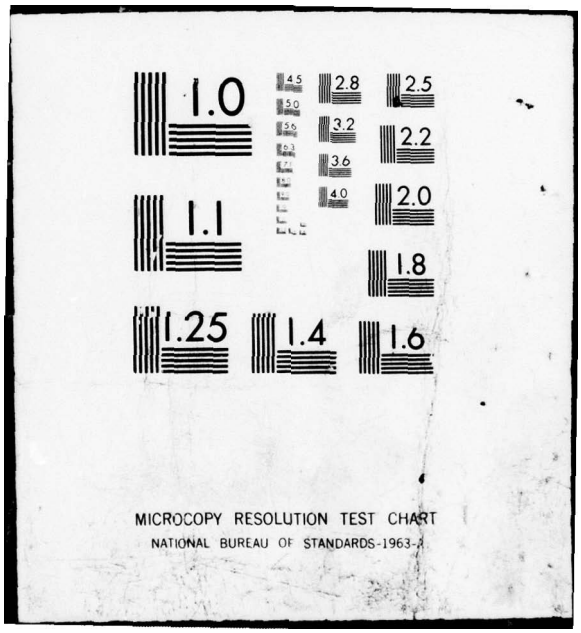
LOATVA - Total number of outside air temperature values present in the OATVAL array. If LOATVA is greater than zero, the first OATVAL value must correspond to the first TMAZMO time.

LSTATV - Total number of static pressure values present in the STATVL array. If LSTATV is greater than zero, the first STATVL value must correspond to the first TMAZMO time.

AD-A065 270 BELL HELICOPTER TEXTRON FORT WORTH TEX F/G 1/2
OPERATIONAL LOADS SURVEY - DATA MANAGEMENT SYSTEM. VOLUME II. S--ETC(U)
JAN 79 R B PHILBRICK, A L EUBANKS DAAJ02-77-C-0053
UNCLASSIFIED BHT-299-099-871-VOL-2 USARTL-TR-78-52B NL

2 OF 2
AD A065270

END
DATE FILMED
4 -79
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
6.3
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

XSTRSC - Time corresponding to the first data value on the scratch file (SCF1 or SCF2) used for input.

XINTSC - Time interval between data values on the scratch file (SCF1 or SCF2) used for input.

NMAXSC - Number of data values present for the first time history on the scratch file (SCF1 or SCF2) used for input.

TMAZMO - Array of time instants forming a time base for the values in the arrays TASVAL, RPMVAL, OATVAL and STATVL. These time instants may or may not correspond to instant of zero degrees azimuth as explained in the heading for common block /ATTPAR/.

TASVAL - Array of true airspeed values in knots.

RPMVAL - Array of rotor speed values in RPM.

OATVAL - Array of outside air temperature values in degrees centigrade.

STATVL - Array of static pressure values in psia.

/BSPARE/ Area for data storage in processing.

XSPARE - Array for storage of data or scales during processing. This array must always be at least one-half the size of XBUFF.

/BUFFER/ Area for data storage in processing.

XBUFF - Array for storage of data or scales during processing. The number of words in this array must correspond to IBFSIZ in the block SIZES.

/CNGBLK/ Communication and work area for command sequence editing function.

NLINES - Number of lines in the command sequence block to be edited.

NAMSEQ - Name of the command sequence block to be edited. Held in 'A4' format.

LOCAT, IDEL1, IDEL2 - Work arrays corresponding to line numbers for command sequence editing.

IWORK - Work array used for display of user input line error diagnostics.

LINE - Array corresponding to command sequence block before, after, and during editing. The second index corresponds to line number. Each line is stored in 16A4 format.

LINECH - Array to hold line changes during editing prior to a renumbering operation ($N).

MERGEL - Array to hold renumbered command sequence block during the renumber operation ($N).

/CNTLIP/ Directive and information values for data input and processing

IPRCOD - Processing code assigned in an ANALYZE, DERIVE or DISPLAY command step. Set in PROSET and interpreted in PRO1 or PRO2.

IPRTYP - Certain types of processes are grouped together for the process flow. IPRTYP = 4 indicates a process using data from multiple row positions for each column position (e.g., a C_n integration) which would be accomplished in PRO2. IPRTYP = 5 indicates a process using data from multiple column positions simultaneously (i.e., blade slope) which is accomplished in SLOPST. Any other value for IPRTYP indicates a process accomplished in PRO1.

NFREE - Source of input data. Allowed values:

1 = SCF1
2 = SCF2
3 = Info file group specifies item code(s)
4 = User specified item code
5 = Info file specifies item code required for derivation by keyword
6 = Attached parameter data is sufficient for derivation

NCOLSI - Number of columns (3rd dimension) to be input for processing.

NROWSI - Number of rows (2nd dimension) to be input for processing.

TIME1 - Time specified by user as either the beginning of the input time history to be used or a time instant included in the rotor cycle just before the beginning of data which will occur at azimuth equals zero degrees.

DURATN - Length of the input time history in seconds when ICYCLS is less than zero.

ICYCLS - Length of the input time history is rotor cycles. ICYCLS = 0 specifies that a single instant corresponding to a user specified azimuth value will be input. ICYCLS less than zero specifies that the length of the time history is given by DURATN.

AZIM - Specifies a single rotor azimuth position for input when ICYCLS = 0.

MISINP - Specified which double-row elements are present input. The values:

0 = Both double-row elements
1 = Top double-row element only
2 = Bottom double-row element only

NCOLI - When NCOLSI = 1, this variable specifies which column element is input.

NROWI - When NROWSI = 1, this variable specifies which row element is input.

IDATPR - A two-element array which specifies how many data points are present in the current input data record. IDATPR(1) corresponds to the top double-row element and IDATPR(2) corresponds to the bottom double-row element.

/CNTLOP/ Directive and information values for data processing and output.

MODOUT - Output mode. Allowed values are:

1 - Plot single curve
2 - Plot multiple curves

3 - Add a curve to an existing plot frame
4 - Print data
5 - Contour plot
6 - Surface plot
7 - Keep results on a scratch file while destroying any data already present on the file
8 - Add results to a scratch file along with any data already present on the file.

ISFOUT - Scratch file to be used for output when MODOUT = 7 or MODOUT = 8. Allowed values are:

1 = SCF1
2 = SCF2

M12OUT - Specifies which double-row elements are present on output. The values are:

0 = Both double-row elements
1 = Top double-row element only
2 = Bottom double-row element only

OUTMAX - Maximum output value from any output time history created during the current command step.

OUTMIN - Minimum output value from any output time history created during the current command step.

LSCALE - Specifies parameter for first independent variable for plot output. Allowed values are:

1 = Time, Frequency or Harmonic Number
2 = Azimuth
3 = True airspeed
4 = Rotor speed

LSCALY - Specifies parameter for second independent variable for plot output. Important only for 3-dimensional plot representations (i.e., SURFACE or CONTOUR). Allowed values are:

1 = Row or column
2 = Azimuth
3 = True airspeed
4 = Rotor speed

/CURRNT/ Block to contain information on status of user interface overlay process.

ISBSTP - Current substep being processed.

IENTRY - Current entry in substep being processed. Set to -1 when substep complete, -2 when command step complete.

ITREE - Current tree position in command input process.

LINHLD - Line held in ICHAR is first line of a new command step when LINHLD = 1. This variable is relevant only when MODSCN = 1 (input scanning only).

IEOF - Normally set to zero. Set to one if end of file condition was found on last system input.

IUENT - Sequence number of entry to be processed on current line of user input.

NUENTS - Number of entries available on current line of user input.

IDEFLT - When set to one, default values are specified for the remainder of the current substep and slash terminating the substep is present. When set to zero, the above conditions do not pertain.

IOPT - Entry option selected for a particular tree position.

NEXT - Number of next substep to be entered.

/DATSET/ Control values and buffer arrays for retrieval of data from the Master File.

ICTRDN - Sequential record number for the portion of the counter directory currently present in ICTRD. If ICTRDN = 0, then no portion of the counter directory is present in ICTRD. The sequential record number need not correspond to the relative record number in the directory.

ITMDN - Sequential record number for the portion of the item code directory currently present in ITEMD. The sequential record number need not correspond to the relative record number in the directory.

ITMDN1 - Relative record number for the portion of the item code directory currently present in ITEMD.

INFOLC - Relative record number for the information record for the current item code and counter.

ICTRD - Array containing all or a portion of the counter directory.

ITEMD - Array containing all or a portion of the item code directory.

ITMINF - Array containing the information record for the current item code/counter pair.

ITMDAT - Array containing a data record for the current item code/counter pair. The particular data record is indicated by ITMDA.

ICTRC - Current counter corresponding to the item code directory present in ITEMD.

ITEMC - Current item code corresponding to the information record in ITMINF and the information record location given by INFOLC.

ITMDA - Record number for the data record contained in ITMDAT as offset from INFOLC. This ITMDA+ INFOLC gives the relative record number for the record in ITMDAT.

ITMPNT - Sequential data point in the current data stream which corresponds to the appropriate next data point if DATAIN is called with the continuation mode (FSEC less than zero).

CB,CM - Calibration factors for integer to floating point conversion during retrieval.

IRAT - Calculated sample rate for the current item code/counter pair data stream.

LAST - Total number of samples in the current item code/counter pair data stream.

ICAL - Equals one if the current item code/counter pair data stream is stored as calibrated data and zero if the data stream is stored as uncalibrated integers.

/DEFLT/ Default user input matrix and general system label.

DEFCOM - General system label. The current date is added to this label in STRTUP. The label is stored 13A4 with additional space available.

IDVAL - Two-dimensional array showing the appropriate defaults for user entries. IDVAL(1,N) controls the nature of the default. L = IDVAL(2,N) gives the actual default value. The possible values for IDVAL(1,N) are:

1 = no default allowed
2 = standard keyword default, L
3 = standard numeric default, IPVAL(L)
4 = keyword default unless there is a previously entered value which then becomes the default
5 = numeric default unless there is a previously entered value which then becomes the default
6 = no standard default but previous entry, if any, becomes the default

IPVAL - Array containing numeric default entries pointed to by IDVAL.

/DIRECD/ Provides user command directives and comment.

IDIRCD - Two-dimensional instruction matrix of user interface entries which is provisional until the command step is complete. When the step is complete, this array is copied to IDIRCT. IDIRCD is commonly equivalanced to DIRCD.

KMMNTD - Provisional comment which is copied to KOMMNT when the step is complete.

NKMMCH - Number of characters in the provisional comment, KMMNTD.

/DIRECT/ User interface communication block

IDIRCT - Two-dimensional instruction matrix containing user interface control values. Each instruction, as indicated in Table 3, will have one or more options and may include a communicated string or numeric value. For instruction N, IDIRCT(1,N) contains the option selection coded as an integer value (which may be negative), and IDIRCT(2,N) contains any string or numeric value communicated. Numeric values communicated in IDIRCT(2,N) are always in floating format and are accessed using an equivalent REAL array which is usually called DIRECD.

/DRW/ Block of plotting information

XMIN - Minimum allowed X value on plot in user coordinates.

DX - Increment in user coordinates of X axis corresponding to one annotated interval on an X-Y plot. On a three-dimensional plot (SURFACE or CONTOUR), DX corresponds to 1 inch in the horizontal direction.

XH - Maximum absolute horizontal position in paper or screen coordinates. Currently set to 7.5

XL - Minimum absolute horizontal position in paper or screen coordinates. Currently set to 0.0

YMIN - Minimum allowed Y value on plot in user coordinates.

DY - Increment in user coordinates of Y axis corresponding to one annotated interval on an X-Y plot. On a three-dimensional plot (SURFACE or CONTOUR), DY corresponds to 1 inch in the vertical direction.

YH - Maximum absolute vertical position in paper or screen coordinates. Currently set to 10.0

YL - Minimum absolute vertical position in paper or screen coordinates. Currently set to 0.0

JUNQ - Array which specifies a schedule for generation of dashed lines. Allowed values for JUNQ are 0 thru 9. When a dashed line is generated, a sequence of dashes having a length of one tenth inch times each integer in sequence is generated. A gap of one-tenth of an inch is inserted between each dash.

INSL - A logical variable. True if point last plotted was inside allowed plotting area; false otherwise.

XOFF - Cumulative X offsets from device origin which have been applied in a frame.

YOFF - Cumulative Y offsets from device origin which have been applied in a frame.

NX - Number of DX intervals in the allowed plotting area.

NY - Number of DY intervals in the allowed plotting area.

IGRID - If IGRID = 1, a grid will be drawn for X-Y plots. If IGRID = 0, the grid will not be drawn.

LOGX - If LOGX = 0, the X scale is linear. If LOGX is greater than zero, the X scale is logarithmic with LOGX cycles.

LOGY - If LOGY = 0, the Y scale is linear. If LOGY is greater than zero, the Y scale is logarithmic with LOGY cycles.

ZSCALE - Scaling factor applied to the plot. DX/ZSCALE corresponds in user coordinates to one inch in the X direction, DY/ZSCALE to one inch in the Y direction. For X-Y plots, ZSCALE = .7. For 3-D plots, ZSCALE = 1.0

NOTICS - If NOTICS = 1, no tic marks will be drawn for the X and Y scales. If NOTICS = 0, tic marks will be drawn.

/ENTOPT/ Entry options and tree structure for user command steps

IENTOP - Array containing sequences of entry options coded by keyword number. If a sequence begins at location 'I' then:

IENTOP(I) = Entry number according to allowed entry list.
IENTOP(I+1) = K = Number of entry options.
IENTOP(I+2) thru IENTOP(I+1+K) = Entry options coded by keyword. If IENTOP(I+1+K) = 1000, the option is a four character string. If IENTOP(I+1+K) = -L is less than zero then the option is a number with allowed range between RANGOP(L) and RANGOP(L+1).

NPOINT - Array containing the tree structure for user input entries. Significance of values is:

NPOINT(1,N) = position in LWORDS giving HELP string for this entry.
NPOINT(2,N) = IENTOP position giving allowed option for this entry.
NPOINT(3,N) = L, points to subsequent entry positions. If L greater than zero, L gives next N subsequent entry. If L = 0, command is complete. If L less than zero, -L points to sequence in LISTP with each LISTP value corresponding to an IENTOP option and giving a new NPOINT position for the subsequent entry.

LISTP - Array of pointers as explained under NPOINT.

RANGOP - Ranges for numerical entries as explained under IENTOP

/FILES/ Input and output file numbers.

NRPS - Master file, file number is normally set to one.

NREA - System input file, file number is normally five.

NWRI - System output file, file number is normally six.

NSC1 - Direct access file corresponding to SCF1 when the scratch files are not concentrated on one file. File number is normally seven.

NSC2 - Direct access file corresponding to SCF2 when the scratch files are not concentrated on one file. File number is normally eight.

NAL1 - Sequential alias for NSC1. File number is normally nine.

NAL2 - Sequential alias for NSC2. File number is normally ten.

NCSG - Direct access file corresponding to temporary scratch file. Alternatively, SCF1, SCF2 and the temporary scratch could be concentrated on this file. File number is normally eleven.

NALG - Sequential alias for NSCG. File number is normally twelve.

NEDI - Direct access file for storage of command sequence blocks. File number is normally thirteen.

NINF - Info file. File number is normally fourteen.

NPRI - File reserved for printout. Currently an alias for NWRI.

/FILLRC/ Contains the parameters which describe the digital filter transfer function in Z-transform space. In particular, the transfer function H(Z) is given by

$$H(Z) = \sum_{K=1}^{N} \frac{AD_k}{1 + Z^{-1}Bl_k} + \sum_{K=1}^{M} \frac{AID_k + AIl_k\, Z^{-1}}{1 + BIl_k Z^{-1} + BI2_k A^{-2}}$$

where the common block variables A0, B1, AI0, AI1, BI1, and BI2 are given by the equation. The variable NRE is related to the number of real poles and is given by N in the equation. Similarly, NCPLX is related to half the number of complex poles and is given by M in the equation. NENDPT is used for double filtering operations and gives the number of values that may be discarded at the end of the time interval.

/GENSCR/ Information and pointers for temporary scratch file.

NEXCLG - Next available record number for storage of data identified by column where a single row is present.

NEXRWG - Next available record number for storage of data identified by row number where multiple rows are present.

IGRWLC - Two-dimensional array giving the starting record number in the temporary scratch file for data from a row element corresponding to the second subscript value. The first subscript corresponds to the top and bottom double-row elements for subscript values of one and two, respectively.

IGRWLN - Two-dimensional array giving the length in data samples for the stored time history from a row element corresponding to the second subscript. The first subscript corresponds to the top and bottom double-row elements for subscript values of one and two respectively.

IGCLLC - Two-dimensional array giving the starting record number for data from a column element corresponding to the second subscript. The first subscript corresponds to the top and bottom double-row elements for subscript values of one and two, respectively.

IGCLLN - Two-dimensional array giving the length in data samples for the stored time history from a column element corresponding to the second subscript. The first subscript corresponds to the top and bottom double-row elements for subscript values of one and two, respectively.

LNLBTP - Three-dimensional array giving labels for annotation of lines on multiple line plots. The first subscript is dimensioned to three and corresponds to three words on twelve allowed characters for the label (3A4). The second subscript corresponds to the top and bottom double-row element for values of one and two, respectively. The third subscript corresponds to row or column position. If multiple columns are present, this subscript corresponds to column position. Otherwise, the subscript corresponds to row position.

/HLPWDS/ Strings and control value for generation of HELP prompting for the user.

LWORDS - Array of strings used in generation of HELP messages. There is one string for each available entry option. The word immediately preceding each string is an integer giving the length of the string in characters. The strings are stored in nA4 format.

IHELP - If IHELP = 1, then HELP is active. If IHELP = 0, then HELP is not active.

/INFGRP/ Block for storage of information provided by an Info file group.

MXGLGP - Number of column elements for the group.

MXRWGP - Number of row elements for the group.

KEYWDT - Four-character keyword corresponding to the top double-row element for the group.

KEYWDB - Four-character keyword corresponding to the bottom double-row element for the group.

NKEYS - Set to one if top double-row element present, set to two if both double-row elements present.

NKPOUT - NKPOUT = 0 if both double-row elements wanted.
NKPOUT = 1 if top double-row element wanted or
NKPOUT = 2 if bottom double-row element wanted.

ROWPGP - Array of geometric row positions.

COLPGP - Array of geometric column positions.

MXITBM - Three-dimensional array giving four-character item codes for row, column, double-row element intersections. The first index gives the double-row element where top and bottom correspond to index values of one and two respectively. The second index gives the column element number and the third index gives the row element number.

POSMX - Three-dimensional array giving a third geometric position parameter (e.g., vertical chord position) for the physical location of sensors corresponding to each item code. The first index gives the double-row element, the second index gives the column element, and the third index gives the row element.

/KARD/ Block for communicating user input lines for scan and return of information about the lines.

ILOC - An array corresponding to the user entries in the line ICHAR. ILOC(I) corresponds to the I'th entry. If ILOC(I) is positive, then the I'th entry is a string beginning at character position ILOC(I). If ILOC(I) is zero, then the I'th entry is a null. If ILOC(I) is negative, then the I'th entry is numeric and -ILOC(I) is the index in the XNUM array for the numeric value.

INUM - An array corresponding to the user entries in the line ICHAR. INUM(I) corresponds to the I'th entry. If the I'th entry is a string, then INUM(I) gives the number of characters in the entry.

XNUM - An array giving numeric values extracted from ICHAR as explained under ILOC.

ICHAR - An array of characters forming one line of user input. The characters are stored in the format 72A1.

/KWCNTL/ Gives prescribed keywords to check that data on scratch file to be used on input are appropriate for certain derivations.

KWMI1 - Prescribed keyword for the top double-row element input for a process.

KWMI2 - Prescribed keyword for the bottom double-row element input for a process.

NKWM - Number of prescribed double-row elements required for process.

/LABELS/ Plot labels. Most of these labels are extracted from the information record preceding each data stream in the Master File.

IDATE - Date the data stream was recorded. The format is 2A4.

ITIME - Time of day the data stream was recorded. The format is 2A4.

ICLABL - Current counter in string format A4,A2.

ITEML - Item code in format A4.

IMODEL - Ship model in format A4,A2.

ISHIPN - Ship number in format A4,A2.

ISHPGW - Ship gross weight in format A4,A2.

IMODLC - Ship model code in format 2A4.

ICGLNG - Ship longitudinal CG in format A4,A2.

ICGCOD - Ship CG code in format A2.

IFLTNM - Flight number in format A4,A2.

IUNITS - Dependent variable units in format 2A4.

ITEMDS - Discription of dependent variable in format 7A4,A2.

LINLAB - Dependent variable label for multiple line plots.

/LEDIT/ Control and information values for command sequence storage on retrieval.

LED - Current command sequence (EDIT) mode.

0 = normal mode
1 = EDIT/NEW mode
2 = BUILD mode
3 = EXECUTE mode

NAMFIL - Four-character name of the current command sequence block being edited, built, or executed.

NUMFIL - Pointer to the current block in the command sequence file being generated or read.

LOCFIL - Pointer to the current line in the command sequence file being generated or read.

LEDLIM - Total number of command lines available on the command sequence file. If LEDLIN = -1 the EDIT capability is not available.

LEDLRC - Number of direct access records in a command sequence block.

LNPREC - Number of command lines which can be stored in a command sequence block.

LWDPLN - Number of words allotted to each command line where four characters are stored in each word.

/MASS/ Offsets, pointers and check values for the direct access routines RMS, WMS, FMS.

NDEVS - Dimension for the arrays in this block.

MDEV - Array giving direct access I/O file numbers. MDEV(I) is the I/O file number for pseudo-device I.

MOFF - Array giving offsets to arrive at correct direct access record numbers. For pseudo-device I, MOFF(I) should be added to the requested record number to arrive at the proper record number for direct access device MDEV(I).

MLEN - Array giving lengths of the pseudo-devices. MLEN(I) is the number of direct access records available to pseudo-device I.

MTOT - Array giving total length of direct access devices. MTOT(I) is the total number of direct access records available on direct access file MDEV(I).

MSIZ - Array giving record size in four-byte words for each pseudo-device. MSIZ(I) is the record size for pseudo-device I.

/MDEP/ Computer, installation, or hardware dependent values

IBAUD - Data communication rate in characters per second between the Tektronix graphics terminal and the computer.

IPLDEV - Plotting device

1 = Calcomp or incremental plotter calcomp emulation (e.g., DP-1)
2 = Other device
3 = Tektronix

PENBGX - Deviation in X of the initial positioning of the incremental plotter pen from the standard starting position which is 1/2-inch to the right of the perforations for DP-1 paper.

PENBGY - Deviation in Y of the initial positioning of the incremental plotter pen from the standard starting position which is the 1/2-inch above the perforations at the bottom of the page.

PLTWID - Total width in inches of a page of plot for determining spacing of frames. This value does not affect the size of the plot frames as drawn.

NPBLKS - Number of blocks allowed in a page of printout where each block contains five data lines and one blank line.

TWARN - Number of CPU seconds which will be consumed before the computer begins to issue time warnings to the user.

ITSTEP - Control value for printout of command step execution times. If ITSTEP = 1, the times will be printed. If ITSTEP = 2, the times will not be printed.

IDONE - Array indicating whether the various scratch files are already initialized. If IDONE(I) = 1, then file I is initialized before the program run began. Otherwise, the file must be initialized at the start of the run. Following are the files corresponding to the index values.

1 = SCF1
2 = SCF2
3 = Temporary scratch.

/MENBUF/ Buffer block for menu generation

IX - Array for generation of menu's

/MLABLS/ Block for output labels.

RDLBL - Row axis label of up to 16 characters stored 4A4.

RULBL - Abbreviated row axis label of up to eight characters stored 2A4.

RTLBL - Label for a geographic feature near the lowest valued row position. The label may contain up to 16 characters stored 4A4.

CDLBL - Column axis label of up to 16 characters stored 4A4.

CULBL - Abbreviated column axis label of up to eight characters stored 2A4.

CTLBL - Label for a geographic feature near the lowest valued column position. The label may contain up to 16 characters stored 4A4.

LABGEN - General label for independent variable(s). This label is entered when a derivation is performed or multiple items are used from an Info file. Stored as 7A4,A2 format.

IPCTL - Control for counter label ICLABL.

1 = Single counter in output
2 = Multiple counters in output

IPCLBL - Counter label. Contains counter in string form for single counter output or the string 'MULTIPLE' for multiple counter output.

LBCEX1 - Control for row, column, or time label. The allowed values are:

1 = Column position label in Info file supplied coordinates
2 = Column position label as provided by the user
3 = Row position label in Info file supplied coordinates
4 = Time associated label
5 = No label

LABEX1 - Label as controlled by LBCEX1. LABEX1(1) contains the numeric value while LABEX1(2) and LABEX1(3) contain a string label. XLBCEX is normally equivalenced to LABEX1.

/MODES/ Operating modes for the program.

MODES - Batch/interactive mode selection.

1 = Batch
2 = Interactive
3 = Interactive graphics

MODSCN - Scan mode for user input.

0 = Normal
1 = Scan for line errors only
2 = Same as '1' but next input line is already present in ICHAR

MODINP - Command input source.

0 = System input
1 = Edit file

MODSCR - Scratch file mode. If MODSCR = 1, all scratch files are concentrated on the device with the

number given by NSCG. If MODSCR = 0, each scratch file is located with a different file number given by NSC1, NSC2 and NSCG.

MODROT - Rotor selection mode. If MODROT = 1, the main rotor is selected. If MODROT = 2, the tail rotor is selected.

/PLABLS/ Stored labels and information for output.

HLABLS - An array containing eight labels to be added to the beginning of the dependent variable description. Each label has the format 5A4.

XLABLS - An array containing seven possible X-axis labels. Each label is stored in the format 6A4.

LINSKP - Array of integers which provide the schedules for dashed lines which are later stored in JUNQ(/DRW/). From a LINSKP entry, each decimal digit is transferred to one JUNQ value. Allowed values for each LINSKP entry are 0 thru 9999.

ULABLS - First independent variable unit labels.

/PRCOM/ Common for process communication.

KOUNTR - Current counter stored as an integer

KITEM - Current item code stored in A4 format.

NPTS - Number of points in output record.

XSTRTV - Starting independent variable value.

XINTVL - Independent variable sampling interval.

NMAXVL - Number of samples in processing record.

INDEPN - First independent variable indicator

1 = time
2 = frequency
3 = harmonic number

LTYPE - Pointer to proper HLABLS label

LXAX - Pointer to proper XLABLS label

KEYWD1 - Top double-row element keyword for output data stream.

KEYWD2 - Bottom double-row element keyword for output data stream.

KEYQ1 - Top double-row element keyword for input data stream.

KEYQ2 - Bottom double-row element keyword for input data stream.

POSZ - Two word array giving the third or minor position value for the current item code(s) in process. The first array value corresponds to the top double-row element and the second array value corresponds to the bottom double-row element.

/SCRTCH/

IOFFXB - Offset in words to the beginning of scratch file output information stored in XBUFF (/BUFFER/)

IRPOFF - Offset in words from XBUFF(1) to the beginning of row position storage for output to a scratch file.

ICPOFF - Offset in words from XBUFF(1) to the beginning of column position storage for output to a scratch file.

KDROFF - Offset in words from XBUFF(1) to the beginning of the data directory buffer to scratch file ouput.

NPREC - Number of data directory blocks in a scratch file record.

ICOLM - Array giving the current column number being worked on for each scratch file.

INILOC - First available data record for either scratch file.

MXRWSC - Maximum number of row positions allowed for a scratch file.

MXCLSC - Maximum number of column positions allowed for a scratch file.

MSCLOC - Array giving the next available data storage record for each scratch file.

IPANAV - Array which gives the multiple storage condition for each scratch file where index = 1 is SCF1 and index = 2 is SCF2. If IPANAV(I) = 0 then all the data stored on the scratch file was written in one KEEP command step. If IPANAV (I) = 1 then the data stored on the scratch file was written with one KEEP and one or more ADD command steps.

ICURR - Data directory record currently in XBUFF for the scratch file currently being written on.

IXBINP - Offset in words to the beginning of scratch file input information stored in XBUFF(/BUFFER/)

ICURIP - Data directory record currently in XBUFF for the scratch file currently being read from.

IXDIRI - XBUFF offset to directory record buffer area for scratch file input.

IXDAT1 - XBUFF offset to data record buffer area for scratch input.

IRPOFX - XBUFF offset to row position storage for scratch file input.

ICPOFX - XBUFF offset to column position storage for scratch file input.

MODE12 - Indicator for one or both double-row elements. MODE12 = 1 implies one double-row element while MODE12 = 2 implies both double-row elements.

ROWPOS - Array which contains physical row element positions.

COLPOS - Array which contains physical column element positions.

MAXCOL - Number of column positions present.

MAXROW - Number of row positions present.

ZMAX - Maximum data value present in a row/column pair time history.

ZMIN - Minimum data value present in a row/column pair time history.

LABCNT - Label control value for generation of LINLAB (in /LABELS/) when input is from a scratch file or multiple items are specified by an Info file group. Allowed values are:

1 = Column position label
2 = Column label using user-supplied coordinates
3 = Row position label
4 = Time related label
5 = Originally saved label

POSUP - Array of minor positions (e.g., vertical position on chord section) corresponding by index to the row elements, and also to the top double-row element.

POSDN - Array of minor positions (e.g., vertical position on chord section) corresponding by index to the row elements, and also to the bottom double-row element.

/SINGIF/ Information extracted from the initial group of the Info file.

KEYWRD - Array of four-character keywords which give the meaning for the corresponding item codes.

ITEMK - Two-dimensional array of item codes which correspond by the first index of the array to the KEYWRD with the same index.

VALUES - Two-dimensional array of numeric values which correspond by both indices to the item codes in ITEMK.

/SIZES/ Various fixed numeric values for the program.

IBFSIZ - Size in words of the XBUFF scratch data area.

IMRSIZ - Size in words of a Master File record.

ISCSIZ - Size of the scratch files in records.

ISRSIZ - Size in words of a scratch file record.

ISPSIZ - Size of the scratch files in records when all scratch file pseudo-devices are assigned to the same I/O file number.

ICOLMS - Number of characters allowed in a user input line.

INCTEK - Vertical raster spaces required for each character line printed on the Tektronix.

ICOMSZ - Number of characters allowed in the user comment line.

NDIRCS - Number of user entry options in the user interface output matrix, IDIRCT.

IBIG - A large integer for use as a dummy limit for DO loops.

MENBSZ - Size in words of the IX scratch area for the menu generation routines.

IDBlSZ - Size of the initial part of the scratch file directory.

ICLDSZ - Size in words of a scratch file column directory block.

IDBLKZ - Size in words of a data directory block.

NKV - Number of words of keyword information held by the routine PROSET for each process option.

NKEYSD - Maximum number of keywords allowed for the initial group of the Info file.

NITMSD - Maximum number of item codes which may be associated with each keyword in the initial group of the Info file.

ICOLIF - Maximum number of character positions available for one line of the Info file.

MAXATT - Maximum number of values for each of the attached parameters.

INCLMS - Maximum number of character positions for a line in the Info file.

NCONRW - Specified number of rows for final output matrix for generation of a contour plot when the independent variables are the third and second dimensions.

NCONCL - Specified number of columns for final output matrix for generation of a contour plot when the independent variables are the third and second dimensions.

NCNRW1 - Array giving the specified number of rows for final output matrix for generation of a contour plot when the independent variables include the first dimension. When the plot format is cylindrical, the first array value is used. When the plot format is rectangular, the second array value is used.

NCNCL1 - Array similar to NCNRW1 giving the specified number of columns for the final output matrix for generation of a contour plot.

NSURRW - Specified number of rows for final output maxtrix for generation of a surface plot when the independent variables are the third and second dimensions.

NSURCL - Specified number of columns for final output matrix for generation of a surface plot when the independent variables are the third and second dimensions.

NSRRW1 - Array giving the specified number of rows for the final output matrix for generation of a surface plot when the independent variables include the first dimension. When the plot format is cylindrical, the first array value is used. When the plot format is rectangular, the second array value is used.

NSRCL1 - Array similar to NSRRW1 giving the specified number of columns for the final output matrix for generation of a surface plot.

MXEDLN - Maximum number of command lines which can be stored in a command sequence block.

NEDSIZ - Number of computer words in a command sequence file (Edit file) direct access record.

NEDCHR - Number of characters which are stored for a command line of the command sequence file (Edit file).

NPCYAV - Specified number of data values which are generated to represent one rotor cycle in the cycle averaging (AVERAGE) process.

/SLIST/ Block to contain listing of the developing command step or, if no entries have been made for the current step, contain a listing of the previous command step.

NCPOS - Character position on the ISLIST line currently being generated.

NCROW - ISLIST line currently being generated corresponding to the second index of ISLIST.

ISLIST - Array which contains listing of the developing command step. The lines are stored in 18A4 format with the second index referencing the lines.

ISLNOW - Indicates whether ISLIST contains the currently developing step or the previously completed step listing. Allowed values:

0 = listing of currently developing step
1 = listing of previous step

/STATUS/ Various information on the status of the program.

LNCNT - Vertical raster position on the Tektronix screen for return of the cursor after a plot is generated.

KOMMNT - Array containing the current user comment for output. Comment is stored in 18A4 format.

NKOMCH - Number of the last non-blank character currently in KOMMNT.

IFRSTP - Indicator for the current plot frame. Allowed values are:

0 = No plots have been generated in this run.
-1 = Single curve plot frame was just generated.
I = (positive) Multiple curve plot frame is currently on screen or paper containing I curves.

/SURPLT/ Control and label values for surface on contour plot generation.

ROW1 - Numerically lowest row position for the final output matrix used for surface or contour plot generation.

ROW2 - Numerically highest row position for the final output matrix used for surface on contour plot generation.

COL1 - Numerically lowest column position for the final output matrix used for surface or contour plot generation.

COL2 - Numerically highest column position for the final output matrix used for surface or contour plot generation.

NCR - Number of rows for the final output matrix used for surface or contour plot generation.

NCC - Number of columns for the final output matrix used for surface or contour plot generation.

NTYPEF - Format for contour or surface plot.

1 = Cylindrical format
2 = Rectangular format

ITRNPS - Indicates whether rows and columns should be transposed in generation of the final output matrix.

0 = No transposition
1 = Transposition

DELZ - Dependent variable increment between contour levels for a contour plot.

LABVRT - Label for the vertical axis of a rectangular format contour plot.

LABTOP - Geographic feature label to be placed at the top of a rectangular format contour plot.

LABHOR - Label for the horizontal axis of a rectangular format contour plot.

SETLEV - Contour height indicator for CONNEC routine. For each plot, this value is initially set to -1×10^{35} as an indicator that no contours have been drawn yet.

/WLIST/ Keyword block.

NWDS - Number of keywords stored in IAA.

IAA - Two-dimensional array containing keywords to be matched with user command entries. Keywords are four characters long stored one character per word in 4A1 format. The second array index corresponds to the keyword number.

APPENDIX C

JOB CONTROL LANGUAGE (JCL)

TO EXECUTE THE FILE CREATION PROGRAM.

```
//ESAR1 JOB (FEAR0100,G38,670612,DP38,TS),'DICK',
//    MSGLEVEL=1,MSGCLASS=A,CLASS=T,NOTIFY=ESAR
/*SETUP          DSN=ENGR.NEWOLS
/*SETUP          DSN=ENGR.F2110423
/*SETUP          DSN=ENGR.F2120423
/*SETUP          DSN=ENGR.F2130423
/*SETUP          DSN=ENGR.F2010415
/*SETUP          DSN=ENGR.F2150423
/*SETUP          DSN=ENGR.F1830414
/*SETUP          DSN=ENGR.F1900415
/*SETUP          DSN=ENGR.F1910415
/*SETUP          DSN=ENGR.F1940415
/*SETUP          DSN=ENGR.F1950415
/*SETUP          DSN=ENGR.F1960415
/*SETUP          DSN=ENGR.F2040417
/*SETUP          DSN=ENGR.F2080419
/*SETUP          DSN=ENGR.F2350424
/*SETUP          DSN=ENGR.F2400423
/*SETUP          DSN=ENGR.F2100423
/*SETUP          DSN=ENGR.F2030417
//STEP1 EXEC PGM=FEAR01,TIME=10
//STEPLIB DD DSN=ENGR.PROD1,DISP=SHR
//FT01F001 DD DSN=ESAR.OLSMAS,DISP=OLD
//FT05F001 DD DSN=ESAR.ESAR.BATCHIN.DATA,DATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//FT14F001 DD UNIT=SYSDA,DSN=ENGR.UNIT14,DISP=NEW,
//    SPACE=(CYL,5),DCB=(RECFM=F,BLKSIZE=1024,DSORG=DA)
//FT12F001 DD DSN=ENGR.UNIT14,UNIT=AFF=FT14F001,VOL=REF=*.FT14F001,
//    DISP=OLD,DCB=(RECFM=F,BLKSIZE=1024,DSORG=DA)
//FT13F001 DD UNIT=SYSDA,DISP=NEW,SPACE=(CYL,2),
//    DCB=(RECFM=F,BLKSIZE=6400)
//SYSUDUMP DD SYSOUT=A
//FT20F001 DD UNIT=TPS,DSN=ENGR.NEWOLS,DISP=OLD
//FT21F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2110423,DISP=OLD
//FT22F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2120423,DISP=OLD
//FT23F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2130423,DISP=OLD
//FT24F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2010415,DISP=OLD
//FT25F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2150423,DISP=OLD
//FT26F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F1830414,DISP=OLD
//FT27F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F1900415,DISP=OLD
//FT28F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F1910415,DISP=OLD
//FT29F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F1940415,DISP=OLD
//FT30F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F1950415,DISP=OLD
//FT31F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F1960415,DISP=OLD
//FT32F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2040417,DISP=OLD
//FT33F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2080419,DISP=OLD
//FT34F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2350424,DISP=OLD
//FT35F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2400423,DISP=OLD
//FT36F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2100423,DISP=OLD
//FT37F001 DD UNIT=AFF=FT20F001,
//     DSN=ENGR.F2030417,DISP=OLD
//
```

TO EXECUTE THE PROCESSING PROGRAM IN BATCH.

```
//ESAR6    JOB (OLSDMS00,G38,67061200,DP38,T,02),'DICK   2841',
//       MSGLEVEL=1,NOTIFY=ESAR,MSGCLASS=A,TIME=(2,00),CLASS=G
//    EXEC PGM=ATAR02
//STEPLIB DD DSN=ENGR.PROD1,DISP=SHR
//FT05F001 DD DDNAME=IN
//FT06F001 DD SYSOUT=A
//FT01F001 DD DSN=ESAR.OLSMAS,DISP=SHR
//FT12F001 DD UNIT=SYSDA,DSN=ENGR.UNIT12,DISP=NEW,
//         SPACE=(CYL,2),DCB=(RECFM=F,BLKSIZE=1024,DSORG=DA)
//FT11F001 DD DSN=ENGR.UNIT12,UNIT=AFF=FT12F001,VOL=REF=*.FT12F001,
//         DISP=OLD,DCB=(RECFM=F,BLKSIZE=1024,DSORG=DA)
//FT14F001 DD DSN=ESAR.INFOHT.FORT,DISP=SHR
//FT13F001 DD DSN=ESAR.COMSEQ,DISP=SHR
//PLOTTAPE DD UNIT=(TPS,,DEFER),VOLUME=PRIVATE,
//     LABEL=EXPDT=98005
//IN DD *
1
GRID
STEP
YES
JIM
EXEC/AAAA/
TERM/
/*
//
```

TSO 'CLIST' TO EXECUTE THE INTERACTIVE MODE OF THE PROCESSING PROGRAM.

```
PROC 2 EDIT INFO
CONTROL NOMSG
FREE DA('ESAR.CLIST')
FREE F(FT01F001 FT05F001 FT06F001 FT22F001)
FREE F(FT11F001 FT12F001 FT13F001 FT14F001)
FREE ATTR(SCRATX)
CONTROL MSG
ATTR SCRATX RECFM(F) DSORG(DA) LRECL(1024) BLKSIZE(1024)
ALLOC DA(VTEMP) USING(SCRATX) NEW SP(2) CYL
ALLOC F(FT12F001) DA(VTEMP) SHR
ALLOC F(FT11F001) DA(VTEMP) SHR
ALLOC F(FT13F001) DA('&EDIT.') SHR
ALLOC F(FT01F001) DA(OLSMAS) SHR
ALLOC F(FT14F001) DA('&INFO.') SHR
ALLOC F(FT05F001) DA(*)
ALLOC F(FT06F001) DA(*)
ALLOC F(FT22F001) DUMMY
CALL 'ENGTEST(ATAR00)'
FREE F(FT01F001 FT22F001)
FREE F(FT11F001 FT12F001 FT13F001 FT14F001)
DELETE VTEMP
FREE ATTR(SCRATX)
```

```
     TO COMPILE THE PROCESSING PROGRAM OR THE FILE CREATION
     PROGRAM WHEN THE SOURCE IS ON DATA SET '&WORK'.  INPUT
     FOR FILE CREATION PROGRAM COMPILATION MUST INCLUDE ALL
     FILE CREATION PROGRAM SOURCE.  INPUT FOR COMPILATION
     OF THE BATCH MODE OF THE PROCESSING PROGRAM MUST INCLUDE
     ALL PROCESSING PROGRAM SOURCE AND DUMMIES FOR THE
     TEKTRONIX PLOT-10 ROUTINES 'ANMODE', 'MOVABS', AND
     'VCURSR'.  INPUT FOR COMPILATION OF THE INTERACTIVE MODE
     OF THE PROCESSING PROGRAM MUST INCLUDE ALL PROCESSING
     PROGRAM SOURCE AND THE SUBROUTINE 'PLOTS' WHICH IS A
     REPLACEMENT FOR THE TEKTRONIX 'CALCOMP PREVIEW' VERSION
     OF 'PLOTS'.  THE OBJECT DECK IS LEFT ON THE DATA SET
     'ESAR.TEMPOBJC.OBJ'.

//ESAR4  JOB (COMPIL00,G38,67061200,DP38,T,02),'DICK          2841',
//         MSGLEVEL=1,NOTIFY=ESAR,MSGCLASS=A,TIME=(5,00),CLASS=D
//FORTX  EXEC PGM=IFEAA8,REGION=256K
//SYSPRINT DD SYSOUT=$
//SYSTERM DD SYSOUT=$
//SYSUT1 DD UNIT=VIO,SPACE=(TRK,100)
//SYSUT2 DD UNIT=VIO,SPACE=(CYL,3)
//SYSLIN DD UNIT=SYSDA,DISP=(NEW,CATLG),DCB=BLKSIZE=3120,
//          SPACE=(CYL,10),DSN=ESAR.TEMPOBJC.OBJ
//SYSIN DD DSN=&WORK,DISP=(OLD,DELETE)
//


     TO LINK THE FILE CREATION PROGRAM ON TSO.  THE INPUT IS
     THE DATA SET 'ESAR.TEMPOBJC.OBJ AS CREATED ABOVE.  THE
     LOAD MODULE IS LEFT ON THE PARTITION 'FEAR02' OF THE
     LIBRARY 'ENGR.PROD1'.

LINK(TEMPOBJC.OBJ *) LOAD('ENGR.PROD1(FEAR02)') MAP FORTLIB LIB('ENGR.FO
RTLIB')
 ENTRY MAIN


     TO LINK THE BATCH VERSION OF THE PROCESSING PROGRAM ON
     TSO.  THE INPUT IS 'ESAR.TEMPOBJC.OBJ' FROM THE COMPIL-
     ATION JCL.  THE LOAD MODULE IS LEFT ON THE PARTITION
     'ATAR02' OF THE LIBRARY 'ENGR.PROD1'.

LINK(TEMPOBJC.OBJ *) LOAD('ENGR.PROD1(ATAR02)') MAP SIZE(290000,90000) F
ORTLIB LIB('ENGR.FORTLIB')
 ENTRY MAIN


     TO LINK THE INTERACTIVE VERSION OF THE PROCESSING PROGRAM
     ON TSO.  THE INPUT IS THE DATA SET 'ESAR.TEMPOBJC.OBJ'
     FROM THE COMPILATION JCL.  THE LOAD MODULE IS THE PARTITION
     'ATAR00' ON THE LIBRARY 'ENGTEST'.

ALLOC F(TEKLIB) DA('ENGR.TCSLOAD1')
LINK (TEMPOBJC.OBJ *) LOAD('ENGTEST(ATAR00)') MAP SIZE(290000,90000) FOR
TLIB LIB('ENGR.TCSLOAD1' 'ENGR.FORTLIB')
 INCLUDE TEKLIB(CALCOMP)
 ENTRY MAIN
```