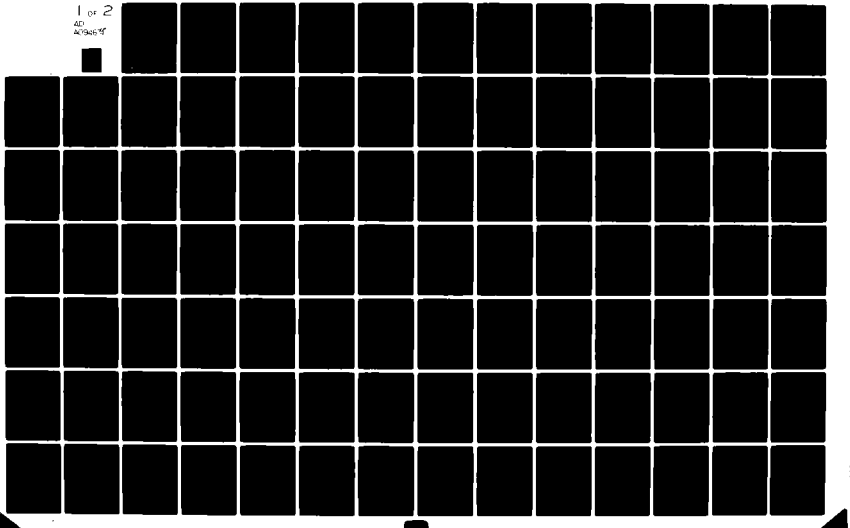
AD-A094 674 BELL HELICOPTER TEXTRON FORT WORTH TX F/G 9/2
THE DATA FROM AEROMECHANICS TEST AND ANALYTICS - MANAGEMENT AND--ETC(U)
DEC 80 R B PHILBRICK DAAK51-79-C-0015
UNCLASSIFIED BHT-699-099-025-VOL-2 USAAVRADCOM-TR-80-D-30B NL



USAAVRADCOM-TR-80-D-30B

AD A094674 LEVEL II

THE DATA FROM AEROMECHANICS TEST AND ANALYTICS — MANAGEMENT AND ANALYSIS PACKAGE (DATAMAP)
Volume II - Systems Manual

Richard B. Philbrick
BELL HELICOPTER TEXTRON
P. O. Box 482
Fort Worth, Tex. 76101

December 1980

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.

DATAMAP is a computer software system which provides direct access to large data bases, performs analysis and derivations, and provides the user with various options for output display, interactively or through batch processing. DATAMAP was designed to utilize the AH-1G Operational Loads Survey data but is general enough to accommodate other large, time-based, data sets resulting from test or analysis.

Improvements have been made to DATAMAP to enhance its graphics, analysis, and operational capabilities in order to expand its versatility and usefulness as an engineering tool. Volume I of this report explains the general structure and capabilities of the improved DATAMAP, and Volume II provides information on programming considerations.

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

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: USAAVRADCOM-TR-80-D-30B	2. GOVT ACCESSION NO.: AD-A094 624
	3. RECIPIENT'S CATALOG NUMBER:
4. TITLE (and Subtitle): THE DATA FROM AEROMECHANICS TEST AND ANALYTICS - MANAGEMENT AND ANALYSIS PACKAGE (DATAMAP) Volume II - Systems Manual	5. TYPE OF REPORT & PERIOD COVERED: Final Technical Report
	6. PERFORMING ORG. REPORT NUMBER: LHT-699-099-025-Vol-2
7. AUTHOR(s): Richard B. Philbrick	8. CONTRACT OR GRANT NUMBER(s): DAAK51-79-C-0015
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Bell Helicopter Textron, P. O. Box 482, Fort Worth, Texas 76101	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 612209 1L162209AH76 00 265 EK
11. CONTROLLING OFFICE NAME AND ADDRESS: Applied Technology Laboratory, US Army Research & Technology Laboratories (AVRADCOM), Fort Eustis, Virginia 23604	12. REPORT DATE: December 1980
	13. NUMBER OF PAGES: 187
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	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)

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, Acoustic Analysis.

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

The Data from Aeromechanics Test and Analytics - Management and Analysis Package (DATAMAP) was designed and programmed as a computer software tool for data management and processing of large, time-based data bases. Particular attention is given to rotorcraft-related analyses. The package will process data stored in two basic formats. The first format is that used for the Operational Loads Survey (OLS) test data base and anticipated for use in planned flight test programs. The second format is more

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)

general; it accommodates various data structures common to analytical data bases. This particular input capability is demonstrated by an interface between the Rotorcraft Flight Simulation Program (C81) and DATAMAP. The package transfers selected data to a large, direct access disc file and maintains the data on a semi-permanent 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, auto-spectral density, frequency response function, acoustic analyses, and blade static pressure and normal force coefficient derivations. 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.

The package is written entirely in FORTRAN. Package specifications require nonstandard FORTRAN coding to be used, but the package has been made as easily transportable as possible. In particular, DATAMAP is installed on a Digital Equipment Corporation VAX 11/780 as well as on the originally intended IBM 360 and 370 systems.

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.

1
B

PREFACE

DATAMAP Version 2.00 (formerly known as the Operational Loads Survey Data Management System) was developed under Contract DAAJ02-77-C-0053 for the Applied Technology Laboratory, U. S. Army Research and Technology Laboratories (AVRADCOM) to process the data measured by the AH-1G Helicopter Aerodynamic and Structural Loads Survey (OLS). This survey, performed for Applied Technology Laboratory (ATL), resulted in a comprehensive base of helicopter test data including measurements such as airfoil surface pressures, leading-edge stagnation point, local flow magnitude and direction, blade accelerations, bending moments, and the attendant responses in the control system and airframe. The data base included 72,000 separate digitized functions of time from 367 transducers. This data base, together with the techniques used for measurement of the data, is documented in Reference 1.

DATAMAP Version 2.00, documented in Reference 2, has been successfully used to process the OLS data base for a number of projects both at ATL (Fort Eustis, Virginia) and at Bell Helicopter Textron (BHT). To enhance the usefulness of this software, Contract DAAK51-79-C-0015 was awarded in May of 1979 by ATL. This contract required BHT to modify DATAMAP to incorporate new analysis, operational, and graphic capabilities, and also to provide an interface between the Rotorcraft Flight Simulation Program (C81) and DATAMAP. Documentation prepared under this contract for DATAMAP consists of two volumes. Volume I provides user instructions and information

---

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

[2]Richard B. Philbrick, and Alfred L. Eubanks, OPERATIONAL LOADS SURVEY - DATA MANAGEMENT SYSTEM, Bell Helicopter Textron, USARTL Technical Report 78-52A and 52B, Applied Technology Laboratory, Fort Eustis, Virginia, January 1979, AD A065129 and AD A065270.

[3]James R. Van Gaasbeek, and P. Y. Hsieh, ROTORCRAFT FLIGHT SIMULATION PROGRAM C81 WITH DATAMAP INTERFACE, Volumes I and II, Bell Helicopter Textron, USAAVRADCOM Technical Report 80-D-38A and 80-D-38B, Applied Technology Laboratory, U. S. Army Research and Technology Laboratories, Fort Eustis, Virginia,

on the analytical methods used in the software. Volume II, the Systems Manual, details the various programming considerations. Information on the C81 - DATAMAP interface is also contained in Reference 3, which was produced for this contract.

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

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Concluded)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Concluded)

LIST OF TABLES

1. INTRODUCTION

This volume documents the source code for the Data from Aeromechanics Test and Analytics - Management and Analysis Package (DATAMAP) and should assist the programmer/analyst in modification, maintenance, and installation of the package. 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 DATAMAP 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. One of the allowed input formats for the File Creation Program, the Data Transfer File (DTF) format, is described in Section 3. Section 4 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 5. Section 6 discusses installation of DATAMAP on non-IBM computers, while Section 7 lists the specific file and linking requirements for DATAMAP. 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, link-edit, and execute the code.

Two specific kinds of system modifications are documented with particular detail. Section 2.3 discusses modification of the File Creation Program to accept tape formats other than the BHT Ground Data Center (GDC) standard tape format or the DTF format. Interface requirements for replacement of a processing module are discussed in Paragraph 4.4.4. 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 4.3.2.

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 DATAMAP, an 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 6. 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 that 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 location of 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 users 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 the 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 the relative

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

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. There are two allowed formats for the information record and these formats are shown in Figures 7 and 8. For Bell Helicopter Textron - Ground Data Center (BHT-GDC) format data input, the information record format in Figure 7 is used. For DTF input, the format in Figure 8 is used instead. DATAMAP determines which format is used from the four-byte integer in entry six. The values zero and one indicate GDC format input (Figure 7), while the value two indicates DTF input (Figure 8). For either format, some of the values in the information record are required for processing the data stream and others in this record are only present for information pruposes. The first data record follows the information record sequentially and all data records for that item code/counter pair follow sequentially and contiguously.

The first four information record entries are the same for either format and are available for future use if a program is written to condense partitions that 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, is the same for either format. This entry 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.

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.

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. Information record for data of BHT-GDC format origin.

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	2=DTF CALIBRATED
7	8	FLIGHT DATE MM/DD/YY
8	4	ALIGNMENT OFFSET (SEC)
9	4	ADDITIONAL OFFSET (SEC)
10	4	DIGITAL FILTER CUTOFF
11	52	ITEM CODE DESCRIPTION
12	12	ITEM CODE UNITS
13	4	0
14	4	SAMPLE RATE (FLOATING)
15	8	MODEL NUMBER
16	8	SHIP NUMBER
17	8	GROSS WEIGHT
18	8	CG
19	8	FLIGHT NUMBER
20	8	MODEL CODE
21	4	DATA TYPE: 1=MIN/MAX, 2=HISTORY
22	12	TIME OF MANEUVER HH.MM.SS.TTT
23	4	CG CODE
24	4	0=UNSPECIFIED, 1=TEST, 2=ANALYTIC

Figure 8. Information record for data of DTF origin.

For the format in Figure 7, 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 22, the sample rate reduction factor. The data type, entry 25, indicates whether the data are time history or min/max data, although use of this package for min/ max input data is not currently planned. For the format in Figure 8, the sample rate is entry 14, which is stored as a four-byte floating word. The time history - min/max indicator is entry 21.

For the format in Figure 7, entries 9 and 10 indicate the time offsets in seconds applied to the data stream during 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 corresponding entry numbers for the format in Figure 8 are 8 and 9.

The other entries are present largely for information and display purposes and are all explained in Reference 4, except for the digital filter cutoff, which is entry 21 for the format in Figure 7 and entry 10 for the format in Figure 8. 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 that 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 9. 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 first record in the partition directory 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.

---

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

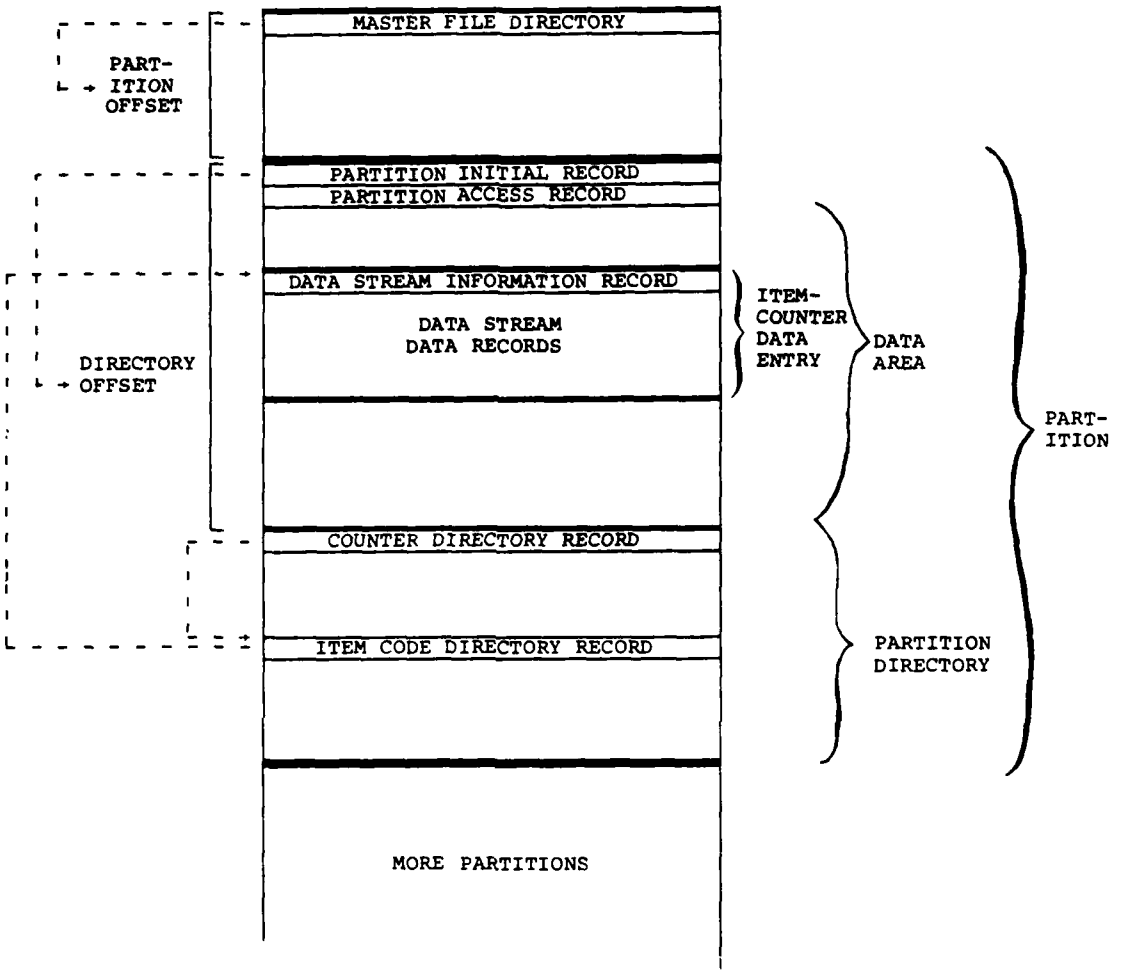


Figure 9. Master File structure.

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 stored or to the amount of data from an item code/counter data stream that 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 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.

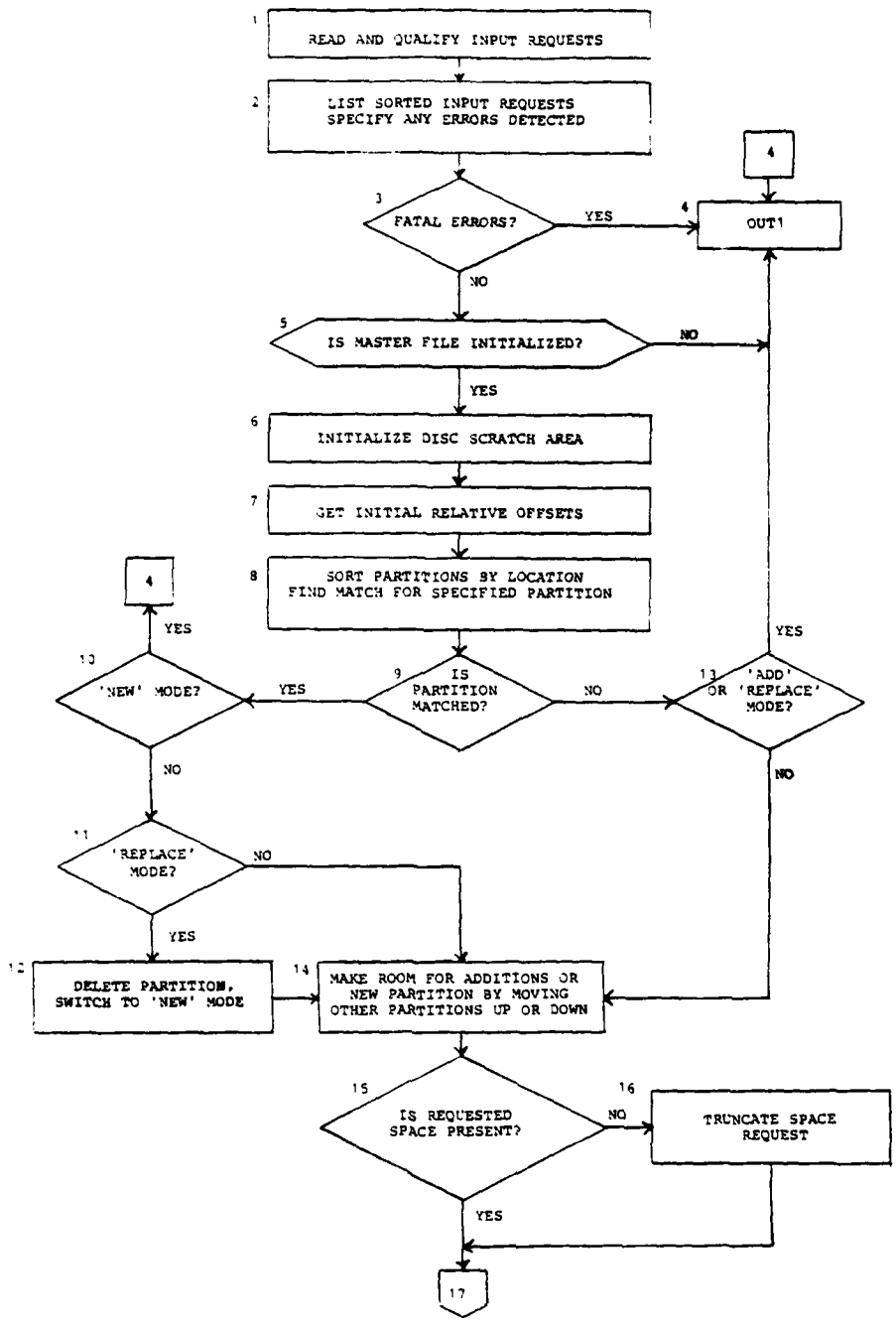


Figure 10. File creation program flow diagram (first part).

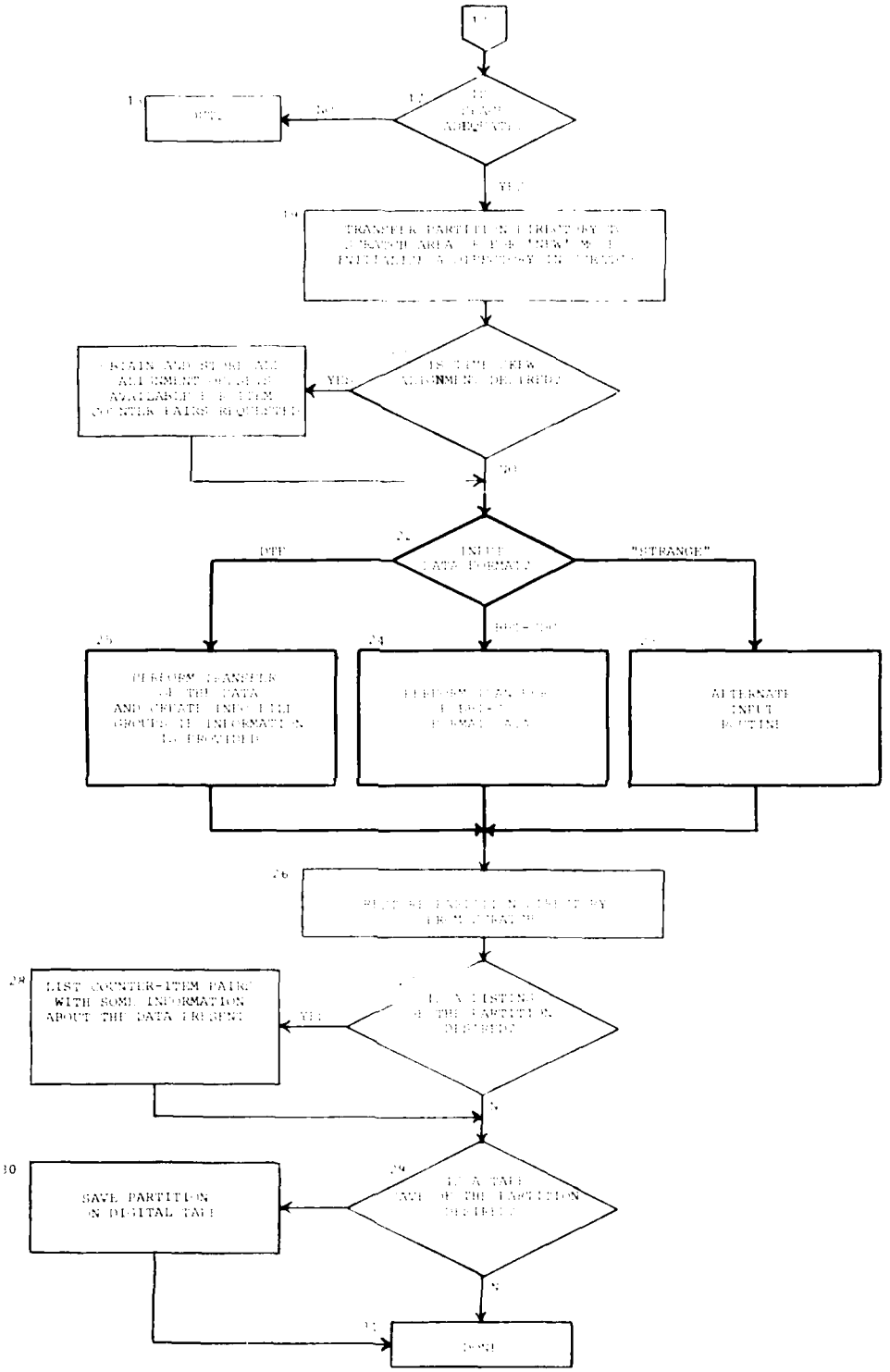


Figure 11. File creation program flow diagram (last part).

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.
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 tests the data input format that was specified by the instruction input. For a nonstandard input format, control passes to block 23. For the standard BHT-GDC format (e.g., OLS data), control is passed to block 24. For DTF format input, block 25 receives control.
Block 23 - MAIN calls subroutine STRNGF to process the nonstandard format data. A different version of this subroutine must be provided for each specific nonstandard format. Consult Section 2.3 for specifications for this subroutine. When processing is complete, control passes to block 26.
Block 24 - MAIN calls subroutine GDCFRM to process standard BHT-GDC format input tapes. A separate block diagram and description is provided for this subroutine. When processing is complete, control passes to block 26.
Block 25 - MAIN calls subroutine DTFFRM to process DTF format input. A separate block diagram and description is provided for this subroutine. When processing is complete, control passes to block 26.

Block 26 - 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 27 - 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 28 - 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 29 - 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 30 - MAIN calls SAVALL to save the partition on digital tape. SAVALL then returns to MAIN and the program goes to block 40.
Block 31 - Done.

At block 22 of the above flow sequence, the flow branches to three different routines based upon the type of input. For input that follows neither the BHT-GDC or the DTF format, a special routine must be written as described in Section 2.3. Following is a description of the GDC format input routine with reference to Figure 12. The DTF input routine will be described subsequently.

Block 1 In subroutine GDCFRM, an assignment record is read from the input data tape.
Loop 2 A loop is entered to process the data that follow each assignment record. The loop assumes that a new assignment record has been read at the start of the loop. This loop will be called loop 2.
Block 3 Subroutine FITEM is called to determine whether the assignment record specifies any item codes that are required for transfer. If none are found, then control is transferred to block 14. If one or more required item codes are found, loop 4 is entered.
Loop 4 A loop is entered to process the several counters that follow on the data tape. For each counter, the data for all item codes specified in the assignment record will appear in parallel.
Block 5 A record is read from the input data tape.
Block 6 The type of record found is tested. If an assignment record is found, control branches immediately to the bottom of loop 2. For a data record, control branches to block 7. If a calibration record or end of data were found, a tape

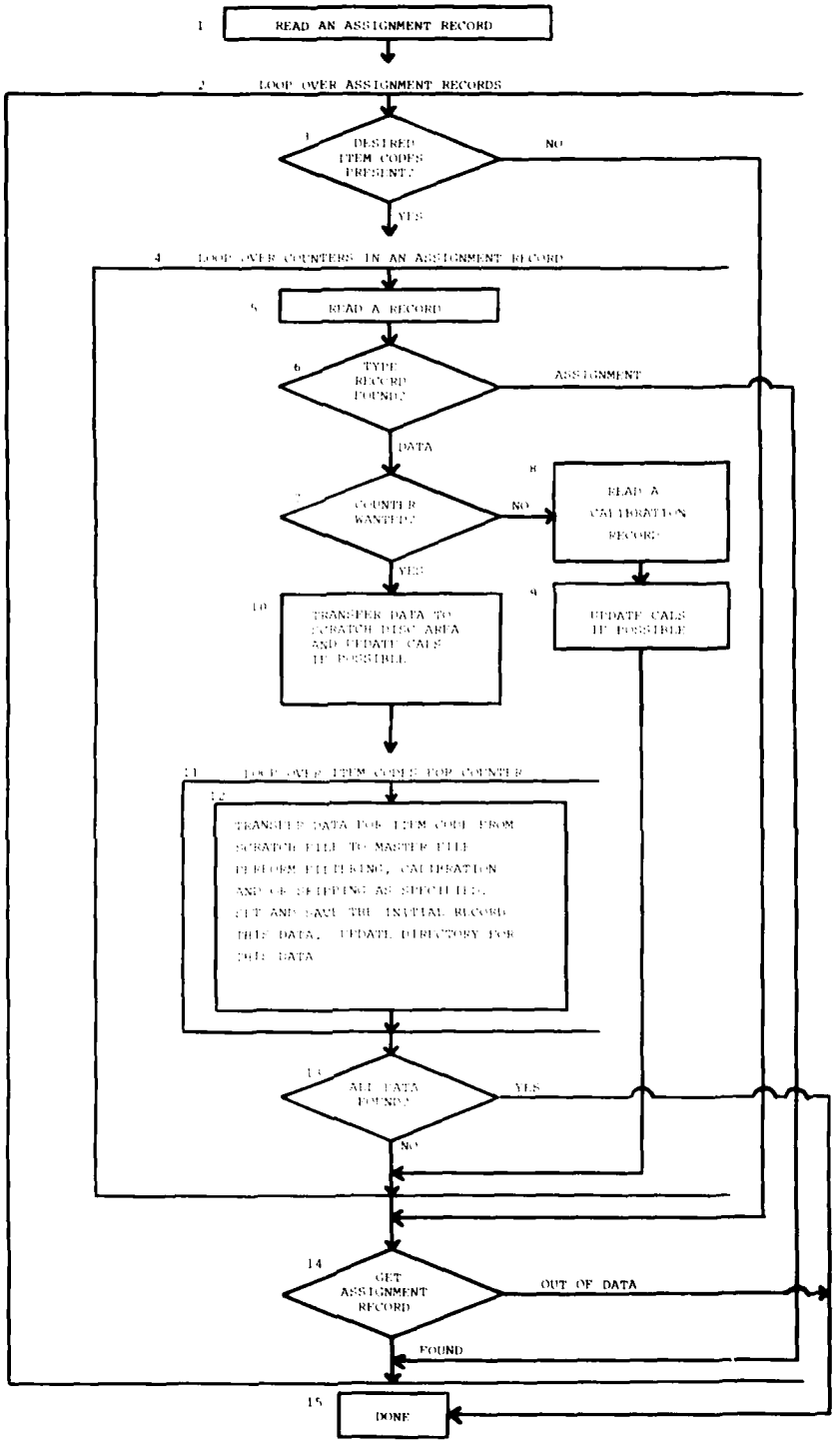


Figure 12. GDC format input subroutine block diagram.

format error would have occurred. Branches to error message codes are included at this point in the program for these occurances but the paths are not shown to simplify the diagram.
Block 7 Subroutine FCNTR is called to test whether the counter for the data record that was found is required for transfer. If not, control branches to block 8. If the counter is required, the flow continues to block 10.
Block 8 The data tape is read until a calibration record is found.
Block 9 Calibration factors are updated if possible by subroutine CALUPD.
Block 10 Data from the tape corresponding to the counter and to all item codes listed in the assignment record are transferred to a sequential scratch file by subroutine TRANSC. Calibration factors are updated if possible by subroutine CALUPD.
Loop 11 A loop is entered to process each item code listed in the assignment record that is required for transfer.
Block 12 Data for the current item code/counter pair are transferred from the sequential scratch file to the Master File. Subroutine SAVD is used if no digital filtering is required. Calibration and/or sample rate reduction are accomplished by either of these subroutines as required.
Block 13 A test is performed to determine whether all required data have been found. If so, control is transferred to block 15. If not, loop 4 is reexecuted.
Block 14 The input data tape is read until an assignment record is read or the end of the input data is encountered. For the end of data, control is transferred to block 15. If an assignment record is found, loop 2 is reexecuted.
Block 15 Subroutine GDCFRM is complete.

Following is a description of the DTF format input routine with reference to Figure 13. Reference to Section 3 may be required to follow the DTF record types.

Loop 1 Subroutine DTFFRM begins by entering a loop over individual DTF records or, for certain cases, sequences of DTF records of the same type.
Block 2 Inside Loop 1, the routine tests whether an unprocessed DTF record is available in program memory. If so, control is passed to block 5. If not, the flow branches to block 3.
Block 3 Subroutine INPDTF is called to attempt to read a DTF record.
Block 4 Subroutine DTFFRM tests the result of the read attempt in block 3. If an error was detected, control is passed to block 10. If an end-of-file was encountered, control is passed to block 11. Otherwise, the normal flow proceeds to block 5.

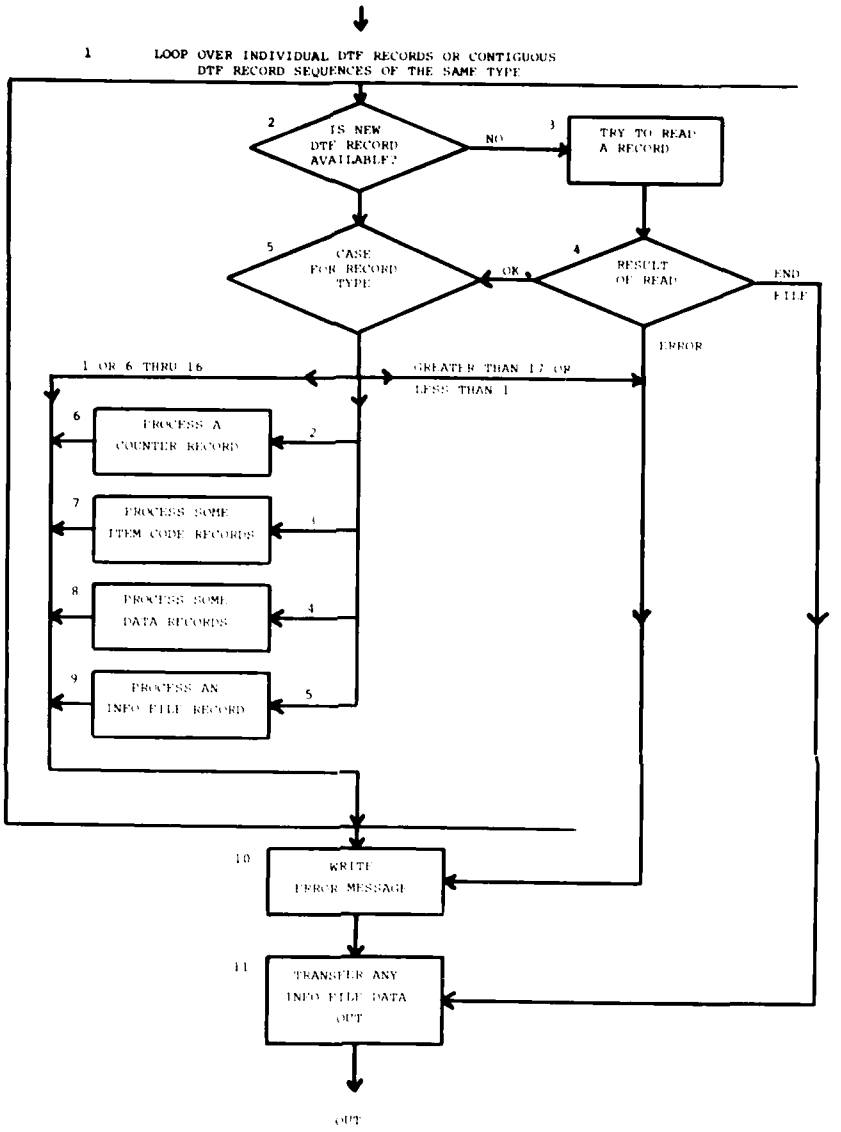


Figure 13. DTF data format input subroutine block diagram.

Block 5 The record type field is tested and based upon the value of the integer in this field, control is passed to one of several blocks. If the integer is out-of-range (i.e., less than 1 or greater than 16) control is passed to block 10. Values of six through sixteen require no action and control is passed to the bottom of loop 1. A value of one requires no action except to test that no other type of record has been read previously. Control is then passed to the bottom of loop 1. For values of 2, 3, 4, or 5, control is passed to blocks 6, 7, 8, or 9 respectively.
Block 6 A counter record is processed for counter and certain other information that applies to all channels (item codes) in the data that follow. A time base specification may be processed.
Block 7 One or more item code records are processed for item codes, item code sequence in the subsequent data, and possibly for Info File information. A time base specification may be processed.
Block 8 One or more data records are processed. The data are transferred to a scratch disc file and then for each channel the necessary filtering and/or interpolation processing is performed and the data are stored in the Master File.
Block 9 An Info File record is processed. Each record contains the basic structure and labels for one or more Info File groups.

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.

Generally, it is more desirable to write a program to convert a data base to DTF format than to write a STRNGF subroutine and incorporate this routine in the FCP. This is true for two reasons. First, the DTF format is specifically organized to accommodate certain data base structures such as parallel data and variable sample rates. A STRNGF subroutine would have to include code to convert these structures, if necessary. Second, the STRNGF subroutine must reference COMMON values from the FCP. Thus, the STRNGF routine may require rewriting for each revision of the FCP.

Table 1 lists a prototype version of STRNGF showing the sequence that 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 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 999999.

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.

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 DATAMAP
C     FILE CREATION PROGRAM.
C
      DIMENSION INREC(256)
      COMMON/LOCOM/ITEMN,ICNTRN,IDROFF,
     $  IDASIZ,IDRSIZ,ITEMRC,ITEMSO
      LOGICAL LCAL
      COMMON/INFO/IRSIZ,MLOC,LOCO,IPOLES,HIGH,LCAL,IRAT,
     $ ISKIP,NPS,NPP,NOFF,ISEQ,LSTRT,IADD,INSIZ,INSIZD,
     $ IRDATS,IRSAVS,ICNTR,XALIGN,MSETN
      COMMON/KARD/ITEMTP(510),ITEMW(510),CALSH(26),
     $ CALCM(26),CXM(26),CXB(26),NMATCH,DCAL(26)
      LOGICAL LALIN,MAPIT,SAVIT,STRANG,LDTF,LDTFIN,
     1 LTHERE,LEXTRN,LALLIT,LALLCN,LSCAN
      COMMON/LIST/ NITEMS,NCNTRS,ISPAC,ITAPES,IADNU,LALIN,
     1 NAME(2),NPWD(4),NUSER(4),MAPIT,SAVIT,STRANG
     2 ,LDTF,NLOOK,LDTFIN,LTHERE,LEXTRN,LALLIT,LALLCN,LSCAN
     3 ,NCTR(100),NOFFST(100),NPWANT(100),
     4 ITEM(510),FILT(510),ICAL(510),ISKP(510),RATE(510)
      COMMON/FILES/NRPS,NRSC,NSSC,NITT,NDIR,NREA,NWRI,
     1 NSAV,IALI,NIFO
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.
C
      DO 500 I = 1, 10000
C
C       DETECT THE PRESENCE OF A SUBSET OF THE REQUESTED DATA
C       CORRESPONDING TO ONE COUNTER AND ONE OR MORE ITEM CODES.
C       ASSIGN A NUMBER BETWEEN ONE AND THE DIMENSION OF THE ARRAY
C       'ITEMW' TO EACH ITEM CODE IN THE SUBSET (ADJUST THE DIMENSIONS
C       OF THE ARRAYS IN THE COMMON BLOCK /KARD/ AS NECESSARY).  SET
C       'ITEMW(N)' FOR EACH ITEM CODE NUMBER, 'N', TO THE CORRESPONDING
C       ARRAY POSITION OF THE ITEM CODE IN THE ARRAY 'ITEM'.  IF DATA
C       ARE TO BE STORED IN INTEGER FORM, SET THE CORRESPONDING 'CXM'
C       AND 'CXB' ARRAY VALUES FOR CALIBRATION ON RETRIEVAL FROM THE
C       MASTER FILE.  SET 'NMATCH' TO THE NUMBER OF ITEM CODES IN THE
C       SUBSET.  INSERT CODE AS NECESSARY TO PERFORM THESE FUNCTIONS.
C
C       LOOP OVER ITEMS IN THE SUBSET
C
        DO 400 J = 1, NMATCH
C
C         SET THE VARIABLES 'ICNTRN' AND 'ITEMN' TO THE COUNTER AND
C         ITEM CODE RESPECTIVELY.  SET THE ARRAY 'INREC' WITH SOME OF
C         REQUIRED VALUES FOR THE CORRESPONDING INFORMATION RECORD.
C         ..INREC(1) = ITEM CODE = ITEMN
C         ..INREC(6) = 1 IF CALIBRATED DATA ARE TO BE STORED,
C                    = 0 IF INTEGER DATA ARE TO BE STORED
C         ..INREC(12-20) = ITEM CODE DESCRIPTION/UNITS WITH UNITS IN
C                          THE LAST SIX BYTES
C         ..INREC(27) = DIGITAL FILTER CUTOFF, -1.0 IF NO FILTER
C                       APPLIED
C         ..INREC(28) = SAMPLE RATE REDUCTION FACTOR,
C                     = ISKP(ITEMW(J))
C         ..INREC(29) = SAMPLE RATE OF DATA ON TAPE BEFORE THE SAMPLE
C                       RATE REDUCTION FACTOR IS APPLIED
C         ..INREC(42) = 2 (TIME HISTORY DATA, NOT MIN/MAX)
```

TABLE 1. PROTOTYPE EXAMPLE FOR SUBROUTINE STRNGF (Continued)

```
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 FOR THE
C             CURRENT COUNTER. I.E., SET MCNTR SO THAT
C             .......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 AN ITEM CODE
C
                ISKIP = 1
                MLOC = IDASIZ + 2
                DO 300 K = 1, 10000
C
C                 READ THE DATA FOR THE NEXT RECORD.  IF OUT OF DATA,
C                 BRANCH TO 350.  GET DATA (256 CALIBRATED OR 512 INTEGER
C                 VALUES) INTO THE ARRAY 'INREC'.  SET NUM TO NUMBER OF
C                 POINTS IN THE RECORD.
C
                    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             THAT WAS JUST WRITTEN ON THE PARTITION.
C
  350           CALL INIDAT(J,MCNTR,NERR,INFO,IERR)
                IF(NERR .NE. 0)GO TO 560
                IDASIZ = MLOC - 1
                IF(IEND .NE. 0)GO TO 570
  400       CONTINUE
  500   CONTINUE
        GO TO 1000
C
C    DIRECT ACCESS WRITE ERROR STORING THE INFORMATION FILE
C    ON SCRATCH DIRECT ACCESS DISC.
C
  550   WRITE(NWRI,9000)IERR
        GO TO 1000
C
C    ERROR ADDING INFORMATION RECORD TO THE PARTITION.
C
  560   WRITE(NWRI,9001)NERR,INFO,IERR
        GO TO 1000
C
C    OUT OF SPACE ON THE PARTITION
C
  570   WRITE(NWRI,9002)IERR
        GO TO 1000
C
C    ERROR ANNOTATING DIRECTORY FOR START OF DATA STREAM
C
  580   WRITE(NWRI,9003)NERR,INFO,IERR
C
C
 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 ,
     1 3I10//)
 9002 FORMAT(3X,36H***RAN OUT OF SPACE ON THE PARITION //)
 9003 FORMAT(3X,26H***ERROR SETTING DIRECTORY //)
      END
```

TABLE 1. PROTOTYPE EXAMPLE FOR SUBROUTINE STRNGF (Concluded)

```
      SUBROUTINE ADDAT(IDAT,NUM,ICHK)
C
C   ROUTINE FOR USE BY ROUTINE 'STRNGF' TO WRITE DATA TO THE
C   MASTER FILE.
C
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
      COMMON/INFO/IRSIZ,MLOC,LOCO,IPOLES,HIGH,LCAL,IRAT,
     $ ISKIP,NPS,NPP,NOFF,ISEQ,LSTRT,IADD,INSIZ,INSIZO,
     $ IRDATS,IRSAVS,ICNTR,XALIGN,MSETN
      COMMON/LOCOM/ITEMN,ICNTRN,IDROFF,
     $  IDASIZ,IDRSIZ,ITEMRC,ITEMSQ
C
C
      ICHK = 0
      IF(NUM .LE. 0)GO TO 1000
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-2)*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
C
 500  ICHK = 1
      GO TO 1000
C
C   OUT OF DATA SPACE IN THE PARTITION
C
 510  ICHK = 2
C
C
1000  RETURN
      END
```

3. STRUCTURE AND FORMAT OF THE DATA TRANSFER FILE

This section describes the required structure and content of a DTF. That is, various rules are presented for the structure and ordering of DTF records, and the proper content of these records is specified. The program that creates a DTF has the responsibility to ensure that the structure and content of the DTF is correct. The File Creation Program (FCP) tests the structure of the DTF for correctness and also tests certain parts of the data content of the DTF for consistency. However, the FCP cannot test the overall content of the DTF for accuracy. When an error is detected in the DTF, the FCP stops processing additional DTF data, stores any data that were correctly processed before the error was detected, and terminates the program run.

3.1 DATA TRANSFER FILE FEATURES

3.1.1 Format Types

There are two alternative DTF formats, internal and external. The format selected must be consistent throughout any one DTF. Internal format is used to transfer data between jobs running on the same computer or on different computers with the same word size, integer and floating word formats, and alphanumeric character representation. Internal format is written and read using direct transfer of computer words without FORTRAN format conversion and uses binary integer and floating data formats and the local character representation for alphanumeric data assuming four left-justified characters per word (the term "left-justified" is used here to indicate the first four character positions in a word).

External format is used to transfer data between jobs running on different computers. The records are written and read using FORTRAN format conversion. External format is less efficient than internal format but allows transfer of data between computers with different internal data representations.

3.1.2 Physical File Characteristics

There is a constant logical record length for any one DTF although this length may change between internal and external format DTF's and between different computers. An internal record always contains 1024 words, while an external record always contains 4096 characters. Thus, certain types of record have different information capacities for internal and external formats. Whenever this section refers to a DTF record, the information transferred by a single FORTRAN "READ" statement is assumed. For external format records, the "READ"

is formatted and requires transfer of 4096 characters. Internal format "READ's" are unformatted and require transfer of 1024 words. The physical block structure of a DTF may be different for different computers and installations as long as the above requirements are satisfied. When a DTF is used to transfer data between different computers, special block structures may be required to satisfy the system requirements for both computers. It is the local installation's responsibility to assure that the data storage structure on tape or disc is appropriate so that the correct data are transferred for each READ.

The DTF is organized sequentially rather than using direct access. The FCP reads the DTF using one pass through the file without rewinding or backspacing.

3.1.3 Record Types

There are five different types of logical record identified by a numeric label located at the start of each record.

1 = Instruction. An instruction record contains directions for the FCP to follow in transferring the DTF data to the Master File.

2 = Counter. A counter record contains identifying counter and other associated information for data that follow (a counter is an integer between 1 and 999999 that is assigned to the data from a flight test maneuver or a particular simulation of a maneuver by an analytic program).

3 = Item Code. An item code record contains item codes, item code descriptions and units, and associated information for data that follow (an item code is a four-character label attached to a sensor or transducer for test data or to some time-dependent function output by a simulation program).

4 = Data. A data record contains data values.

5 = Info File. An Info File record contains basic information for Info File groups.

There are also eleven types of record that are ignored by the FCP. This feature allows for later expansion of the DTF format and allows the format to be used for purposes other than processing by DATAMAP. These types of record are labeled six through sixteen.

3.1.4 Data Structure

Data may be written in parallel. That is, values from more than one channel for a specific time instant may be written contiguously on the file. Data may also be written in series as a special case of parallel. A bundle is defined as a set of one or more data values from a like number of data channels that all correspond to one instant of time and that are written consecutively on the DTF.

3.1.5 Sample Rates

Data must be time based but the sample rate may be variable. If the sample rate is variable, each bundle must contain the time instant for the bundle as a data value in the bundle corresponding to a prespecified item code.

3.1.6 Info File Group Information

The DTF format will accomodate the information necessary to write DATAMAP Info File groups. The FCP has the capability to extract this information and generate a file of valid Info File groups. This file must be concatenated with an Info File base data set that contains the initial group for the Info File and other groups if necessary. This concatenated pair of data sets would form the Info File for the DATAMAP Processing Program.

3.2 DTF RECORD FORMATS

3.2.1 Specific Data Representation

Internal format records are treated as sequences of words with only two specific assumptions about the internal word size or structure in the host computer. First, integer words must be the same size (i.e., number of bits) as floating words. Second, a word must hold at least four characters in the internal character representation used by the computer. The DATAMAP source code makes the same assumptions so that no new restriction is imposed on the applicability of DATAMAP. Information is stored as binary integer or floating numbers, or as character strings with four left-justified characters allocated to each word. Unused word positions must be occupied in the file even though the contents of the words are ignored.

External format data are treated as a sequence of characters. The assumption is made that the computer system will be able to translate the character representation on the DTF (e.g., ASCII) to the internal character representation of the computer running DATAMAP (e.g., EBCDIC). A preprocessing step

may be required to translate character representations such as from ASCII to EBCDIC. The capability is not specifically provided in DATAMAP. Information is coded using the 'I', 'A', and 'E' format specifications. Unused character positions must always be occupied, although the contents of a position may be ignored. Thus, a DTF external format record must always contain the full 4096 characters.

3.2.2 Record Type Label

The first data field for every kind of record is the type label. This label contains the number corresponding to the type of record as specified in Paragraph 3.1.3. For internal format records, this label is an integer word. For external format records, this label is an integer coded in "I4" format.

3.2.3 Instruction Records

Instruction records (type number = 1) allow the control input for the FCP to be carried inside the DTF. An instruction record, as depicted in Figure 14, can contain as many as 56 lines of instructions in either the internal or external formats. The second field of the instruction record is an integer indicating the number of lines actually stored in the record. If more than 56 program instruction lines are required, more instruction records can be written. A line of instructions is 72 characters long, although the FCP requires that the entries on the line may not extend past character position 60. Positions 61 through 72 should contain blanks.

3.2.4 Counter Records

The primary purpose of a counter record is to associate a counter with a set of data. As indicated in Figure 15, time base information may optionally be included in a counter record. Four of the other quantities, Gross Weight, Center of Gravity, Model Number, and Ship Number, are used only for labeling purposes. If they are left blank or zero, the operation of DATAMAP will not be impaired. The other fields are stored on the Master File but are not used for any other purpose.

Figure 16 shows the sub-fields of the time base field in the counter records; the same time base sub-field organization is used in the item code records. There are four entries in this field. The second entry is reserved for the start time of data but is currently ignored and assumed to be zero. Thus, the time base is specified by the first, third, and fourth entries. The first entry is an integer between zero and three that indicates both the method of time base specification and

Field Contents	Internal: Word Numbers	Internal: Format	External: Character Positions	External: Format
• Record type label = 1	1	Binary Integer	1-4	I4
• Number of instruction lines in record	2	Binary Integer	5-8	I4
• Unused	3-16	/	9-64	/ (56X)
• Instruction line 1	17-34	18A4	65-136	18A4
• Instruction line 2	35-52	18A4	137-208	18A4
• Instruction line 3	53-70	18A4	209-280	18A4
		• • • •		• • • •
• Instruction line 56	1007 - 1024	18A4	4025 - 4096	18A4

Figure 14. DTF instruction record format.

Field Contents	Internal Word Numbers	Internal Field Format	External Character Positions	External Field Format
• Record type label = 2	1	Binary Integer	1-4	I4
• Time base information	2-5	See Fig. 16	5-36	See Fig. 16
• Unused	6-21		37-200	
• Counter	22	Binary Integer	201-208	I8
• Flight Number	23-24	2A4	209-216	2A4
• Gross Weight	25	Binary Floating	217-228	E12.6
• Center of Gravity	26	Binary Floating	229-240	E12.6
• CG Code	27	A4	241-244	A4
• Model Number	28-29	2A4	245-252	2A4
• Ship Number	30-31	2A4	253-260	2A4
• Date	32-33	2A4	261-268	2A4
• Time	34-36	3A4	269-280	3A4
• Model Code	37-38	2A4	281-288	2A4
• Test/Analytic Indicator 0=unspec,1=test,2=anal	39	Binary Integer	289-292	I4
• Unused	40-1024		293-4096	

Figure 15. Counter record format.

Field Contents	Internal		External	
	Word Numbers In Field	Sub-Field Format	Character Positions In Field	Sub-Field Format
• Time base type (see text)	1	Binary Integer	1-4	I4
• Start time (currently ignored)	2	Binary Floating	5-16	E12.6
• Time increment or sample rate (in seconds or samples per second)	3	Binary Floating	17-28	E12.6
• Item code (currently must be "TIME")	4	A4	29-32	A4

Figure 16. Time base field format.

the meaning of the third and fourth entries. Following is a description of the time base specification for each first entry value.

0... The time base is not specified in this record.

1... The time base is specified with a constant sample interval. This interval is the third entry (in seconds). The fourth entry is ignored.

2... The time base is specified with a constant sample rate. This rate is the third entry (in samples per second). The fourth entry is ignored.

3... The time base has a variable sample rate and is specified with a time instant for each bundle. The time instants are contained in the bundles as data values corresponding to the item code named in entry four. Currently, this item code name must be "TIME". The third entry is ignored.

3.2.5 Item Code Records

Item code records serve to associate item codes, item code descriptions, unit labels, and Info File information with the subsequent data records. As shown in Figure 17, the fourth field in the item code record is an integer indicating the number of item codes listed in the record. Information for as many as 32 item codes may be contained in one item code record. Several contiguous item code records can be used to identify a greater number of item codes. Each item code corresponds by position in the list with the data values having the same position in the subsequent bundles. For example, with two contiguous item code records, the tenth item code entry in the second record corresponds to the 42nd data value in each of the subsequent bundles.

Time base information may be included in an item code record. This field has the same format as shown in Figure 16.

Figure 18 is a breakdown of the information listed for each item code. The item code itself must always be present. An item code must be four characters in length and may not include a space, comma, slash, or single quote. The item code description and the units label are used by the Processing Program in displays so that if they were left blank the corresponding display fields would be left blank as well. Most of the balance of the information is for an Info File if the item code is to be associated with an Info File group. Refer to the description of the Info File format in Volume I for a

Field Contents	Internal Word Numbers	Internal Field Format	External Character Positions	External Field Format
• Record type label = 3	1	Binary Integer	1-4	I4
• Time base information	2-5	See Fig. 16	5-36	See Fig. 16
• Unused	6-31		37-124	
• Number of items in this record	32	Binary Integer	125-128	I4
• Information for item code number 1. See Figure 18.	33-63	See Fig. 18	129-252	See Fig. 18
• Information for item code number 2. See Figure 18.	64-94	See Fig. 18	253-376	See Fig. 18
•		•		•
•		•		•
•		•		•
•		•		•
•		•		•
• Information for item code number 32. See Figure 18.	994-1024	See Fig. 18	3973-4096	See Fig. 13

Figure 17. Item code record format.

Sub-Field Contents	Internal: Word Numbers In Field	Internal: Sub-Field Format	External: Character Positions In Field	External: Sub-Field Format
• Item Code	1	A4	1-4	A4
• Item Code Description	2-14	13A4	5-56	13A4
• Item Code Units	15-17	3A4	57-68	3A4
• Info File Group Affiliation	18	A4	69-72	A4
• Info File Column Number	19	Binary Integer	73-76	I4
• Info File Row Number	20	Binary Integer	77-80	I4
• Info File Doublerow 0=top, 1=bottom	21	Binary Integer	81-84	I4
• Info File Minor Position	22	Binary Floating	85-96	E12.6
• Modulo Value	23	Binary Floating	97-108	E12.6
• Unused	24-31		109-124	

Figure 18. Item code information field format.

description of the fields. The last sub-field contains the modulus value for modulo data. In particular, azimuth in degrees should be modulo 360. For non-modulo data, this value must be set to zero. The modulus value must not be less than zero.

3.2.6 Data Records

Figure 19 shows the structure of a data record. The second entry in the record is an integer that gives the number of data values contained. Notice that an internal format record can contain as many as 1020 data values while an external format data record can contain no more than 340 data values because 12 characters are used for each value. Thus, more records are required to store data in external format than are required for internal format. Section 3.3 will discuss the storage of bundles of data values in data records.

Modulo data must be between zero and the modulus for the channel (inclusive). Modulo data are assumed to represent a piecewise continuous function that is monotonically increasing except where the modulus limit is crossed, whereupon the function begins at zero again. The slope of the function before and after the modulus crossing is assumed to approach the same limit as the crossing is approached. Accordingly, when a datum value for a modulo channel is less than the previous datum value for the channel, a break or modulus crossing is indicated. When a break occurs, interpolation is performed by adding the modulus to the data after the break, interpolating, and then subtracting the modulus from any interpolated value that is greater than the modulus. Two points in succession must not indicate breaks (i.e., the input data values must not decrease for two points in succession).

5
F

3.2.7 Info File Records

An Info File record contains the basic information for generation of one or more Info File groups. As shown in Figure 20, basic Info File group information sets for as many as six groups may be contained in an internal format Info File record. An external format record can contain information for no more than three groups. Figure 21 shows the contents of a set of information for one group. Refer to Volume I, Section 5 for a description of the various fields. The row positions and labels are not required for one-dimensional groups.

Notice that Info File records do not contain all of the information required for production of Info File groups. Item code records contain the item code names and point to the appropriate group name and row/column/double-row element position within the group.

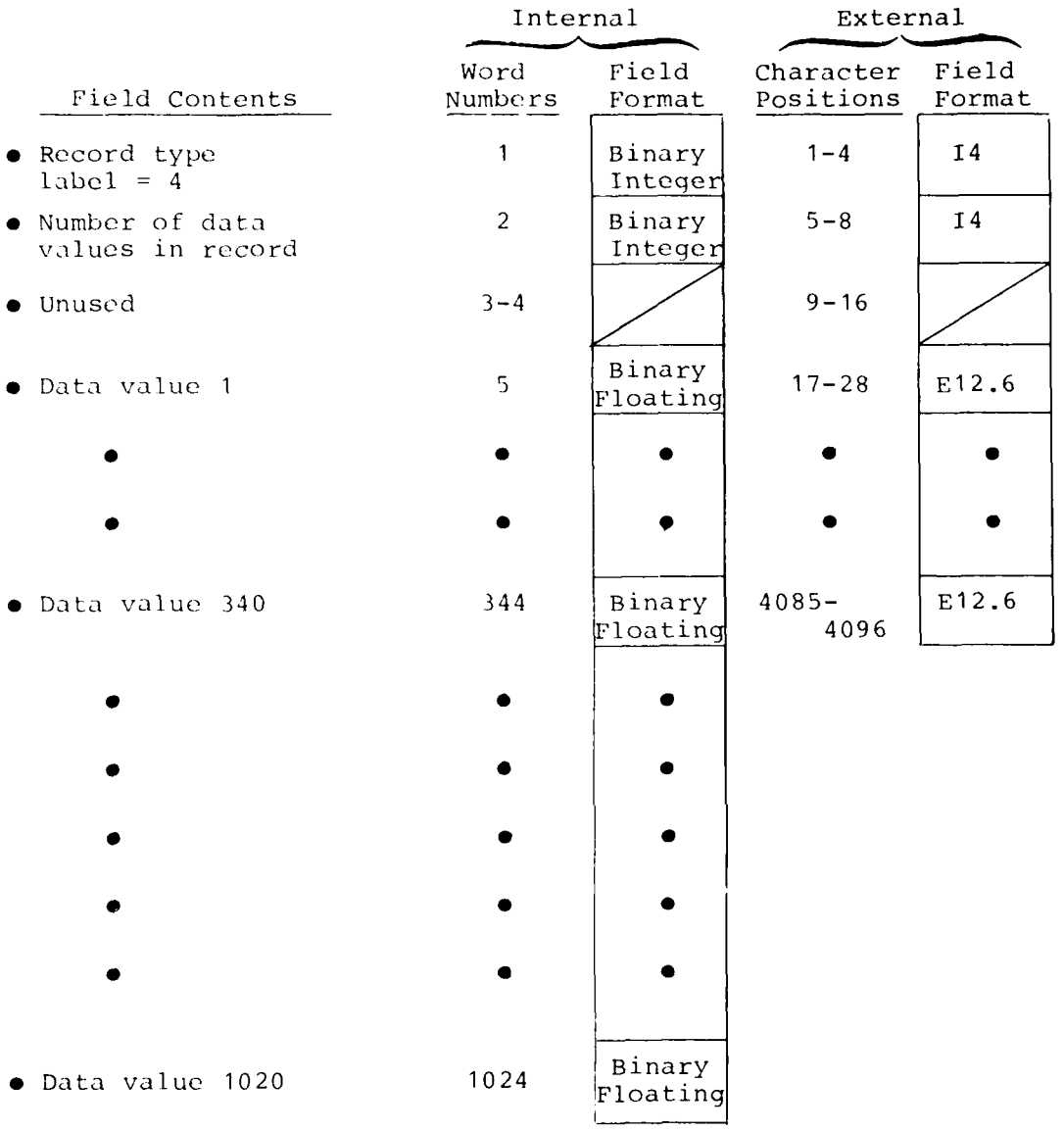


Figure 19. Data record format.

Field Contents	Internal Word Numbers	Internal Format	External Character Positions	External Format
• Record type label = 5	1	Binary Integer	1-4	I4
• Number of groups represented in record	2	Binary Integer	5-8	I4
• Unused	3-4		9-52	
• First Info File Group Information See Figure 21	5-174	See Fig. 21	53-1400	See Fig. 21
• Second Info File Group Information See Figure 21	175-344	See Fig. 21	1401-2748	See Fig. 21
• Third Info File Group Information See Figure 21	345-514	See Fig. 21	2749-4096	See Fig. 21
• Fourth Info File Group Information See Figure 21	515-684	See Fig. 21		
• Fifth Info File Group Information See Figure 21	685-854	See Fig. 21		
• Sixth Info File Group Information See Figure 21	855-1024	See Fig. 21		

Figure 20. Info file record format.

Sub-Field Contents	Internal		External	
	Word Numbers In Field	Sub-Field Format	Character Position In Field	Sub-Field Format
• Group Name	1	A4	1-4	A4
• Group Title	2-11	10A4	5-44	10A4
• Column Title (current 16 characters max.)	12-17	6A4	45-68	6A4
• Short Column Title/units (current 8 characters max)	18-20	3A4	69-80	3A4
• Feature near smallest column position (current max 16)	21-26	6A4	81-104	6A4
• Number of Column Positions (current max = 18)	27	Binary Integer	105-108	I4
• Column Positions	28-59	Binary Floating	109-492	32E12.6
• Row Title (current 16 characters max.)	60-65	6A4	493-516	6A4
• Short Row Title/Units (current 8 characters max.)	66-68	3A4	517-528	3A4
• Feature near smallest rows position (current max. 16)	69-74	6A4	529-552	6A4
• Number of Row Positions (current max = 18	75	Binary Integer	553-556	I4
• Row Positions	76-107	Binary Floating	557-940	32E12.6
• Doublerow labels. Top then bottom	108-117	5A4, 5A4	941-980	5A4, 5A4
• Keywords, Top then bottom	118-119	2A4	981-988	2A4
• Azimuth correction angle	120	Binary Floating	989-1000	E12.6
• Unused	121-170		1001-1348	

Figure 21. Info file record group-information field format.

3.3 DTF RECORD SEQUENCE

A prescribed order for the different types of records must be followed so that the different types of information are available as required. This sequence has been made as flexible as possible to accommodate the idiosyncrasies of various data bases.

3.3.1 Instruction Records

If instruction records are included in a DTF, they must form a contiguous group at the beginning of the DTF. No other type of record may appear before the last instruction record. The order of the instruction records must maintain the proper sequence of the control input lines. Thus, the first line of the first instruction record must contain the intended first instruction line, and the last active line of the last instruction record must contain an 'END' entry.

Instruction records are not required in a DTF. The FCP is able to read instructions from the system input file. In fact, the FCP must always read one line of instructions from the system input file, and the instructions on this line may indicate that data input is from a DTF and that control input is also from the DTF (see Volume I).

3.3.2 Data Records

A sequence of bundles of data values that correspond to a strictly monotonically increasing sequence of time instants and that are written contiguously on a DTF is called a bundle sequence. A sequence of data records containing a bundle sequence may not include any interspersed records of another type, any data records containing data values outside the bundle sequence, or any data records containing no data values. The number of data values for each bundle in a bundle sequence must be a constant and must equal the number of item codes specified for the bundle sequence.

As shown in Figure 22, a bundle may span more than one data record, or one or more bundles may be stored in each data record. A bundle may not span more than one data record if it does not start at the beginning of a data record. Thus, unused space will frequently be present in a data record. All unused space in a data record must be at the end of the record. If there is sufficient unused space at the end of a data record to contain a bundle, that record must be the last for the bundle sequence. The end of the bundle sequence is denoted by a record of some type other than a data record or by the end of the DTF.

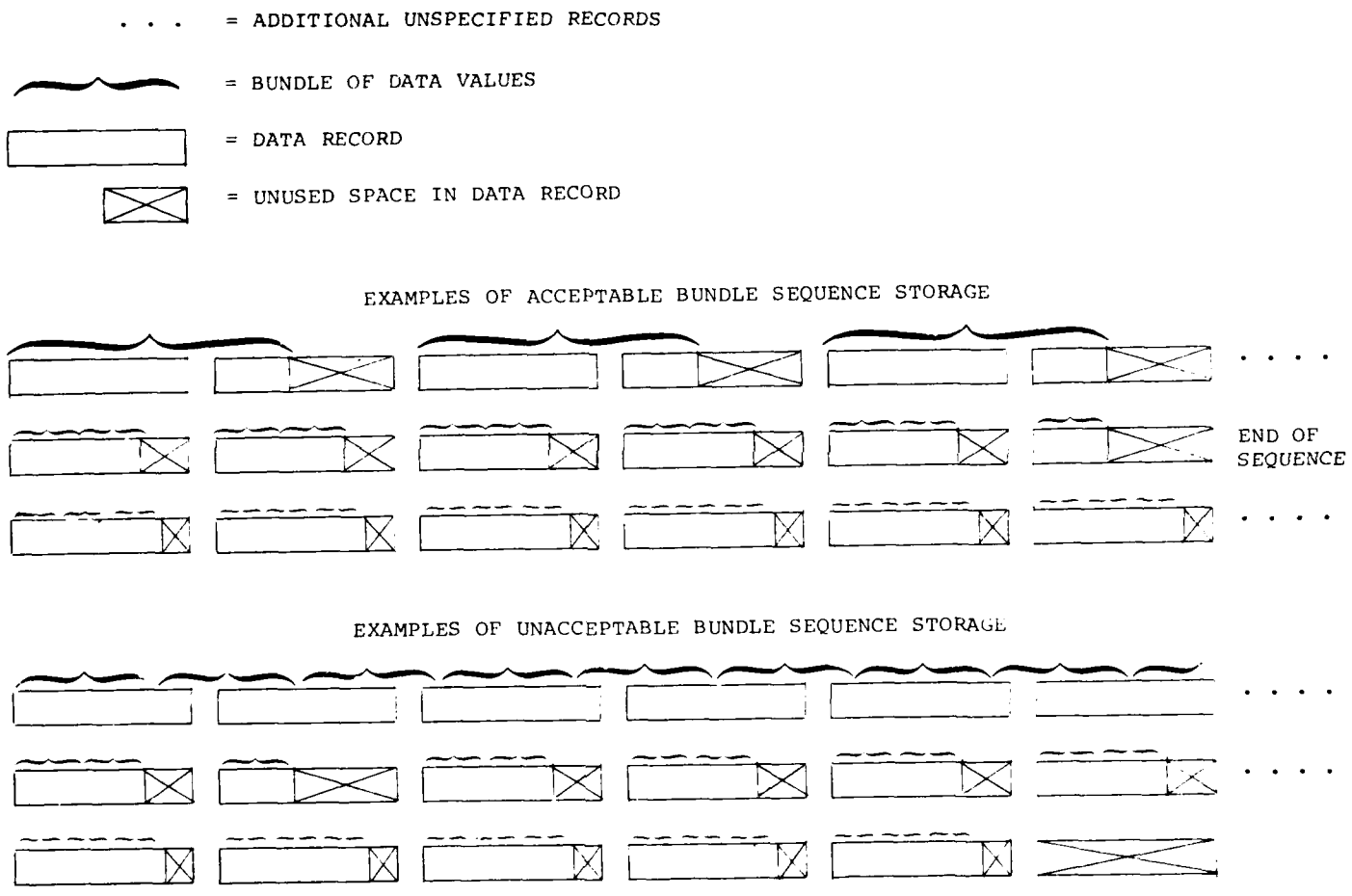


Figure 22. Storage of bundle sequences in data record sequences.

In Figure 22, three examples of unacceptable bundle sequence storage are given. In the first example, bundles that do not start at the beginning of a record span more than one record. In the second example, one record does not contain as many bundles as possible and this record is not the last for the bundle sequence. In the third example, an empty data record occurs.

3.3.3 Item Code and Counter Records

Each bundle sequence must be preceded by a definition sequence, which consists of one counter record and one or more item code records. Either the counter record or the item code records (but not both) may be implied. That is, the counter or item code records may be left out of the definition sequence and implied to be the same item code or counter records that applied to the previous bundle sequence. Clearly, the first definition sequence in the DTF may not have any implied records.

If more than one item code record is required, these records must be in contiguous sequence with no interspersed records of any other type. If an item code record specifies fewer than 128 item codes, it must be the last in this sequence. The counter record may appear before or after the item code records.

3.3.4 Info File Records

An Info File record must appear before all the item code records that reference the groups named in the Info File record. An Info File record may appear inside a definition sequence between a counter record and a set of item code records. Otherwise, an Info File record may be placed anywhere in the DTF that is not in violation of the rules stated so far.

An Info File group name should not appear twice in the Info File records of a DTF. If this event should occur, the first declaration of the group name, together with the accompanying structure and labels, will be assigned all item code record references to that group name. Groups will be generated for all other mentions of the group name, but these groups will contain only the "NULL" item code. Info Files with redundant group names will work properly if the first group with the redundant name is correct. That is, the subsequent, identical group names will be ignored. However, this is both inefficient and confusing.

There may be several item code references to the same row/column/double-row element of an Info File group. In this event, the last reference in the DTF is used.

3.3.5 Unspecified Record Types

Record types six through sixteen may be placed anywhere in the DTF that is not in violation of the rules stated so far. In such locations, these records will be ignored by the FCP.

3.3.6 Examples of Record Sequences

Figure 23 gives examples of acceptable and unacceptable record sequences. Six examples of unacceptable sequences are given. In the first example, the instruction records are not the first records on the DTF. For example two, two item code records are separated by a counter record. In the third sequence, there are two counter records in sequence with no intervening data records. In the fourth example, two item code records are separated, this time by an Info File record. In example five, the first bundle sequence in a DTF is not preceded by a complete definition sequence. That is, there is no item code record before the sequence and no item code record can be implied. In the last example, there is no definition sequence for the second bundle sequence. That is, the counter record and item code record(s) cannot both be implied as the previous counter and item code records.

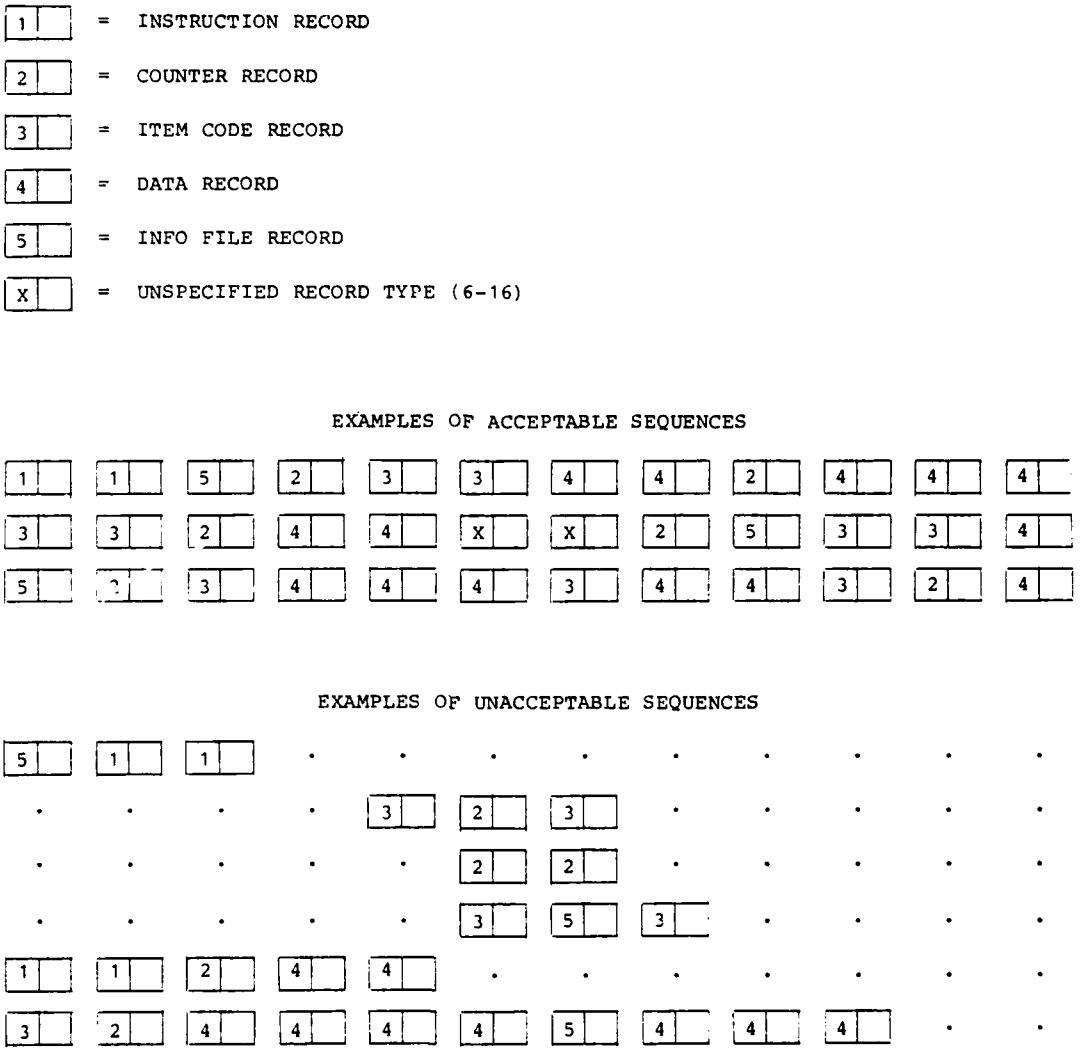


Figure 23. Examples of acceptable and unacceptable record sequences.

4. PROCESSING PROGRAM

4.1 STRUCTURE AND FLOW

The DATAMAP Processing Program was designed to be broken into overlays corresponding to various functions of the program. Figure 24 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 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/) that 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 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.

4.2 PROGRAM INITIALIZATION

Subroutine STRTUP is the control routine for this block. The required and optional user inputs for this phase are described in Section 5.1 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 that starts the CPU timer (see Section 6.4).

After the user options have been specified, STRTUP calls ALLSCR to initialize each of the direct access disc scratch areas, including SCF1, SCF2, SCF3, and the temporary scratch area. All of the scratch files are contained in a single direct access data set and are addressed by a single I/O file reference number. The files are addressed individually as different pseudo-devices using the RMS, WMS, and FMS subroutines. The pseudo-device numbers are listed in Section 5.1. The scratch files may be "PERMANENT" or "TEMPORARY." If

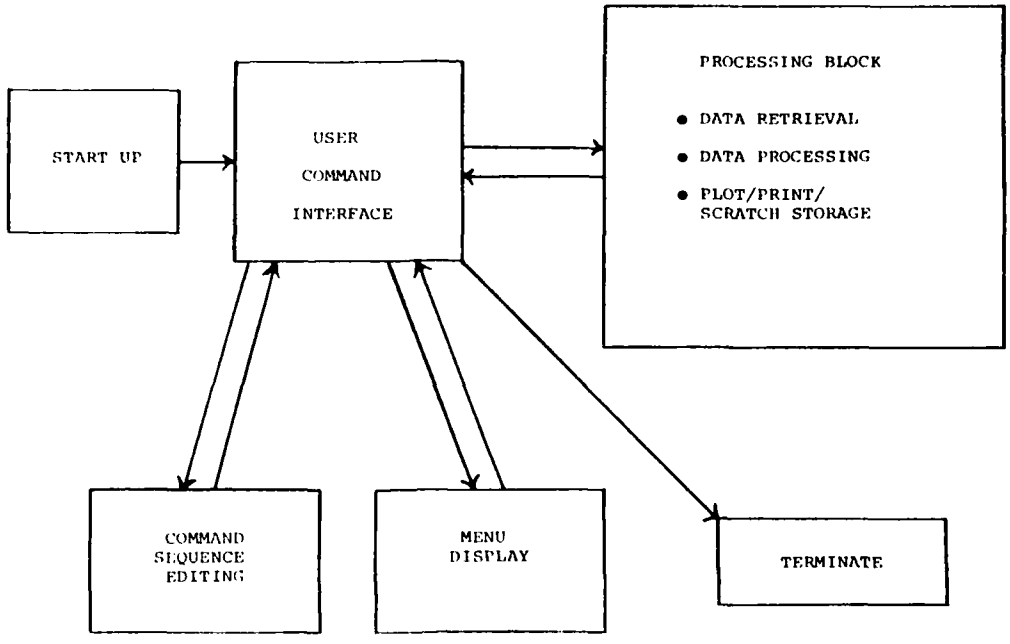


Figure 24. General flow of processing program.

the scratch files are labeled to be "PERMANENT," then no specific initialization is performed. Instead, the first and last record in the data set are read to test the initialization, Then, certain information about each scratch file that describes the file contents is stored in program memory. "TEMPORARY" scratch files must be initialized by writing dummy records on every record position of the sequential alias for the direct access file. This initialization is performed using subroutine INIDAF as described in Section 5. Notice that the "TEMPORARY" designation means that the scratch files will require initialization and does not describe the planned disposition of the files after the Processing Program run is complete. Thus, "TEMPORARY" scratch files may later be stored on a permanent basis.

The information that the scratch files are "PERMANENT" or "TEMPORARY" cannot be stored in the scratch files because an attempted direct access read from an uninitialized file could result in immediate termination of the program run. Instead this information is specified by parameter communication from the computer system, by user option specification (see Volume I, Section 5.1), or by default. Parameter communication is accomplished for IBM systems using subroutine PARMGT. A parameter string value of "PERM" indicates "PERMANENT" or any other value indicates "TEMPORARY." If parameter communication from the system to a program is not available for a particular computer operating system, then a dummy PARMGT subroutine may be used that indicates a "PERMANENT" or "TEMPORARY" setting by default. In either case, the user can override the setting during the Initialization Phase.

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 that is to be accessed in the first partition access slot during the program run (although the partition can be changed 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. The user may request a menu listing of partitions and subroutine MPARTX is used for this listing.

4.3 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 that 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 that 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 that 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 4.3.1 discusses this code, while Paragraph 4.3.2 covers the requirements for the block data tree structure.

4.3.1 *User Interface Routines*

USER is the main routine for the user interface block. Figures 25 and 26 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.

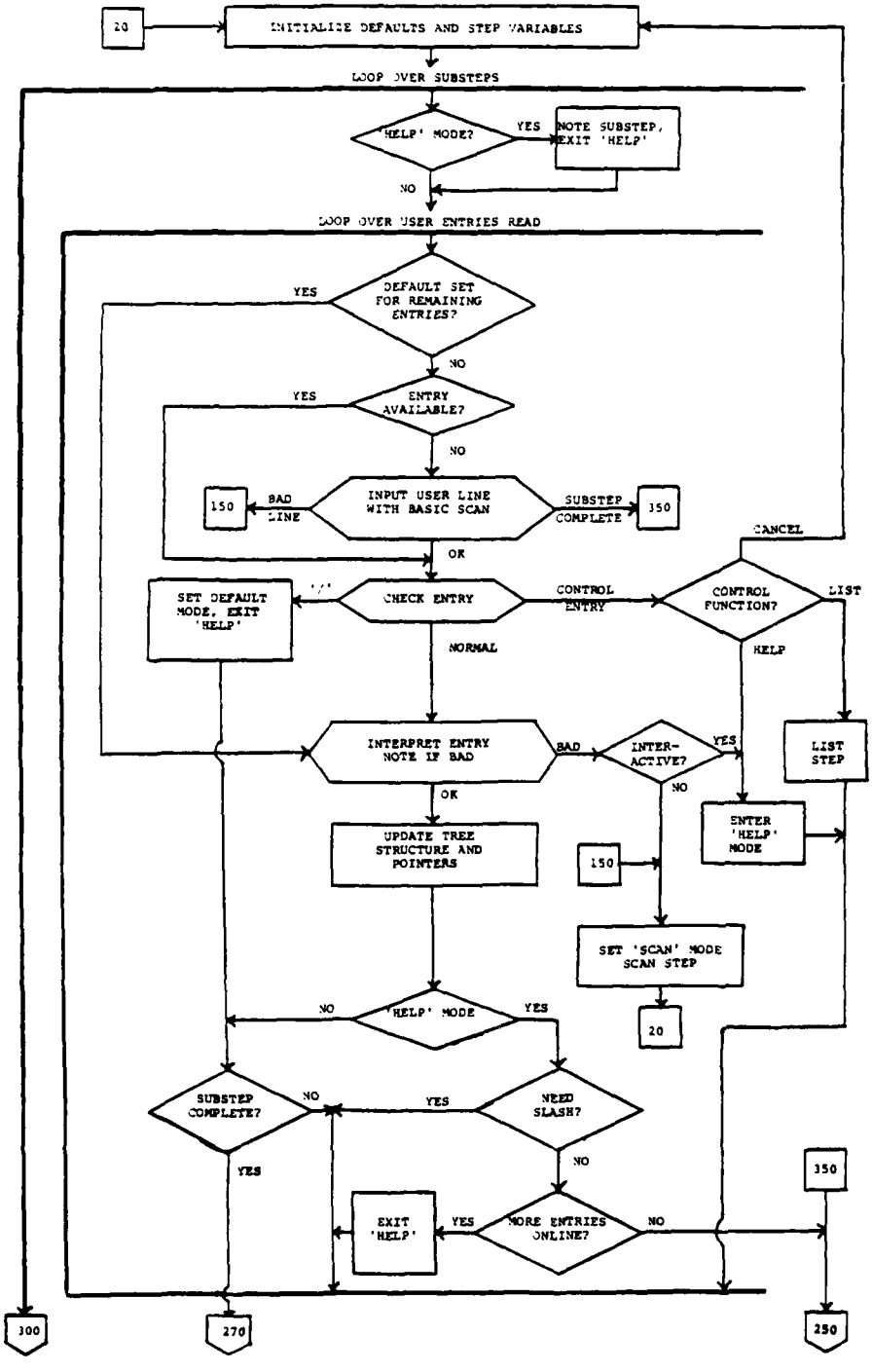


Figure 25. User interface flow diagram (first part).

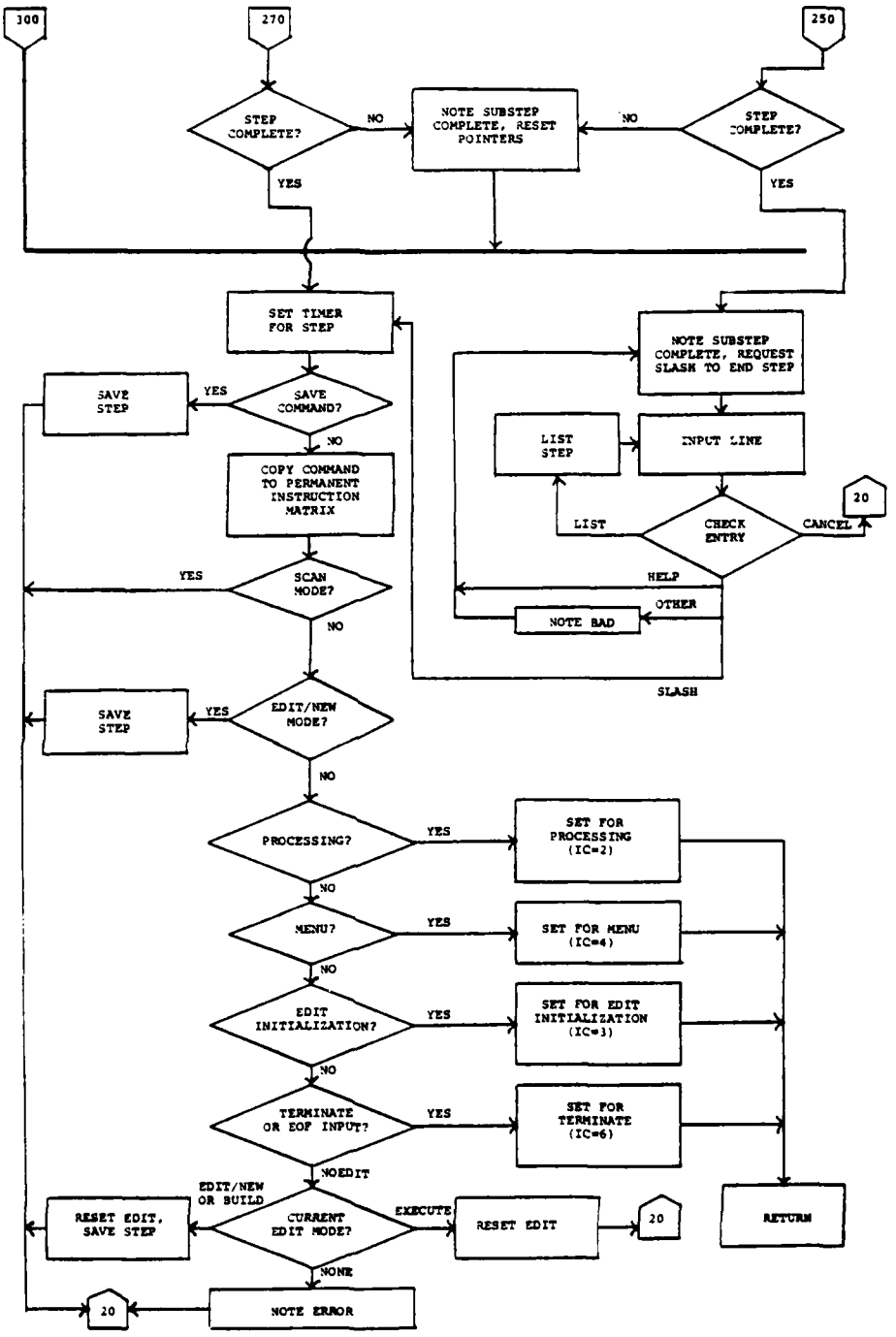


Figure 26. User interface flow diagram (second part).

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 that 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.

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 4.5.1) 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.

Subroutine USERLC is used to perform the required actions for certain commands inside the user interface block. The processing required for these commands is trivial. This routine performs processing for the commands "UTILITY," "SET," "COMMENT," "BUILD," "EDIT/NEW," and "NOEDIT."

4.3.2 User Input Encoding

The basic tree structure for the user interface is contained in the two-dimensional array NPOINT. The structure consists of many "nodes" and each node specifies a required entry for a command. A complete command is a specific path through one or more nodes. Some nodes allow branching in the command path depending upon the entry for the node. Each node has a specific index number. The second array subscript for NPOINT corresponds to the tree node index. Thus, each tree node 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 subscript 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 27 shows a general example of part of the NPOINT structure.

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, 98, 9, 0, 44, 0, 13, 58)	1
2	2	1	ANALYZE ACTION SELECTION (2)	710	-110 (14,15,16, 17,19,19,117,118, 119,120,121,122)	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	1, 46, 47,	DATA SOURCE (SCF1,SCF2, SCF3, GROUP OR ITEM CODE) (29)	630	-49, (39,39,39, 40,41)	151
5	3	1	COMMAND SEQUENCE FUNCTION (26)	189	3044	134
6	3	7	COMMAND SEQUENCE EXECUTION FIRST ARGUMENT (52)	535	3008	324
7	3	1	COMMAND SEQUENCE NAME (25)	183	3006	131
8	3	6	COMMAND SEQUENCE EXECUTION SECOND ARGUMENT (53)	542	3018	327
9	3	1	MENU TYPE (28)	205	0	303

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
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
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	3	8	COMMAND SEQUENCE EXECUTION THIRD ARGUMENT (54)	549	3089	330
19	3	2, 12	DATA SOURCE (SCF1,SCF2, SCF3, GROUP OR ITEM CODE) (29)	630	-81 (39,39,39,27, 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

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
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
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	110
36	3	38	SCF1, SCF2, OR SCF3 (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

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
41	3	4	COUNTER (19)	156	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
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, 56,150)	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

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
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, SCF2, OR SCF3 (60)	442	4096	416
61	3	35, 50	DATA SOURCE (SCF1, SCF2, SCF3, GROUP OR ITEM CODE) (29)	630	-69 (39,39,39, 71,72)	151
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, SCF3, OR GROUP) (29)	455	-75 (73,73,73, 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

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
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
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	DOUBLE-ROW 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

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
87	4	77	OBSERVER LOC IN X (44)	507	4095	249
88	3	71	DOUBLE-ROW SELECTION (65)	487	3097	381
89	3	18	COMMAND SEQUENCE EXECUTION FOURTH ARGUMENT (58)	556	0	345
90	3	81	AZIMUTH	173	4049	125
91	3	82	ALL OR COLUMN ELEMENT (33)	251	4049	173
92	3	40	DOUBLE-ROW SELECTION (65)	487	3105	381
93	4	84, 85	X-INTERVAL (49)	396	-91 (100,101, 100)	280
94	4	86	CONTOUR VALUE (43)	352	113	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	2	1	RUN SETTING CHANGE (66)	578	0	388
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

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
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	DOUBLE-ROW 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
110	3	69	X-AXIS SCALE (51)	497	3083	293
111	3	73	X-AXIS SCALE (51)	497	3082	293
112	3	71	DOUBLE-ROW SELECTION (65)	487	3097	381
113	4	94	MAX NUMBER OF CONTOUR LEVELS (11)	571	0	83
114	3	158	MASKED ITEM CODE (20)	600	0	116
115	3	158	PARTITION NAME (67)	608	3116	406
116	3	115	PARTITION ACCESS SLOT (68)	618	0	411

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
117	2	2	MAX FREQ FOR COHERENCE FUNCTION (8)	102	2138	71
118	2	2, 119	MAX FREQ FOR RESPONSE OR CROSS-DENSITY (8)	102	2140	71
119	2	2	CROSS ANALYSIS TYPE (69)	645	-128(118,125)	443
120	2	2	AUTO ANALYSIS TYPE (69)	645	-130(126,127)	443
121	2	2	STATISTICS ANALYSIS TYPE (70)	653	-150(128,128, 128,137)	447
122	2	2	TYPE OF ACOUSTIC ANALYSIS (71)	668	-157(149,149,148, 149,149,149,149, 149)	457
123	3	117, 124	INPUT FOR COHERENCE FUNCTION (29)	442	3130	203
124	3	118	ENSEMBLE SELECTION FOR RESPONSE AND CROSS ANALYSIS (72)	685	-146(123,131)	471
125	2	119	MAX OFFSET FOR CROSS CORRELATION (73)	694	2143	475
126	2	120	MAX FREQ FOR AUTO DENSITY (8)	102	2144	71
127	2	120	MAX OFFSET FOR AUTO CORRE-LATION (73)	694	2147	475

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
128	3	121, 126, 127	ENSEMBLE SELECTION FOR STATISTICS AND AUTO PROCESSES (72)	685	-148(129,4)	471
129	3	122	SCRATCH FILE FOR AUTO AND STATISTICS INPUT (29)	442	3132	203
130	3	123	COHERENCE FUNCTION SECOND SCRATCH FILE INPUT (74)	700	3132	478
131	3	124	SCRATCH FILE FOR RESPONSE AND CROSS ASSUMING INDIVIDUAL - FIRST INPUT (29)	442	3133	203
132	3	130	ROW SELECTION FOR COHERENCE FUNCTION (32)	244	3134	168
133	3	131	SCRATCH FILE SECOND INPUT FOR RESPONSE AND CROSS (74)	700	3135	478
134	3	132	DOUBLE-ROW FOR COHERENCE FUNCTION (65)	487	4049	381
135	3	133	ROW SELECTION FOR RESPONSE AND CROSS (32)	244	3136	168
136	3	135	COLUMN SELECTION FOR RESPONSE AND CROSS (33)	265	3134	173
137	2	121	CHI-SQUARE TEST NUMBER OF BINS (75)	755	3128	485

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
138	2	117	WINDOW FOR COHERENCE FUNC-TION (9)	767	2139	489
139	2	138	ADJACENT POINT AVERAGING FOR COHERENCE FUNCTION (76)	781	3123	497
140	2	118	WINDOW FOR RESPONSE AND CROSS DENSITY (9)	767	2141	489
141	2	140	ADJACENT POINT AVERAGING FOR RESPONSE AND CROSS DENSITY (76)	781	3124	497
142	-	-	-	-	-	-
143	2	125	NORMALIZE SELECT FOR CROSS CORRELATION (78)	812	3124	507
144	2	126	WINDOW FOR AUTO DENSITY (9)	767	2145	489
145	2	144	ADJACENT POINT AVERAGING FOR AUTO DENSITY (76)	781	3128	497
146	-	-	-	-	-	-
147	2	127	NORMALIZE SELECT FOR AUTO CORRELATION (78)	812	3128	507
148	2	122	NARROW BAND FILTER BAND-WIDTH (79)	825	2149	512

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
149	2	122, 148	ACOUSTIC CORRECTION LEVEL (80)	838	3004	515
150	4	49	X SCALE FOR DPLOT (49)	302	4151	210
151	4	150	CURSOR SELECT FOR DPLOT (40)	325	4152	227
152	4	151	TOP DOUBLE-ROW Y-SCALE INTERVAL (47)	850	-166(153,154, 153)	267
153	4	152	TOP DOUBLE-ROW Y-SCALE MIN VALUE (48)	865	4155	274
154	4	152	TOP DOUBLE-ROW Y-SCALE NUMBER OF DECADES (56)	422	4155	339
155	4	153, 154	BOTTOM DOUBLE-ROW Y-SCALE INTERVAL (81)	880	-170(156,157, 156)	518
156	4	155	BOTTOM DOUBLE-ROW Y-SCALE MIN VALUE (82)	896	4093	524
157	4	155	BOTTOM DOUBLE-ROW Y-SCALE NUMBER OF DECADES (83)	422	4093	527
158	2	1	UTILITY ACTION (84)	912	-174(114,159, 115,0)	531
159	3	158	UNMASKED ITEM CODE (20)	925	0	116

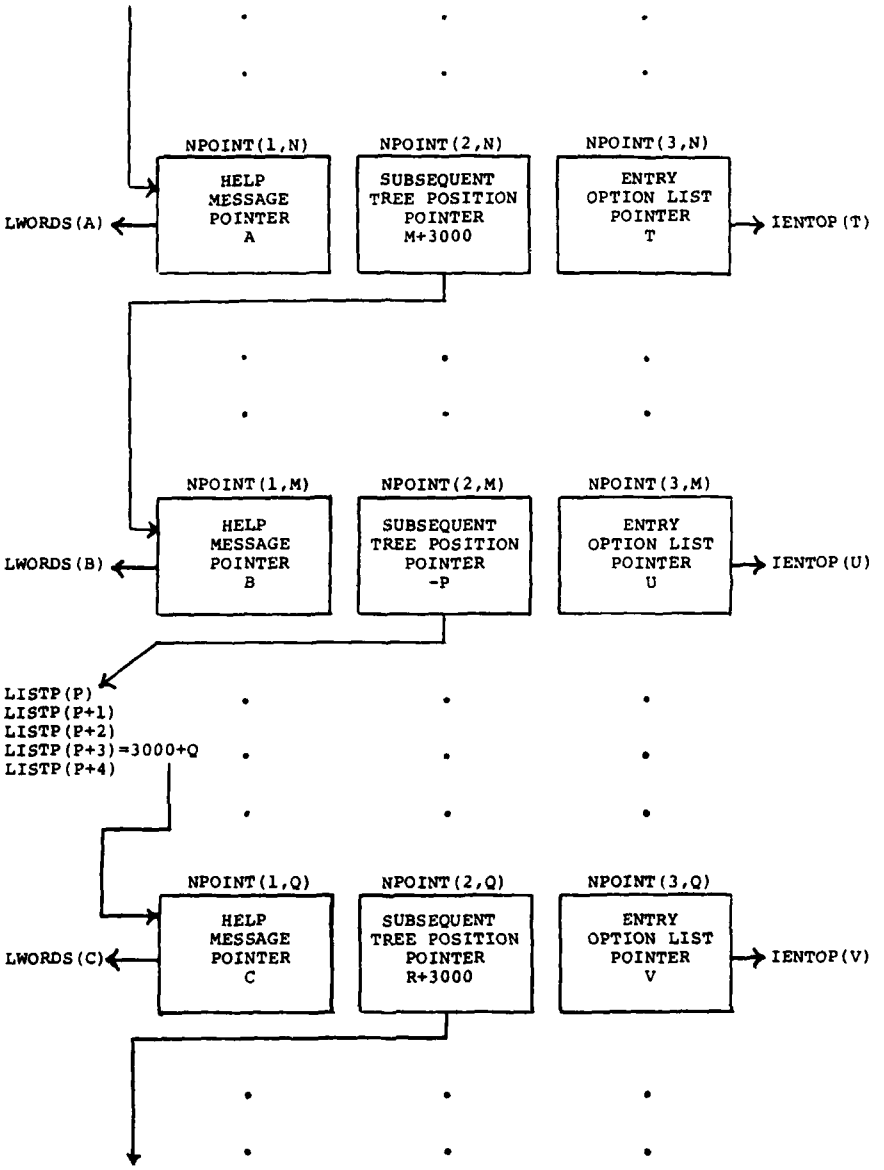


Figure 27. Example of part of the command entry tree structure.

NPOINT (1,N) gives the subscript location in LWORDS (in common block /HLPWDS/) for the appropriate HELP message for the entry options corresponding to the tree node N. The actual subscript 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 28 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 are the tree node index 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 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, points to possible next tree nodes 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 that 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 3 lists the meaning of the instruction matrix positions. Following the second word in the IENTOP sequence are one or more numeric values that specify allowed options for the entry. The second word is an integer that specifies the number of option specifications. The option specifications in IENTOP are integers that 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 29 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

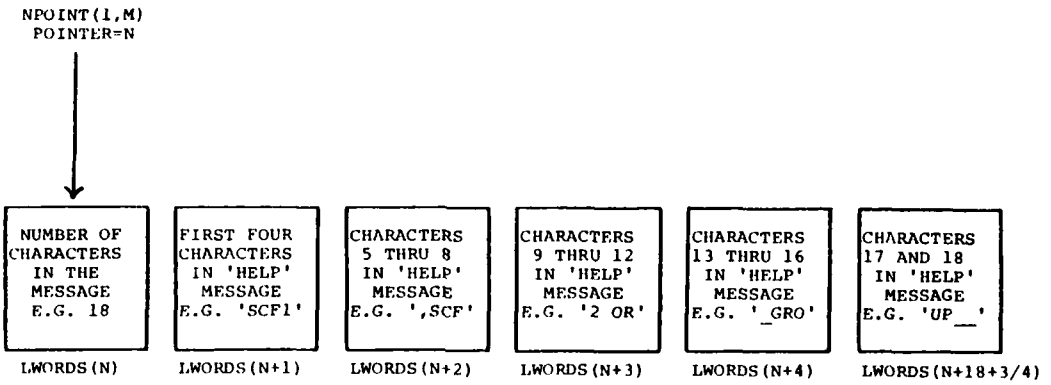


Figure 28. Structure of typical "HELP" message.

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 SET 8 MENU 9 TERMINATE 10 BUILD 11 SAVE 12 COMMENT 13 UTILITY	-	1
2	ANALYSIS ACTION	1 HARMONIC 2 FILTER 3 SPECTRUM 4 DAMPING 5 AVERAGE 6 MMAX 7 COHERENCE 8 RESPONSE 9 CROSS 10 AUTO 11 STATISTIC 12 ACOUSTIC	-	1
3	DERIVE ACTION	1 TAS 2 MRPM 3 ---- 4 MSHP	-	1

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
		5	----		
		6	CP		
		7	CN		
		8	CC		
		9	CM		
		10	MCT		
		11	---		
		12	MCQ		
		13	---		
		14	MFLO		
		15	DFLO		
		16	BLDIS		
		17	SLOPE		
		18	DENALT		
		19	MRAZ		
		20	----		
4	NUMBER OF HARMONICS	1	1 THRU 100	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		
		4	HALF		

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
10	DAMPING FREQUENCY	1	0. THRU 1.E+5	-	1
11	MAX NUMBER OF CONTOUR LEVELS	1	1. THRU 128.	16	5
12	OUTSIDE AIR TEMPERATURE	1 2	CALCULATE -60 THRU +60	CALC	4
13	STATIC PRESSURE	1 2	CALCULATE 1.0 THRU 18.0	CALC	4
14	CONVERSION SLOPE FOR TRUE AIRSPEED CALCULATION	1	.5 THRU 2.0	1.0	5
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

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
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 3 4 5 6 7 8	DATA INFO SCRATCH EDIT PARTITION MASK SET 1 THRU 999,999	-	1
29	INPUT SOURCE	1 2 3 4 5	SCF1 SCF2 SCF3 GROUP 4 CHARACTER STRING (ITEM)	-	1
30	SCF1, SCF2 or SCF3 INPUT FIRST DIMENSION	1 2	ALL 0.0 THRU 1000	ALL	2

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
31	-	-	-	-	-
32	SCF1, SCF2 or SCF3 INPUT, 2ND DIMENSION EXTENT	1 2	ALL 1 THRU 64	ALL	2
33	SCF1, SCF2 or SCF3 INPUT, 3RD DIMENSION EXTENT	1 2	ALL 1 THRU 64	ALL	2
34	INFO FILE GROUP NAME	1	4 CHARACTER STRING	-	1
35	INFO FILE GROUP INPUT, COLUMN ELEMENT SPECIFICATION	1 2	ALL 1 THRU 64	ALL	2
36	INFO FILE GROUP INPUT, ROW ELEMENT SPECIFICATION	1 2	ALL 1 THRU 64	ALL	2
37	OUTPUT SELECTION	1 2 3 4 5 6 7 8 9 10	PLOT MPLOT APLOT PRINT CONTOUR SURFACE KEEP ADD LPLOT DPLOT	NONE	1
38	-	-	-	-	-

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

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

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
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
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

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS	INITIAL DEFAULT	DEFAULT CONTROL**
51	SCF1 OR SCF2 INPUT, SCALE FOR FIRST INDEPENDENT VARIABLE VALUE SPECIFIED AS INSTRUCTION 30	1 IMPL 2 MRAZ 3 TAS 4 MRPM	IMPL	2
52	1ST COMMAND SEQUENCE EXECUTION ARGUMENT	(SPECIAL CASE)		
53	2ND COMMAND SEQUENCE EXECUTION ARGUMENT	(SPECIAL CASE)		
54	3RD COMMAND SEQUENCE EXECUTION ARGUMENT	(SPECIAL CASE)		
55	USER SUPPLIED VALUE FOR COLUMN POSITION	1 NONE 2 -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	FOURTH COMMAND SEQUENCE EXECUTION ARGUMENT	(SPECIAL CASE)		

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
59	ANGLE BETWEEN THE INBOARD POINTING DETECTOR OF THE BOUNDARY LAYER BUTTONS AND THE CHORDLINE	1	-90 THRU +90	45	5
60	SCRATCH FILE SELECTION FOR OUTPUT	1 2 3	SCF1 SCF2 SCF3	-	1
61	FORMAT FOR SURFACE OR CONTOUR PLOT	1 2	CYLI RECT	RECT	2
62	-	-	-	-	-
63	-	-	-	-	-
64	-	-	-	-	-
65	DOUBLEROW ELEMENT SELECTION	1 2 3	BOTH TOP BOTTOM	BOTH	2
66	RUN SETTING	1 2 3 4 5 6 7 8 9 10	MAIN TAIL FULL HALF GRID NOGRID TICS NOTICS QUICK SLOW	-	1

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
		11	COPY		
		12	NOCOPY		
67	PARTITION	1	NONE	-	1
		2	(Partition Name)		
68	PARTITION ACCESS SLOT	1	FIRST	FIRST	2
		2	SECOND		
69	CROSS OR AUTO ANALYSIS TYPE	1	DENSITY	-	1
		2	CORRELATION		
70	STATISTICS ANALYSIS TYPE	1	MEAN	-	1
		2	VARIANCE		
		3	DEVIATION		
		4	FIT		
71	TYPE OF ACOUSTIC ANALYSIS	1	OCTAVE	-	1
		2	THIRD		
		3	NARROW		
		4	DBA		
		5	DBB		
		6	DBC		
		7	DBD		
		8	PNL		
72	ENSEMBLE SELECTION	1	ENSEMBLE	-	1
		2	INDIVIDUAL		
73	MAX OFFSET	1	0.0 THRU 1000.	0.0= HALF THE RECORD LENGTH	3

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (CONTINUED)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS		INITIAL DEFAULT	DEFAULT CONTROL**
74	SECOND INPUT SCRATCH FILE	1 2 3	SCF1 SCF2 SCF3	-	1
75	CHI-SQUARE TEST NUMBER OF BINS	1	3 THRU 100	7.0	3
76	ADJACENT POINTS TO AVERAGE	1	0. THRU 256.	0.0	3
77	-	-	-	-	-
78	NORMALIZE SELECT FOR CORRELATION	1 2	NORM NONORM	NORM	2
79	NARROW BAND FILTER BAND-WIDTH	1	1.0 THRU 99.0	8.0	5
80	ACOUSTIC ATTENUATOR CORRECTION LEVEL	1	-100. THRU 100.	0.0	5
81	DPLOT BOTTOM DOUBLE-ROW INTERVAL SELECTION	1 2 3	AUTO LOG -1.E34 THRU 1.E34	AUTO	2
82	DPLOT BOTTOM DOUBLE-ROW START POINT	1 2	AUTO -1.E34 THRU 1.E34	AUTO	2
83	DPLOT BOTTOM DOUBLE-ROW NUMBER OF DECADES	1	1 THRU 12	3	3

TABLE 3. USER INTERFACE INSTRUCTION MATRIX (Concluded)

MATRIX NUMBER	INSTRUCTION MEANING	INSTRUCTION OPTIONS	INITIAL DEFAULT	DEFAULT CONTROL**
84	UTILITY ACTION	1 MASK 2 UNMASK 3 PARTITION 4 COPY	-	1

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

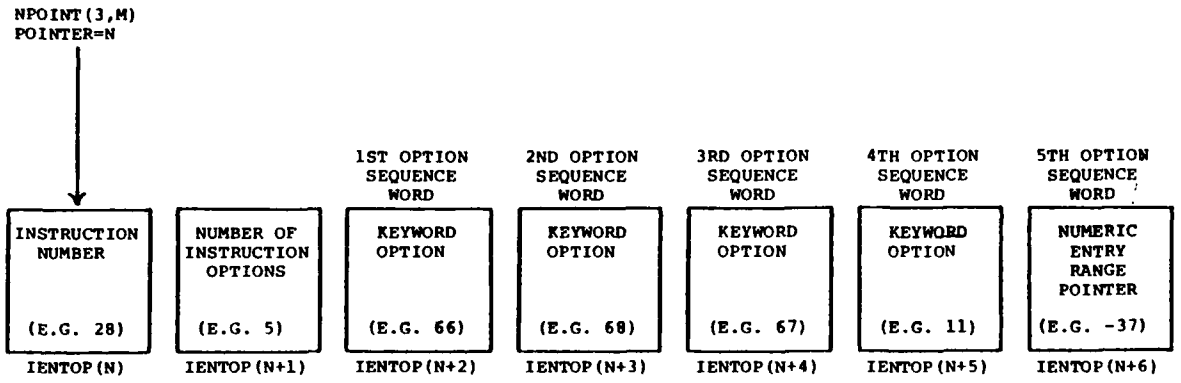


Figure 29. Typical IENTOP instruction option sequence.

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 that is positive and less than 10^0 indicates the position in the IENTOP option sequence for the option selected. For 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 that 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 index 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.

4.4 PROCESSING

4.4.1 Processing Flow

The control routine for the processing block is PROCES and the flow for this routine is shown in Figures 30 and 31. 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 must loop over all row and column positions for the specified input data. Normally, the outside loop is over column positions and the inside loop is over the row positions. However, for ensemble averaging, this order is reversed. In the non-ensemble averaging case, PROCES enters a DO loop that covers the column positions (radial stations). Inside this loop, PROCES calls ATTGET to retrieve and/or calculate the

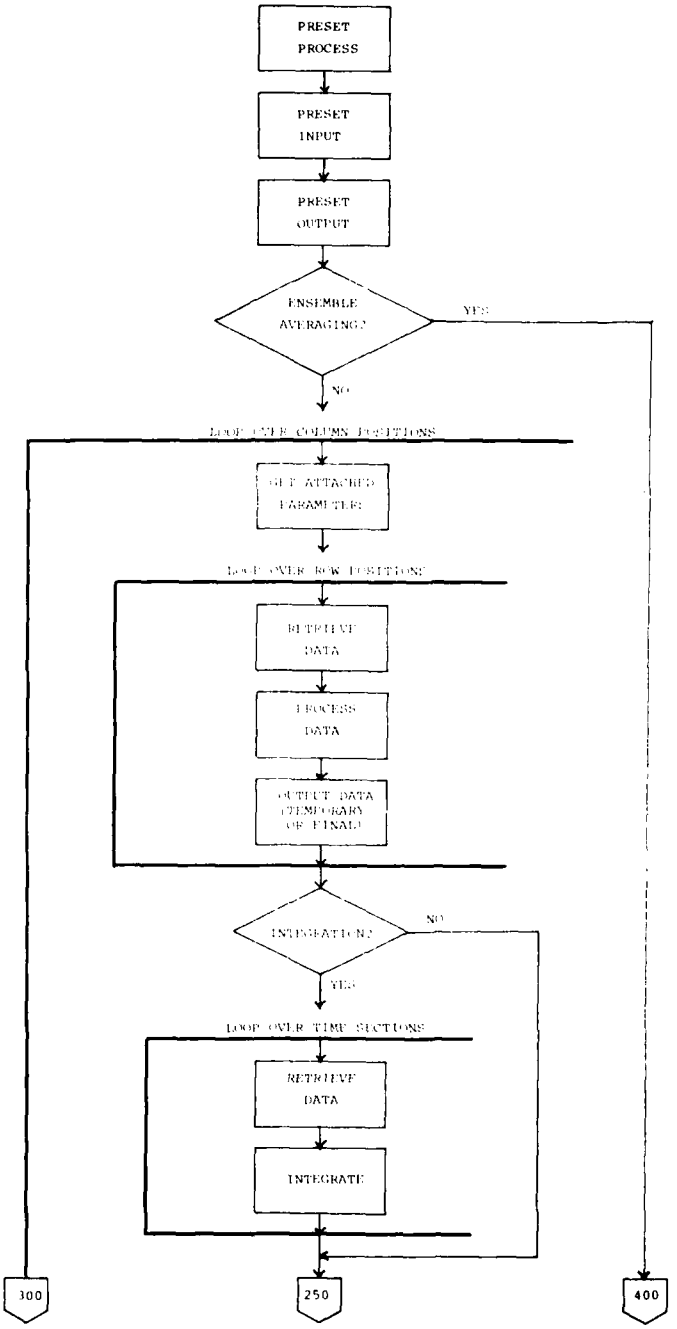


Figure 30. Processing flow diagram (first part).

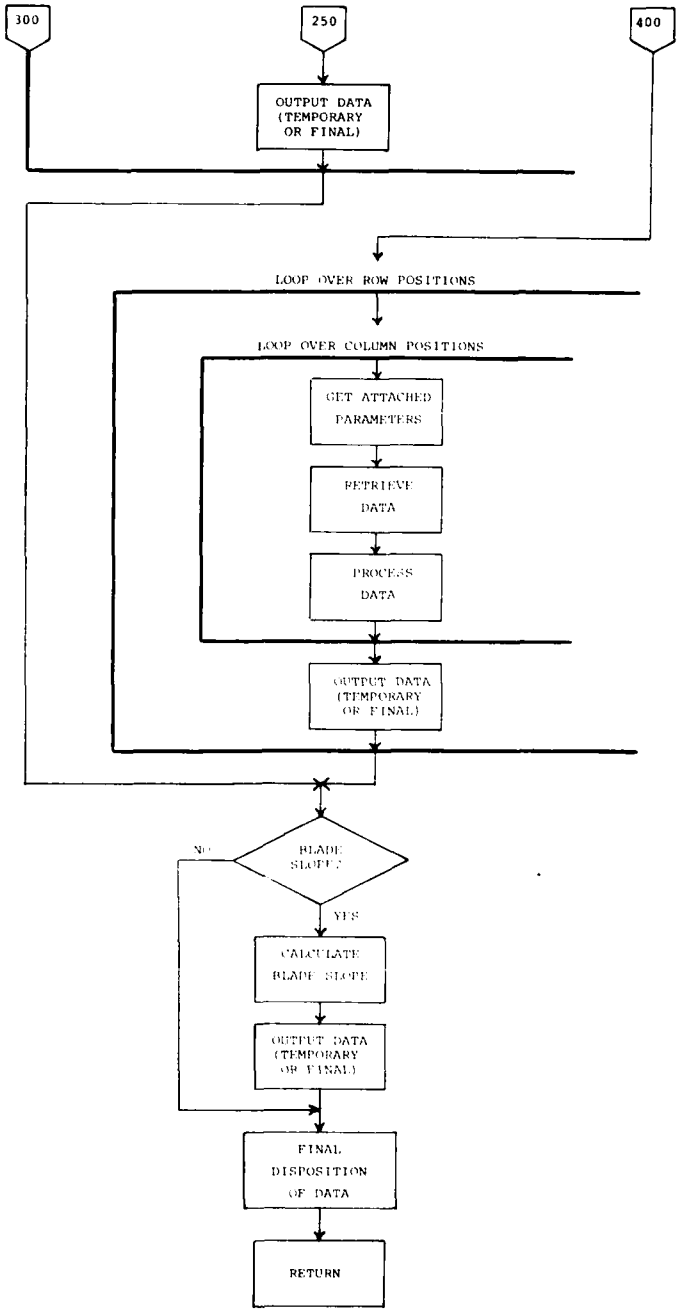


Figure 31. Processing flow diagram (second part).

attached parameters for the appropriate counter and time span. ATTGET is called once for each column position since the counter or start time within a 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, SCF2, or SCF3. GETDAT may retrieve the data stream from the Master File according to the user item code or Info File specification, or GETDAT may call RTRVSC to retrieve the data from SCF1, SCF2, or SCF3.

PRO1 addresses most of the available processing routines. PRO1 does not address processes that 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, SCF2, or SCF3 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 that 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_m integrations). If not, PROCES jumps ahead to a call to TSAV2. If so, PROCES enters a DO loop that 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, SCF2, or SCF3, 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 that 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, SCF2, or SCF3 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.

At this point, at address 400, the block diagram depicts the flow for processing that uses ensemble averaging. Ensemble averaging is available for a restricted set of analyses that does not include the multiple chord position integrations. The outside loop for ensemble-averaged processing is over row positions and the inside loop is over columns. Inside the loops, attached parameter data are retrieved using subroutine ATTGET, input data are retrieved using subroutine GETDAT, and the appropriate process subroutine is selected by subroutine PRO1. Results of a process need not be stored until the loop over the columns is complete and subroutine TSAV1 is used for this purpose. Outside the loop over row positions, flow joins the flow from non-ensemble-averaged processing.

When the flow sequences from ensemble-averaged and non-ensemble-averaged processing rejoin, 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.

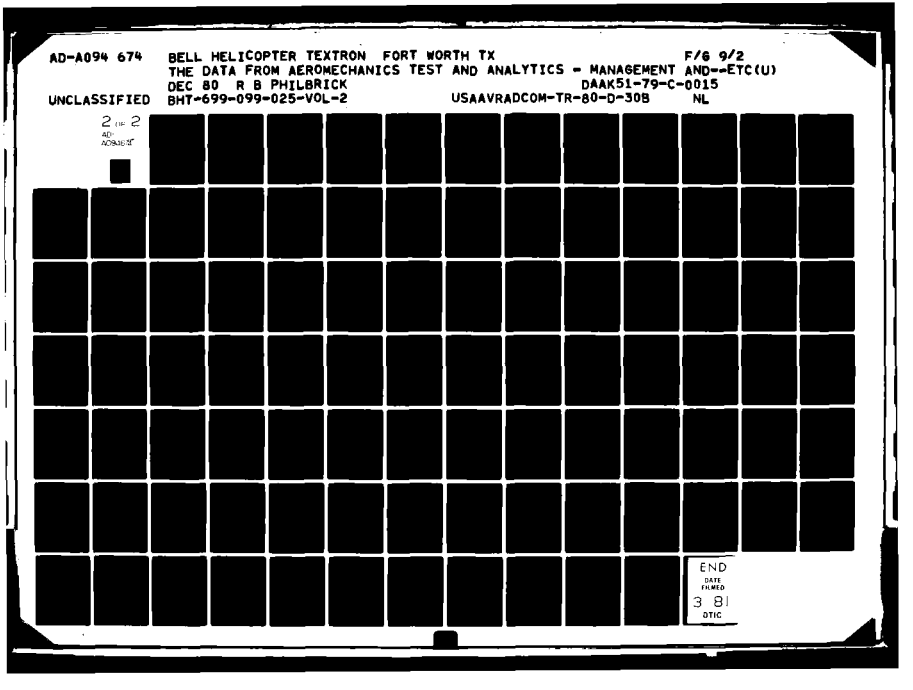
The final routine called by PROCES is DISPOS. DISPOS selects the proper routine to perform the output. If the output is to SCF1, SCF2, or SCF3 or to printout, then the output process has been completed in TSAV1, TSAV2 or TSAV3 and DISPOS simply returns. Otherwise, DISPOS calls the appropriate routine 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.

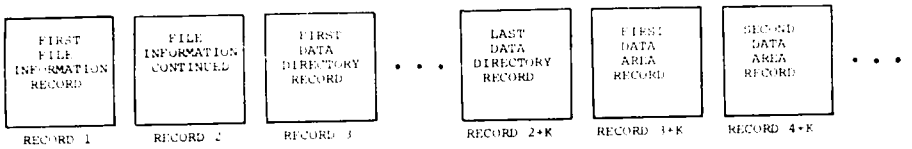
When DISPOS returns, PROCES sets the subroutine argument, IC, to one and returns. MAIN transfers program control back to USER.

4.4.2 Scratch Files

Scratch files SCF1, SCF2, and SCF3 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 32 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 data and other information including the counter for the column. The column position and attached parameter location pointers may continue into the second record of the scratch file. Figure 33 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 34 shows a data directory record. Data directory blocks require 12 words (48 bytes) and scratch file records contain 256 words (1024 bytes) so that 21 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.


AD-A094 674
BELL HELICOPTER TEXTRON FORT WORTH TX
THE DATA FROM AEROMECHANICS TEST AND ANALYTICS - MANAGEMENT AND--ETC(U)
DEC 80 R B PHILBRICK
DAAK51-79-C-0015
UNCLASSIFIED
BHT-699-099-025-VOL-2
USAAVRADCOM-TR-80-D-30B
NL
2 OF 2
AD-A094674
END
DATE FILMED
3 81
DTIC



K=((MAXIMUM ALLOWED ROWS) x (MAXIMUM ALLOWED COLUMNS) + 20)/21

Figure 32. Scratch file record assignments.

START WORD	LENGTH (WORDS)	ENTRY CONTENTS
1	3	OVERALL UNITS (3A4)
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 LOCATION
26	1	ATTACHED PARAMETER INDICATOR: 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 DOUBLE-ROW ELEMENT KEYWORD
40	1	BOTTOM DOUBLE-ROW ELEMENT KEYWORD
41	13	GENERAL LABEL FOR DATA (13A4)
54	1	INDEPENDENT VARIABLE: 0=NO DATA, 1=TIME, 2=FREQ, 3=HARM
55	1	TOP DOUBLE-ROW LABEL INDICATOR: 1=LABEL, 2=NO LABEL
56	1	BOTTOM DOUBLE-ROW LABEL INDICATOR: 1=LABEL, 2=NO LABEL
57	1	AZIMUTH OFFSET FOR INPUT FILE GROUP
58	5	TOP DOUBLE-ROW LABEL (5A4)
63	5	BOTTOM DOUBLE-ROW LABEL (5A4)
68	1	LUSCRD UNIT COMBINATION INDICATOR
69	13	LABEL FOR SECOND DATA INPUT (i.e., FOR CROSS-PROCESS)(13A4)
82	3	UNIT LABEL FOR SECOND DATA INPUT (3A4)
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
ICPOFF+2	1	FIRST DIMENSION VALUE FOR FIRST DATA POINT
ICPOFF+3	1	DATA STREAM NUMBER OF ENTRIES
ICPOFF+4	1	RECORD LOCATION FOR AZIMUTH DATA
ICPOFF+5	1	NUMBER OF POINTS OF AZIMUTH DATA
ICPOFF+6	1	RECORD LOCATION FOR AIRSPEED DATA
ICPOFF+7	1	NUMBER OF POINTS OF AIRSPEED DATA
ICPOFF+8	1	RECORD LOCATION FOR RPM DATA
ICPOFF+9	1	NUMBER OF POINTS OF RPM DATA
ICPOFF+10	1	RECORD LOCATION FOR STATIC PRESSURE DATA
ICPOFF+11	1	NUMBER OF POINTS OF STATIC PRESSURE DATA
ICPOFF+12	1	RECORD LOCATION FOR OUTSIDE AIR TEMPERATURE DATA
ICPOFF+13	1	NUMBER OF POINTS OF OUTSIDE AIR TEMPERATURE DATA
ICPOFF+14	1	COUNTER (INTEGER)
ICPOFF+15	1	COLUMN POSITION NUMBER 2
.	.	.
.	.	.
.	.	.
ICPOFF+29	1	COLUMN POSITION NUMBER 3
.	.	.
.	.	.
.	.	.
.	.	.

Figure 33. First scratch file record.

START WORD	LENGTH (WORDS)	ENTRY CONTENTS
.	.	.
.	.	.
.	.	.
L+1	1	TOP DOUBLE-ROW ELEMENT START LOCATION IN FILE
L+2	1	TOP DOUBLE-ROW ELEMENT MINOR GEOMETRIC POSITION
L+3	3	TOP DOUBLE-ROW SHORT DATA STREAM LABEL (3A4)
L+6	1	TOP DOUBLE-ROW ELEMENT NUMBER OF DATA POINTS
L+7	1	BOTTOM DOUBLE-ROW ELEMENT START LOCATION IN FILE
L+8	1	BOTTOM DOUBLE-ROW ELEMENT MINOR GEOMETRIC POSITION
L+9	3	BOTTOM DOUBLE-ROW SHORT DATA STREAM LABEL (3A4)
L+12	1	BOTTOM DOUBLE-ROW ELEMENT NUMBER OF DATA POINTS
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.

Figure 34. Structure of a data directory block.

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, SCF2, or SCF3. 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.

4.4.3 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 common block /SINGIF/ (see Appendix B). In the processing block, subroutine SINGGP interrogates this common block to extract the appropriate item code for a particular keyword. If INFOST encounters unit conversion instructions, it calls subroutine UNINIT to read and store these instructions in common block /SINGIF/. UNINIT can read and store in program memory as many as 64 unit conversion lines. In the processing block, subroutine CONVCK tests dependent variable unit labels against the stored units and reports a conversion if one is specified. If the Info File includes more than 64 unit conversion lines and the lines stored in program memory do not match the output dependent variable units, then CONVCK scans the additional unit conversion lines for a match. The input and processing of the initial group of the Info File by INFOST and UNINIT constitutes a complete test for format accuracy of the initial group.

For Info File geometric groups, the processing block calls subroutine COMPGP to scan the Info File and find the specified group name. COMPGP then calls subroutine READGP to read and transfer the group information into the common block /INFGRP/. The processing block calls subroutine INFO2 to extract the proper item code and geometric position from /INFGRP/ for each specified row/column intersection.

All of the geometric groups are tested for format accuracy the first time that the menu subroutine INFRED is called (see Section 4.6). INFRED calls subroutine READGX to perform the format testing for each group.

4.4.4 Replacement/Addition of Analysis or Derivation Routines

Most routines that 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 half of XBUFF at location IBFSIZ/2 + 1 where IBFSIZ (in common block /SIZES/) is the array size of XBUFF.

For cross processes (e.g., for cross-correlation analysis), XBUFF may contain as many as four separate time histories. These histories correspond to the top and bottom double-row elements of the first and second input functions for the cross process. Following are the time history start points in XBUFF for this case.

1	-	Top double-row, first input function
IBFSIZ/4 + 1	-	Top double-row, second input function
IBFSIZ/2 + 1	-	Bottom double-row, first input function
3*IBFSIZ/4 + 1	-	Bottom double-row, second input function

For cross processes with a single double-row element input (i.e., the top double-row element), the first input function begins at location one in XBUFF and the second function is stored beginning at location IBFSIZ/2 + 1. Both input data functions must be the same length (in time and in number of datum points) for cross processes.

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. In particular, XSPARE will store intermediate results for ensemble averaging processes.

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/2 + 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 keyword should be set to zero. If a double-row element is present, there are three cases for output keyword selection. The keyword 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).

The interface routine must also set the double-row element labels, LBDTOP and LABBOT, and the indicators for these labels, LTOPON and LBOTON. All of these arrays and variables are in common block /MLABLS/. The settings that are required depend upon the relationship between the input and output double-rows. These relationships and the appropriate settings are listed below.

> There are both input double-row elements and these are individually processed into both output double-row elements in the same order. The arrays and variables should be left unchanged.

- There are both input double-row elements and these are processed together to create the top output double-row element or to create both output double-row elements. Both LTOPON and LBOTON should be reset to zero.

- There is only a top double-row element, which is processed into the top double-row element on output. The arrays and variables should be left unchanged.

- There is only a top double-row element, which is processed into both double-row elements on output. LTOPON and LBDTOP should be left unchanged, and LBOTON should be reset equal to LBDTOP.

These variables and indicators depend upon original data sources and should never be reset to labels contained within the interface routine itself.

4.5 COMMAND SEQUENCING

4.5.1 Command Sequencing File

The Command Sequence File is a direct access file with a structure as shown in Figure 35. 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 that 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.

4.5.2 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.

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

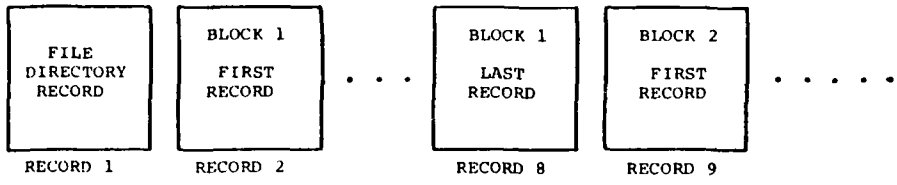


Figure 35. Structure of command sequence file.

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. Subrouting INTERP isolates any parameters passed to the command sequence and stores these parameters in the arrays NLARG and LARG of common block /LEDIT/. EDINP retrieves these parameters as they are referenced by the command sequence.

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 that 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.

4.6 MENUS

Menu displays are controlled by the routine MENU. This routine simply calls the appropriate routine to create the specified menu display. Following is a listing of these subroutines and the functions they perform.

MCOUNT - List the counters on the Master File partition or partitions that are currently accessed.

MITEMS - List the item codes for a given counter that are present on the currently accessed partition.

INFRED - List the Info File initial group and the geometric groups by name. For the first call to INFRED, use subroutine READGX to test the format of each geometric group for accuracy.

LSCRAT - List the contents of the scratch files.

EDITLS - List the Command Sequence blocks present on the Command Sequence file.

MPARTS - List the partitions on the Master File and indicate the partition(s) that is currently accessed.

MNMASK - List the item codes that are currently masked.

MENSET - List the current run settings.

4.7 GRAPHICS

4.7.1 Tektronix/Calcomp Plotting Interface

DATAMAP generates plots on the Tektronix 4014 screen that 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 PLOT and PLOTS routines supplied by Tektronix have been replaced by BHT modified versions for this specific application.

The plotting interface routines replace the functions of the Calcomp 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 that 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 four additional PLOT-10 routines are called by the modified PLOT and PLOTS routines. The modified PLOTS calls the routines INITT, TERM and CHRSIZ and PLOT calls MOVABS and DRWABS. The plotting interface calls the routines MOVABS, ANMODE, and SCURSR. Dummy versions of these last three routines are provided for the Batch mode load module. The PLOTS subroutine that is used for the Interactive Graphics mode controls the size and positioning of the plotting area on the Tektronix for the full-screen plotting mode. This routine sets multipliers and offsets as necessary for the plotting area in common block /CLCOMP/ and then PLOT uses these values in interpreting the plot commands on the Tektronix.

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.

4.7.2 X-Y Plots

Simple, multiple curve, double scale, and comparison X-Y plots are all generated through the routine XYPLOT. XYPLOT calls STPLT or STALL as necessary to initialize the plot frame. X and Y scaling values are determined using SCALEV and the axes

are drawn using AXES. For double-scale X-Y plots, subroutine DAXES is called instead, and this routine calls AXES twice. This portion of the code is skipped if a curve is being added to an existing plot frame.

LYNX is then called to draw the curve on the plot. For double-scale plots, DLYNX is called, which in turn call LYNX. 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.

4.7.3 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, while 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 input data matrix to obtain a new 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 or Z scale is set, CONTUR is called to draw the contour plot. CONTUR finds the sequences of X-Y positions that 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.

4.7.4 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.

4.8 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. During the second execution of FINDIT, the requested item code is compared with the list of masked item codes. If this item code matches a masked item code entry, FINDIT returns with an indication that the item code was not found. Otherwise, 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.

5. UTILITY ROUTINES

Certain subroutines are used in more than one of the DATAMAP programs. These routines have been written to be general in nature.

5.1 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
9 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 that 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 that 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 that perform initialization on the direct access files are listed in Section 6.1. 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 read the partition of the Master File that is specified for access using slot one.
2 = Directory for partition accessed through slot one
3 = SCF1
4 = SCF2
5 = Temporary Scratch
6 = Command Sequence File
7 = SCF3
8 = Partition accessed through slot two
9 = Directory for partition accessed through slot two.
10 = All of Master File

5.2 STRING HANDLING

Several routines are used by the system to process strings. Subroutine PACK transfers the leftmost character (i.e., the first character) 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 6.2).

NTOSTR is a routine that converts floating numeric values to strings. 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 that matched for the previous character. If, after every character of the input string has been processed, there is more than one keyword that 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. Thus, fewer than four characters may

be an acceptable entry to match a keyword, even though the matched keyword contains more characters than the test string supplied. If an unambiguous match is found for the test string before all the test characters have been processed, the remainder of the available test characters (up to the fourth character) are still compared and a mismatch will result in a return with IOUT= -1.

5.3 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, SORT0, 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, carrys 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.

5.4 SUBROUTINES TO ENHANCE TRANSPORTABILITY

Certain subroutines are used to concentrate non-standard code in a single location. These routines are described in Section 6 and are only listed here for completeness. The subroutines are QUIKI0, READLN, DATEQQ, INIDAF, PARMGT, and NEXTAP. The routines WMS, RMS, FMS, SHFSTR, and PACK also perform this function but they have additional reasons for use.

6. TRANSPORTABILITY CONSIDERATIONS

DATAMAP 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 that 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.

6.1 THE DATAMAP LIBRARY

As mentioned in Section 5.4, six subroutines are used to concentrate certain non-standard code in a single location. These six routines are contained in the "DATAMAP Library" or "DATMAPLB." DATMAPLB should be made available during the link-edit process for each of the DATAMAP programs. The six subroutines, DATEQQ, INIDAF, NEXTAP, PARMGT, QUIKIO, and READLN are all described in subsequent parts of Section 6.

6.2 DIRECT ACCESS

DATAMAP 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.

Four of the DATAMAP programs perform the initialization process of writing dummy data on each record of a direct access file. The initialization functions of the Master File Initialization Program and the Command Sequence File Initialization Program are clear from their titles. The File Creation Program initializes a direct access scratch file in the subroutine SETUP1. The Processing Program initializes scratch files in the subroutine INITSC. Subroutine INIDAF is called to perform each of the initializations. INIDAF writes on each direct access record in the sequential mode using subroutine FASTIO (see Section 6.4). In many cases, conversion to another initialization method could be accomplished by modification of INIDAF without change to the specific routines that call INIDAF.

6.3 CODING VARIATIONS

DATAMAP 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 that 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 data input that is not in DTF or BHT-GDC format.

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 DATAMAP to detect end files or errors on input.

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

All DATAMAP programs restrict use of this form to the subroutine READLN so that conversion of this statement type could be performed on this subroutine alone. READLN is one of the subroutines on DATMAPLB.

6.4 COMPUTER WORD PROBLEMS

Certain problems could be introduced in the conversion of DATAMAP to a new computer system from changes in size and format of the computer word.

6.4.1 String Storage and Processing

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, XYCMLB, 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 that can be stored in a word.

Special problems can arise in the interpretation of a string by arithmetic means. All of the character encoding methods that have been considered (BCDIC, EBCDIC, ASCII, and CDC internal display code) encode the digits 0-9 in a continuous sequence and in order from lowest unsigned arithmetic interpretation of a byte to highest (this is not true for the seven-track tape encoding of BCDIC). READF uses this assumption in the evaluation of numbers. READF and certain subroutines in the SYMBOL/NUMBER replacement package (see Section 6.6) also assume that the first character in a word is in the position of the most significant bits arithmetically when the word is evaluated as an integer. Just the reverse situation is true on the Digital Equipment Corporation (DEC) VAX 11/780, where the first character in a word occupies the least significant bits of the word. Alternate code is provided as commentary in the READF subroutine and in the SYMBOL/NUMBER replacement subroutines to correct this situation.

6.4.2 BHT-GDC Format Tape Processing

One of the options for the File Creation Program (FCP) is to read BHT-GDC format data input. This format includes data coded in IBM internal floating word format, IBM internal integer word format, and EBCDIC characters (see Reference 4). This mixture of data encoding methods is not a problem for DATAMAP versions installed on an IBM computer. However, for other computer systems, each BHT-GDC format input record may require conversion after input and before processing. The most logical point for insertion of such code is in the subroutine READD of the FCP. This conversion has been accomplished for the DEC VAX 11/780.

6.5 SPECIAL ROUTINES

Certain installation-provided routines are used in DATAMAP. 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.

DATAMAP uses subroutine FASTIO to avoid FORTRAN conversion routines for input and output using fixed length records. A second reason for using FASTIO is to read blocks that 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.

DATAMAP restricts all FASTIO calls to the subroutines QUIKIO and INIDAF. Both of these subroutines are in DATMAPLB.

DATAMAP calls the local BHT assembly language subroutine DATE to return the current Gregorian date. DATE returns the date in the format "mm/dd/yy." All calls to DATE are made through subroutine DATEQQ, which is on DATMAPLB. DATEQQ must return the date in an eight-character format but strict adherence to the format, "mm/dd/yy" is not necessary. For example, the DATE subroutine on the NASA Ames Research Center VAX 11/780 returns the date in a nine-character format (e.g., "28-MAY-80"). For the NASA installation, DATEQQ was rewritten to convert the above format and return an eight-character date (e.g., "28MAY'80).

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. DATAMAP uses the first eight characters that are returned and ignores the last four. Several of the DATAMAP programs call TIMOD. For conversion to another computer system, a replacement TIMOD subroutine should be written and inserted on DATMAPLB. This routine should call local routines to obtain the time of day and make format conversions as necessary. Alternatively, a dummy TIMOD subroutine can be written that always returns a blank array.

DATAMAP calls subroutines TIMEX and SETIME to monitor execution time in the Processing Program. For the BHT and USARTL (AVRADCOM, St. Louis, Missouri) installations, these subroutines are entries to the DATE subroutine. 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 DATAMAP application, except that a reasonably large number must be 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 DATAMAP. 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.

SETIME and TIMEX can be replaced with subroutines that call local CPU timing subroutines. Alternatively, these routines could be replaced with dummy subroutines. A dummy SETIME should have one argument and perform no action and a dummy TIMEX should have three arguments that are each set to zero on return. The replacement subroutines should be inserted on DATMAPLB.

Subroutine PLTIME is a special routine at BHT that 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. A do-nothing subroutine with no arguments may be substituted for PLTIME.

Subroutine CORE is a BHT-written routine for operation on IBM computers that allows FORTRAN format conversion of character strings stored in main memory. Most non-IBM installations have an equivalent FORTRAN capability that is invoked with the DECODE instruction. CORE is called with two arguments, the input character array to be converted and the number of characters in the array. The conversion is performed by a subsequent READ statement that specifies a dummy file reference number and the required format for conversion. The use of CORE is limited to subroutine CONVERT in the File Creation Program. Alternate code is provided in CONVRT to invoke DECODE, but this code is commented and thus inactive.

6.6 GRAPHICS

The graphic software was discussed extensively in Section 4.7. However, the graphics features related specifically to transportability of the code are discussed here. For non-interactive, off-line graphic output, the software assumes that the Calcomp routines PLOTS, and PLOT 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. The user may specify a larger increment by changing the default value for the variable PLTWID (in common block /MDEP/) or by 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 and the PLOT-10 software package are available. Substitute PLOT and PLOTS subroutines are provided for the interactive and interactive graphics modes of operation. These routines call Tektronix PLOT-10 subroutines to actually draw the individual vectors that compose a plot. In addition, the substitute PLOT and PLOTS subroutines can write a copy of each of the PLOT command arguments to file 22 for the COPY capability (see the SET/COPY and UTIL/COPY commands in Section 5 of Volume I).

A package of subroutines that emulates the Calcomp SYMBOL and NUMBER routines is provided with DATAMAP. This package is written entirely in FORTRAN, but the code does use the LOGICAL *1 data type extensively so that it may not be easily transportable. As mentioned in Paragraph 6.4.1, commented alternative code is included in these subroutines for installation on the DEC VAX 11/780. The QUICK plotting capability (see Section 5 of Volume I) for the interactive graphics mode of operation requires this package.

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 that provided the routines PLOT and PLOTS. The PLOTS routine, called once for each plot frame for the interactive graphics mode, must set up a simulated plot area of 9.5l vertical inches and 7.5l horizontal inches. The origin must be set initially at the lower left-hand corner of the simulated plot area. For the normal plotting mode (i.e., for the HALF plot setting), this entire simulated plot area must be mapped to the available screen area. For the FULL plot setting, the simulated plot area is expanded so that the plot annotation area is off the available screen area and the actual plot occupies as much of the available screen area as possible while retaining the proportion of the plot. This expansion is different for each type of plot and the proper expansion is communicated to PLOTS with the first argument. Following is a list of the allowed arguments, the type of plot, and the required plotting area.

1 - Normal plot area setting for HALF mode, 7.5l horizontal by 9.5l vertical.

2 - X-Y plot, 7.5l horizontal by 6.8l vertical.

3 - Cylindrical format contour plot, 7.5l horizontal by 7.07 vertical.

4 - Rectangular format contour plot, 7.5l horizontal by 6.43 vertical.

5 - Cylindrical format surface plot, 7.5l horizontal by 6.27 vertical.

6 - Rectangular format surface plot, 7.5l horizontal by 6.27 vertical.

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 that 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 screen cursor using subroutine SCURSR. SCURSR returns the cross-hair location in terms of the Tektronix raster locations. For a different graphics device, the graphics cursor function might be eliminated or some substitute for the cursor position evaluation might be found.

7. FILE AND LINKING REQUIREMENTS FOR DATAMAP PROGRAMS

This section describes the data sets that are required to link-edit each DATAMAP program and lists the FORTRAN file reference numbers that are referenced during the execution of each of these programs.

7.1 PROGRAM LINK INPUT REQUIREMENTS

Following is a listing of the data sets required to link all of the DATAMAP programs. For each program, one or more of these data sets will be required. The data set names are only for reference; the actual file or data set names may be different depending upon the computer installation.

DATMAPLB - This data set is a library of the subroutines described in Section 6.1, of any assembly language subroutines that are provided with DATAMAP, and of any special routines that must be provided for conversion of DATAMAP. For example, if a substitute TIMOD subroutine is written or provided, it should be included in this library.

PLOT10 This data set is a library of the Tektronix PLOT-10 subroutines. It is not provided with DATAMAP.

FORTLIB - This data set represents all subroutines provided by the computer system. It is not provided with DATAMAP.

CALCOMP - This data set represents either a library of CALCOMP subroutines or of CALCOMP emulation subroutines for an off-line plot device. It contains, as a minimum, the subroutines PLOT, PLOTS, SYMBOL, and NUMBER. It is not necessarily provided with DATAMAP.

NEWCPREV - This data set includes the object code for the substitute PLOT and PLOTS subroutines for Tektronix operation that were discussed in Section 6.6.

SYMBPACK - This data set includes the object code for the substitute routines for the CALCOMP SYMBOL and NUMBER subroutines as discussed in Section 6.6.

NPLDEVTO - This data set is an object module for a single subroutine called NPLDEV, which always returns a single argument equal to three (integer), which indicates Tektronix plot output.

NPLDEVCO - This data set is an object module for a single subroutine called NPLDEV, which always returns a single argument equal to one (integer), which indicates Calcomp plot output.

DUMMYS - The object code for the dummy subroutines ANMODE, MOVABS, SCURSR, and PLTIME.

CHRSIZ - Object module for the single dummy subroutine CHRSIZ (one argument).

INISYM - An object module for a single dummy subroutine called INISYM with no arguments.

INITL - This data set is the object code for most of the Master File Initialization Program.

EDINIT - This data set is the object code for most of the Command Sequence File Initialization Program.

MANAGE - This data set is the object code for most of the Master File Maintenance Program.

TYBTH - This data set is the object code for most of the Question and Answer Program to Create User Input for the File Creation Program.

FCP - This data set is the object code for most of the File Creation Program.

PROCESS - This data set is the object code for most of the Processing Program.

Table 4 lists the subset of the above-listed data sets required to link each DATAMAP program. The link instructions must adhere to the sequence of libraries specified in this table. Notice that there are several different configurations for the DATAMAP Processing Program.

7.2 PROGRAM RUN TIME FILE REQUIREMENTS

Each DATAMAP program accesses several Input/Output (I/O) files that must be satisfied either explictly or implictly (i.e., by default) by the computer system command language (e.g., IBM Job Control Language). Each file is referenced with a FORTRAN file reference number with the possible exception of the output

TABLE 4. PROGRAM LINKING REQUIREMENTS

Program	Required Input	
Master File Initialization Program	Input: Libraries:	INITL DATMAPLB FORTLIB
Command Sequence File Initialization Program	Input: Libraries:	EDINIT DATMAPLB FORTLIB
Master File Maintenance Program	Input: Libraries:	MANAGE DATMAPLB FORTLIB
Question and Answer Program to Create User Input Data Sets for the File Creation Program	Input: Libraries:	TYBTH DATMAPLB FORTLIB
File Creation Program	Input: Libraries:	FCP DATMAPLB FORTLIB
Processing Program (Tektronix interactive graphics version with QUICK plot mode)	Input: Libraries:	PROCESS NPLDEVTO NEWCPREV SYMBPACK DATMAPLB PLOT10 FORTLIB

TABLE 4. (Concluded)

Processing Program (Tektronic interactive graphics version without QUICK plot mode and without using the SYMBOL/ NUMBER package provided)	Input:	PROCESS NPLDEVTO NEWCPREV INISYM
	Libraries:	DATAMAPLB PLOT10 CALCOMP FORTLIB
Processing Program (Batch version using the SYMBOL/NUMBER package provided)	Input:	PROCESS NPLDEVCO DUMMYS CHRSIZ SYMBPACK
	Libraries:	DATMAPLB CALCOMP FORTLIB
Processing Program (Batch version without using the SYMBOL/NUMBER package provided)	Input:	PROCESS NPLDEVCO DUMMYS INISYM
	Libraries:	DATMAPLB CALCOMP FORTLIB

file for off-line graphics. Following is a listing of the files that must be satisfied for execution of each DATAMAP program together with an explanation of the required content and structure of each file. In the listing, record sizes are given as the actual data referenced by FORTRAN I/O statements. System appendages, if any, must be added to the record sizes.

Master File Initialization Program

01 - Master File, direct access disc file with 256-word records.

05 - System Input (e.g., cards or card images)

06 - System Output (e.g., line printer)

09 - Alias for file 01. The same data set as file 01 addressed sequentially. This file may not be required for non-IBM installations depending upon the rewritten INIDAF subroutine (see Section 6).

Command Sequence File Initialization Program

01 - Command sequence file, direct access disc file with 256-word records.

10
F

05 - System Input (e.g., cards or card images)

06 - System Output (e.g., line printer)

09 - Alias for file 01. The same data set as file 01 addressed sequentially. This file may not be required for non-IBM installation depending upon the rewritten INIDAF subroutine (see Section 6).

Master File Maintenance Program

01 - Master File, direct access disc file with 256-word records.

05 - System Input (e.g., interactive terminal or card images)

06 - System Output (e.g., interactive terminal or line printer.

07 - RESTORE Input. File containing a partition or Master File image. 256-word records read with FASTIO or replacement in QUIKIO (e.g., 256A4 formatted records).

08 - SAVE Output. A partition or Master File image may be written on this file. 256-word records written with FASTIO or replacement in QUIKIO (e.g., 256A4 formatted records).

09 - Sequential Scratch File. Disc file that will be used during movement of partitions on the Master File during a RESTORE operation. 256-word records written with FASTIO or replacement in QUIKIO (e.g., 256A4 formatted records). Space is required for 100 records.

Question and Answer Program (TYBTH)

05 - System Input (e.g., interactive terminal).

06 - System Output (e.g., interactive terminal).

08 - Instruction set. Card images of at least 72 characters.

File Creation Program (FCP)

01 - Master File, direct access disc file with 256-word records. 10 B

05 - System Input (e.g., card images)

06 - System Output (e.g., line printer)

12 - Direct Access Scratch File. Direct access disc data set with 256-word records. Space must be available for 1000 records.

13 - Sequential Scratch File. The structure of this file is different for DTF input processing and for BHT-GDC format input processing. For BHT-GDC format tape input, the record size is related to the processing of BHT-GDC format records. Thus, for systems that do not use 8-bit bytes and 4-byte words, the record size must be related to the actual handling of the data input records in subroutines READD and TRANSC. For IBM and similar systems, the records must be 6400 bytes in length. The records will be written and read

with FASTIO or appropriate replacement in QUIKIO (e.g., 10(160A4) formatted records). For DTF data input, the records must be 1020 words. These records are written and read using FORTRAN unformatted READS and WRITES. The number of records that are required for this file is difficult to predict in advance of any single job. About 150 records with a possibility of dynamic extension of the file should be sufficient.

14 - Alias for file 12. The same data set as file 12 addressed sequentially. This file may not be required for non-IBM installation depending upon the rewritten INIDAF subroutine (see Section 6).

20 - Time Alignment File. This file contains time alignment corrections for BHT-GDC format data input when the ALIGN option is specified in the user command input. For DTF input or for GDC format input without the ALIGN option, this file will not be referenced.

21-40 - Data input file(s). For DTF input, file 21 should be the DTF. Internal format DTF's are input with FORTRAN unformatted READ statements. 1024 words are read for each record. External format DTF's are input with FORTRAN formatted READ statements and 4096 characters are read for each record. For DTF input, files 22-40 are not referenced. At IBM installations, for BHT-GDC format input, the FCP references the first data tape as file 21 and each succeeding tape as a file reference number one higher than the previous file. Thus, if seven tapes are specified as input in the user command input, files 21 through 27 will be referenced.

Processing Program

01 - Master File, direct access disc file with 256-word records.

05 - System Input (e.g., card images or terminal)

06 - System Output (e.g., line printer or terminal)

11 - Scratch Files. Direct access disc file with 256-word records.

12 - Alias for file 11. The same data set as file 12 addressed sequentially. This file may not be required for non-IBM installations depending upon the rewritten INIDAF subroutine (see Section 6). This file is not referenced for a permanent scratch file that does not require initialization.

13 - Command Sequence Storage File. A direct access disc file with 256-word records.

14 - Info file. Card image format.

22 - Temporary plot copy file. Temporary disc file for storage of a single plot frame. Has three-word records that are written and read with FORTRAN unformatted WRITE and READ statements. This file is not referenced in the batch mode of operation or if the copy mode is not set in the interactive or interactive graphics modes of operation.

23 - Semi-permanent plot copy file. Disc file for storage of one or more plot frames until a batch job transfers the plots to an off-line graphics device. Has three-word records that are written and read with FORTRAN unformatted WRITE and READ statements. This file is not referenced in the batch mode of operation or if the COPY mode is not set in the interactive or interactive graphics modes of operation.

8. REFERENCES

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

2. Richard B. Philbrick, and Alfred L. Eubanks, OPERATIONAL LOADS SURVEY - DATA MANAGEMENT SYSTEM, Bell Helicopter Textron, USARTL Technical Report 78-52A and 52B, Applied Technology Laboratories, Fort Eustis, Virginia, January 1979, ADA065129 and ADA065270.

3. James R. Van Gaasbeek, and P. Y. Hsieh, ROTORCRAFT FLIGHT SIMULATION PROGRAM C81 WITH DATAMAP INTERFACE, Volumes I and II, Bell Helicopter Textron, USAAVRADCOM Technical Report 80-D-38A and 80-D-38B, Applied Technology Laboratory, U. S. Army Research and Technology Laboratories, Fort Eustis, Virginia.

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

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

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

/CNDATA/ Common block used to store information related to the counter for transfer of DTF data.

- IDFLTM - Flight number in 2A4 format
- CGROSW - Gross weight (REAL)
- CCENGU - Center of gravity (REAL)
- ICGCOD - Center of gravity code in A4 format.
- IMODEL - Model number in 2A4 format
- ISHPNM - Ship number in 2A4 format
- ICDATE - Flight date in 2A4 format
- ICTIME - Flight time in 3A4 format
- IMODCD - Model code in 2A4 format
- ICTEST - Test/Analytic Indicator (INTEGER). 0 = unspecified, 1 = test, 2 = analytic
- COFFST - Offset from start of data on DTF bundle sequence to first data value meant for tranfer.
- CRECLN - Number of seconds of data wanted for transfer from current bundle sequence.
- ICGRSW - Gross weight in 2A4 format.
- ICENGV - Center of gravity in 2A4 format.

/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.

/DTFINF/ Common block of general DTF record processing variables.

IDRTYP - Record type for current DTF record stored in IDTREC. Zero means no record present.

NDILIN - For instruction records, set to next available line of instructions.

NDILNS - For instruction records, set to number of available instruction lines in the record.

LNCONT - Set to true if a counter record has been read for the next data bundle sequence.

LNITEM - (LOGICAL) Set to TRUE. if an item code record or record sequence has been read for the next data bundle sequence.

LRAVAL - (LOGICAL) Set to .TRUE. if the DTF record currently stored in IDTREC has not yet been processed.

LDTEND - (LOGICAL) Set to .TRUE. if an end of file has been encountered on the DTF.

LALLIN - (LOGICAL) Set to .TRUE. when every DTF record encountered so far has been an instruction record.

NUMREC - (INTEGER) Sequential number of the current DTF record stored in IDTREC.

LDTFST - (LOGICAL) Set to .TRUE. if the first data bundle sequence has not yet been processed (so that both an item code and counter record must be read for processing).

LDTIME - (LOGICAL) Set to .TRUE. if a time base field has been processed for the next bundle sequence.

LBYPAS - (LOGICAL) Set to .TRUE. if none of the counters and/or none of the item codes from the next bundle sequence are desired for transfer.

ITBTYP - (INTEGER) Time base type for next data sequence on DTF. 1 = FSRTSI = sample interval 2 = FSRTSI = sample rate. 3 = ITBITM = item code for variable sample rate time base channel.

FSRTSI - (FLOATING) Constant sample interval or sample rate as specified by ITBTYP

ITBITM - (A4) Item code for variable sample rate as indicated by ITBTYP.

/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 Input File (=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)

/INFILE/ DTF input Info File information

NINFRC - Scratch file offset for the next Info File group to be processed.

NGRUPS - Number of Info File groups currently stored.

IGRUPS - Two-dimensional array, which serves as a directory of the Info File groups that are stored.

IGRUPS(1,N) = Group Name (A4)

IGRUPS(2,N) = Scratch file offset for the start of this group.

IGRUPS(3,N) = Number of columns.

IGRUPS(4,N) = Number of rows.

IGRUPS(5,N) = Number of double-row elements.

/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.

LOCO - Not used

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 that should be skipped before data

from a particular item code are 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 that 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).

IRDATS - Not used

IRSAUS - Not used

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.

MSETN - Sequence number in Master File directory of the partition currently being processed.

/ITMDAT/ Common block of item code related variables for DTF input processing.

NMTCHS - Number of item codes in the current bundle that have been requested for transfer by the user instruction (Init-DTFITM)

ITIMEB - Sequential position in the bundle of the time base item code. (Init-DTFITM)

FILMIN - Minimum roll-off point of the digital filters selected for the current bundle. (Init-DTFITM)

NBUNSZ - Number of items in the current bundle. (Init-DTFITM)

FMODLO - Modulo specifications for each of the selected items in the current bundle. Subscript is from sequence of selected item codes only. The same subscript is used for ITEMW and ITEMTP. (Init-DTFITM)

ITMTRN - List of keys indicating whether or not a selected item code has been transferred from scratch to the master file. 0 = not transferred, 1 = transferred. Subscript is same as for FMODLO, ITEMW, and ITEMTP.

/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.

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.

LDTF - Logical variable. True if data input is in DTF format. False otherwise.

NLOOK - Product of the number of item codes and the number of counters requested for transfer.

LDTFIN - Logical variable. True if user instructions are to be read from the DTF. False otherwise.

LTHERE - Logical variable. True if FRSTLN has read a line of user input into ICHAR and has left the line for further processing by INLIST.

LEXTRN - Logical variable. True if DTF input is in external format. False otherwise.

LALLIT - Logical variable. True if all item codes that occur on a DTF are wanted for transfer. False otherwise.

LALLCN - Logical variable. True if all counters that occur on a DTF are wanted for transfer. False otherwise.

LSCAN - Logical variable. True if only a scan of DTF input for format validity without transfer of data is desired. False otherwise.

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).

RATE - Specified sample rate for storage of data from a DTF.

/LOCOM/ Information for data transfer process.

ITEMN - Current item code to be transferred.

ICNTRN - 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.

/SCRAT/ This common block specifies a general scratch area.

KDUMMY - General scratch array.

/SIZES/ General constants stored as variables for use throughout the program.

MAXCN - Maximum number of counters that may be specified for transfer in one run of the File Creation Program.

IDAFWD - Number of words in a record in the direct access files assessed by the File Creation Program.

INBYTE - Number of bytes in a GDC format tape input block.

CTRMAX - Maximum size entry for a counter.

INFSPC - Number of scratch direct access records reserved for prototype initial records of time histories.

IDAFBT - Number of bytes in a record in the direct access files accessed by the File Creation Program.

KDSIZE - Size of the scratch common array KDUMMY in /SCRAT/

MAXIT - Maximum number of item codes that may be specified for transfer to the Master File.

NCREAD - Number of character positions to be examined by READF for character input. Characters other than blanks may not be allowed towards the end of the line.

IBIG - Large integer for use as limit for DO Loop where exit from the loop will be by a decision jump.

NCHARS - Largest character position in an instruction input line that may contain a non-blank character.

ISZDSR - Size in words of a sequential scratch record for DTF data transfer.

LACCSZ - Size of the LACCUM array

LTRPSZ - Not used

IFLBSZ - Size of the area in KDUMMY allocated to the pre-filter accumulators.

NTRPSZ - Size of the area in KDUMMY allocated to pre-interpolation accumulators.

MIFGRI - Number of groups that may be specified in an internal format DTF record.

MIFGRE - Number of groups that may be specified in an external format DTF record.

MXIFGP - Max number of info file groups that may be specified in one DTF.

MXIFRC - Max number of rows and/or columns that may be specified for an info file group.

MXDASR - Highest direct access scratch record that can be used for temporary storage of info file data during DTF data transfer.

INCLMS - Largest character position in an Info File output line that can be non-blank.

/TMPRCM/ Common block of values used for transfer of data records from DTF to scratch file.

LSTRTD - (LOGICAL) TRUE if a bundle corresponding to the required starting offset is found. Relavent for initial transfer of data to scratch file. (Init-CPYTSC, Set-TMPRSM)

TOFFST - Required starting offset. Before LSTRTD = .TRUE., this is the total offset from the start of data on the DTF to the first required data. When LSTRTD = .TRUE., this is the offset excluding time in the scratch records that are skipped. (Init-CPYTSC, Set-TMPRSM)

NBNSKP - Number of bundles in skipped scratch records for TMPRSM. (Init-CPYTSC, Set-TMPRSM)

TFRSTP - Initial time for previous record processed by TMPRSM. (Init-CPYTSC, Set-TMPRSM)

TMAX - Max time to copy to the scratch file excluding skipped records. (Init-TMPRSM)

TSAVED - Time stored on scratch file excluding skipped records. (Init-TMPRSM)

LCMPLT - (LOGICAL) .TRUE. if all necessary data is saved on the scratch file already. (Init-CPYTSC, Set-TMPRSM)

TCORR1 - For variable sample rate data, the time for the first bundle before correction. (Init-TMPRSM)

TSUBTR - Time to subtract from variable sample rate time instants during transfer of data to the scratch file.

LNODAT - (LOGICAL) .TRUE. if no data record is available in IDTREC. (Init-CPYTSC, Set-CPYTSC)

LENDAT - (LOGICAL) .TRUE. if the end of the input DTF data is encountered. (Init-CPYTSC, Set-PRFORM)

NRPRVI - For DTF input with data bundles greater than the capacity of a data record. Number of records that have been processed for the current bundle.

LOCREC(8) - Location on the input bundles of the data from the item codes in the current set. (Init-SETUP)

ITEMFD(8) - Item code for each position in the current set. (Init-SETUP)

LOCL(8) - Starting record offsets on the Master File partition for the data streams corresponding to each item code. Actual locations given are the record positions for the info records. (Init-SETINF)

VLAST(8) -Last data value for each time history in the current set. Used to extend the time histories to satisfy convolution filter edge requirements. 11 B

LACCUM(2048) - Array that contains data records to be written to the Master File partition. The first record starts at subscript 1 and each succeeding record starts at a subscript IDAFWD larger.

/TRNCOM/ General common variables for transfer at DTF data to Master File.

NORECS - Number of data records to be written to the Master File for each item code in the current set of item codes being transferred from scratch.

NITSET - Number of item codes in the current item code set for transfer from scratch to the Master File.

NLEADR - Number of points required before and after the data interval to be transferred for the current convolution filter.

IAFLSZ - Size of the pre-filter accumulators in KDUMMY for each item code in the item code set.

LASFLT - Position of the last filtered data value in each pre-filter accumulator.

LASDAT - Position of the last unfiltered data value in each pre-filter accumulator.

NSKIP - Not used

NUMCRR - Number of values in each data accumulator buffer (LACCUM) for transfer to the Master File.

NBUNDL - Number of bundles in current bundle sequence excluding bundles on records to be skipped.

NBUND1 - Not used

NBUND2 - Not used

NSRSKP - Number of scratch records to skip before data to be processed is encountered.

TINC - Sample interval for data to be written to the pre-filter buffers. (Init-CPYTSC)

TLEADR - Equivalent amount of time required for filter initialization for the current set of item codes to be transferred from scratch.

TINCMN - Sample interval for data to be written to the pre-interpolation buffers. (Init-CPYTSC)

XOFFST - Time offset from first data point required for any kind of processing (i.e., transfer or filtering) to first data point required for transfer.

NBUNSR - Max number of bundles stored in a scratch file record.

LENDSC - (LOGICAL) True if the end of the scratch file has been encountered for the current transfer from scratch.

IRCRAW - Number of records read from the scratch file in the current pass.

MXNTRP - Maximum number of values that can be accommodated by the pre-interpolation buffers.

TNEXT - Time for next value to be placed in pre-filter buffer by interpolation (Init-SETIME, Set-INTRPG)

FILTRT - Low-pass filter break-frequency for item code set to be transferred from scratch. (Init-SETUP)

FMODLX - Modulo value for item code set to be transferred from scratch (Init-SETUP)

ISKIPR - Skip rate specified by user for current item code set. A positive value specifies that the rate was not specified. (Init-SETUP)

ISKIPX - Skip rate to be applied to points in the pre-filter buffer by the filtering or unfiltered skipping process. (Init-SETUP)

NRESLT - Number of interpolated data values in each interpolated data buffer.

NPTOUT - Number of data points to be generated for each item code in the current transfer. (Init-SETUP)

NXNTRP - Next relative point location to be added in the pre-interpolation array - 'INTRPR' (Init-SETAPE, Set-INTRPG)

NPRAW - Number of points in the raw data input buffer. (Init-SETAPE, Set-INTRPG)

NPT - Bundle number from the current scratch input record that is being processed from the pre-filter buffers.

NEXTND - Number of points that must be simulated as values equal to the last available value and appended to the end of available data to satisfy the edge requirement for the convolution filter (Init-SETUP)

NEXSTR - Number of points that must be simulated as values equal to the first available value and appended to the start of available data to satisfy the edge requirement for the convolution filter (Init-SETUP)

NBPROS - Bundle number in the bundle sequence that is currently being processed from the scratch file.

/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.

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.

AZMGRP - Azimuth offset specified by Info File geometric group.

NPRCRA - Partition access slot pseudo file (i.e., 1 or 9) for the partition source for the current attached parameter data.

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 three-quarters 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.

/CLCOMP/ Common block used exclusively by the PLOT and PLOTS emulation package for the Tektronix, and by the PLOC subroutine.

XMULT - Multiplier used to convert simulated horizontal plot paper position in inches to horizontal raster position on the Tektronix screen.

YMULT - Multiplier used to convert simulated vertical plot paper position in inches to vertical raster position on the Tektronix screen.

XACUM - Horizontal offset to plot origin in plot paper coordinates.

YACUM - Vertical offset to plot origin in plot paper coordinates.

ILEFT - Horizontal raster position on the Tektronix screen that corresponds to the initial horizontal paper position of 0.0.

IBOTT - Vertical raster position on the Tektronix screen that corresponds to the initial vertical paper position of 0.0.

/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.

IPRTWO - Set to one if process must have two input data streams. Set to one otherwise.

ISCFIP - Not used

NFREE - Source of input data. Allowed values:

1 = SCF1
2 = SCF2
3 = SCF3
4 = Info file group specifies item code(s)
5 = User specified item code
6 = Info file specifies item code required for derivation by keyword
7 = 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.

LROWP - Not used

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.

M12INP - 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 column 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.

IEPROS - Set to one for ensemble averaging. Set to zero otherwise.

/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.
9 - Comparison plot
10 - Double scale plot

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

1 = SCF1
2 = SCF2
3 = SCF3

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.

NMAXOT - Maximum number of output values in the first dimension.

OUTXMX - Maximum first dimension scale value that occurs for the function that is being processed.

OUTXMN - Minimum first dimension scale value that occurs for the function that is being processed.

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

OUTMX2 - Maximum second double-row element dependent variable value for all rows and columns. Only set if both double-row elements are processed.

OUTMN2 - Minimum second double-row element dependent variable value for all rows and columns. Only set if both double-row elements are processed.

/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.

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.

SRATE - 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.

NPRMOD - Indicator of partition access slots that are in use. 1 = first slot taken, 2 = second slot taken, 3 = both slots taken.

NPRCRN - Indicator of source of data in ICTRD array. 1 = first slot partition, 2 = second slot partition, 0 = no data in ICTRD.

NMASK - Number of masked item codes listed in array MASKIT.

NMPARI - Name of partition addressed through the first access slot. Stored as 2A4.

NMPAR2 - Name of partition addressed through the second access slot. Stored as 2A4.

MASKIT - Array of masked item codes each stored as a 4-character name. The first NMASK array locations are occupied.

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.

/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 provisional user command directives and comment for use in the user interface while command is developing.

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 COMMENT command 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.

LRL - Not used

LHL - Not used

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.

MODFUL - Set to one for Tektronix full-screen mode, set to zero otherwise.

DPLOFF - Vertical offset for second set of axes using DPLOT option. Set to this offset while curves are being drawn on the second set of axes. Set to zero otherwise.

IDPLOF - Set to one if two sets of axes are present. Set to zero otherwise.

12
F

NCCX - Set to zero for a continuing curve. Set to one if the next curve segment that is drawn will start fresh.

ISEQ - Current sequence position for the curve that is being drawn in the dash-dot sequence specified by JUNQ.

DONE - Amount of the current dash or gap that has been drawn in the current curve.

ZLENX - Width in screen or paper units (e.g., inches) of each dash or gap in the currently used dash-dot sequence.

/DRW2/ Common for double-scale plots (DPLOT option).

YMIN22 - Equivalent top axes variable (for the DPLOT option) to the YMIN variable when the DPLOT option is not used.

DY22 - Equivalent to DY for the top axes when DPLOT is used.

YH22 - Equivalent to YH for the top axes when DPLOT is used.

YL22 - Equivalent to YL for the top axes when DPLOT is used.

INSL22 - Equivalent to INSL for the top axes when DPLOT is used.

YPLUS2 - Vertical offset from lower set of axes to upper set when DPLOT is used.

XL22 - Equivalent to XL for the top axes when DPLOT is used.

/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). 12 B

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, SCF3, 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.

NPTM - File for temporary storage of the last plot frame drawn when the COPY mode was set.

NPCP - File for storage of plot frame copies for later replotting on an off-line plot device.

/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, Bl, 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.

NMCURR - Not used.

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. Currently not set or used.

ITIME - Time of day the data stream was recorded. The format is 2A4. Currently not set or used.

ICLABL - Current counter in string format A4,A2.

ITEML - Item code in format A4.

LUSQRD - Indicator for modification of input units labe(s) to achieve correct output units labels. Values are

-3 = cross process with normalized units.

-2 = cross process with units per herz.
-1 = cross process
 0 = no change
 1 = square first units label
 2 = square units label per hertz

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.

IUNIT2 - Unit label for second input function for processes that require two inputs (e.g., cross-correlation).

ITEMD2 - Data label for second input function for processes that require two inputs (e.g., cross-correlation).

/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.

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

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

LNPREC - Number of command lines that 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.

LARGCH - Set to character that specifies an execution argument or parameter. Currently set to 1H%.

LMXARG - Maximum number of arguments that can be passed during execution of a command sequence.

MXARGC - Maximum number of characters that can be passed in one command sequence argument.

NLNTMP - Number of characters available on the temporary character line, LNTEMP.

NLARG - Array. Each variable contains the number of characters stored for the corresponding index of LARG.

LARG - Two-dimensional array of parameter inputs. Second index corresponds to NLARG index. String is stored over multiple values of the first index.

/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 - Variable indicating whether the scratch files are already initialized. If IDONE = 1, then the files were initialized before the program run began. Otherwise, the files must be initialized at the start of the run.

/MENBUF/ Buffer block for menu generation

IX - Array for generation of menus

/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.

LTOPON - Set to one if a top double-row element origin label is present in LBDTOP. Set to zero otherwise.

LBOTON - Set to one if a bottom double-row element origin label is present in LABBOT. Set to zero otherwise.

LBDTOP - Label for origin of top double-row element, if any. 5A4 format.

LABBOT - Label for origin of bottom double-row element, if any. 5A4 format.

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.

MODEZ - 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

IQUICK - Set to one for QUICK plot mode. Set to zero for SLOW plot mode.

ICPSET - Set to one for COPY mode. Set to zero for NOCOPY mode.

ICPRLG - Set to one if the COPY mode has ever been set in the current run. Set to zero otherwise.

NCPRES - Number of PLOT call records stored on the temporary plot storage file. Set to zero if no frame has been stored or if NOCOPY mode is set.

/PRCOM/ Common for process communication.

KOUNTR - Current counter stored as an integer

KITEM - Current item code stored in A4 format.

TIMOFF - Not used

RECLEN - Not used

TINT - Sample interval

NPTS - Number of points in output record.

XSTRTV - Starting independent variable value.

XINTVL - Independent variable sampling interval.

NMAXVL - Number of samples in processing record.

PMINOR - Not used

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.

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.

/PARMS/ Common to hold parameters passed to the program from the computer system command language.

IPARCN - Number of characters that were transferred.

IPARMS - Array to hold the characters that were passed.

/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 through 9999.

ULABLS - First independent variable unit labels.

/PLSPCL/ Common for control of the special plotting modes QUICK and COPY.

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.

/SCRTBL/ Block for storage of directory blocks of the input and/or output scratch files.

ISCOIN - For scratch file output. Set to zero if the scratch file has not been addressed (SCADD call) with a data stream that was not missing. Set to one if the scratch file has been addressed with a data stream that was not missing.

ISCLIM - For scratch file output. Set to zero if the scratch file has not been addressed with a data stream that was not missing for the current column. Set to one otherwise.

XSCRT - Array for storage of directory blocks of the input and/or output scratch files. The first half of XSCRT holds directory data for the designated output scratch file (if any). The second half of XSCRT holds directory data for the designated input scratch file.

/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.

INITBG - Indicates whether scratch files have been initialized. Set to zero if not. Set to scratch file size in records if so.

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 persent.

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:

13
F

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.

NUNIN - Number of input unit conversion specifications that are currently stored in common (the input unit conversion capability is not yet implemented).

NUMOUT - Number of output unit conversion specifications that are currently stored in common.

MOREIN - Info File line number where more input unit conversion specifications begin (the input unit conversion capability is not yet implemented). Set to zero if none.

MOROUT - Info File line number where more input unit conversion specifications begin. Set to zero if none.

NGRUPS - Number of group names and starting lines numbers stored in IGRUP and IGRPLC.

MORGRP - Info File line number for the line just before the first geometric group that is not recorded in IGRUP and IGRPLC. Thus, MOREGRP is the number of Info File lines that are skipped before the next Info File group is read.

MSCAND - Initially zero. Reset to one at the end of the first successful call to the INFRED program. This variable indicates whether the Info File geometric groups have been scanned for correct format with a MENU/INFO/ call.

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.

IGRUP - Array of geometric group names in the Info File that have corresponding locations stored in IGRPLC. The group names are contiguous at the bottom of IGRUP with unused space at the

top. The four-character group names are stored in A4 format.

IGRPLC - Array of group locations in the Info File that correspond by index to the group names in IGRUP. The values given are the number of Info File lines that should be skipped to reach the first line of an Info File group.

UNCNSL - Array of unit conversion slope factors. Factors correspond by index to the unit labels in IUNCNV.

UNCNIN - Array of unit conversion Y-intercept terms. Terms correspond by index to the unit labels in IUNCNV.

IUNCNV - Array of unit conversion unit labels. The second index corresponds to the indices of UNCNSL and UNCNIN. The first index is dimensioned to six. The first three values correspond to the original unit label. The last three values correspond to the converted unit label.

/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.

NCNRWl - 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.

MXRCHR - Number of characters that are read during an input from system input, the Info File, or the Command Sequence Storage File.

MAXGRP - The maximum number of geometric group names and locations that can be stored in the arrays IGRUP and IGRPLC.

MXUNCN - The maximum number of unit conversion specifications that can be stored in the arrays UNCNSL, UNCNIN, and IUNCNV.

MXMASK - The maximum number of masked item codes that can be stored in the array MASKIT.

TSTINF - Test value that is used for comparison with a datum value to determine whether the datum value indicates the data are missing. A value less than TSTINF indicates data are missing. TSTINF must be set greater than SETINF. Currently, TSTINF is not used throughout the program.

SETINF - Prototype data missing value. When data are missing for a time history, the first datum value should be set to SETINF. SETINF must be less than TSTINF.

/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

/SMPNTR/ Common used for the SYMBOL/NUMBER emulation package.

LOCSYM - Array of pointers set by INISYM. A character code (e.g., under EBCDIC) is interpreted as an unsigned integer. The integer is used as an index and the value of LOCSYM for that index is the pointer to the correct character in the list of allowed characters. A pointer of zero indicates there is no allowed character for the code.

/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.

CLBPOS - Used for comarison plot option (LPLOT). This variable is the vertical position of the top horiztonal line that will form part of the box around the annotation for the current curve.

IDBLPT - Set to zero normally. Set to one if a double-scale plot (DPLOT) is being drawn or was the last plot frame drawn.

/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.

/VSIZE/ Common block used exclusively for the Versatec plotting adaptation of DATAMAP.

VXUSED - Inches of plot that have been used so far in current Versatec "frame."

VERMAX - Maximum number of inches that can be plotted before a new Versatec "frame" must be started.

/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)

FOLLOWING ARE SEVERAL EXAMPLES OF IBM JOB CONTROL LANGUAGE (JCL) SEQUENCES AND IBM TSO COMMAND LANGUAGE CLISTS TO BE USED TO COMPILE, LINK-EDIT, AND EXECUTE THE VARIOUS DATAMAP PROGRAMS. THESE EXAMPLES ARE NOT COMPLETE AND SECTION 7 OF THIS VOLUME SHOULD BE CONSULTED FOR A COMPLETE DESCRIPTION OF FILE AND LINK-EDIT REQUIREMENTS. THESE EXAMPLES DO GIVE DEMONSTRATIONS OF HOW VARIOUS REQUIREMENTS THAT ARE SPECIFIED IN SECTION 7 ARE ACTUALLY ACCOMPLISHED USING JCL OR CLISTS. THESE EXAMPLES HAVE BEEN CONSTRUCTED TO WORK ON BHT'S COMPUTER INSTALLATION OF IBM 370/166'S RUNNING UNDER MVS. SOME CHANGES WILL BE REQUIRED FOR OPERATION ON OTHER IBM SYSTEMS.

EXAMPLE 1 - TO EXECUTE THE FILE CREATION PROGRAM TO READ BHT-GDC FORMAT INPUT DATA TAPES.

```
//ESAREB JOB (FEAB(200,G3E,63(999,DP3E,TS),'DICK  2841',
//    MSGLEVEL=1,MSGCLASS=A,CLASS=T,NOTIFY=ESAR
/*SETUP          DSN=ENGR.NEWOLS
/*SETUP          DSN=ENGR.F1653966
/*SETUP          DSN=ENGR.FE352194
/*SETUP          DSN=ENGR.F3196024
/*SETUP          DSN=ENGR.F73(7461
/*SETUP          DSN=ENGR.F2460312
/*SETUP          DSN=ENGR.F4280703
/*SETUP          DSN=ENGR.F6045285
//STEP1 EXEC PGM=FEAR02,TIME=20
//STEPLIB DD DSN=ENGR.TEST1,DISP=SHR
//FT(1F(C1 DD DSN=ESAR.OLSMAS,DISP=OLD
//FT(5F(C1 DD DSN=ESAR.FCP1NPUT.DATA,DISP=SHR
//FT(6F(C1 DD SYSOUT=A
//FT14F(C1 DD UNIT=SYSDA,DSN=ESAR.UNIT14,DISP=NEW,
//   SPACE=(CYL,3),DCB=(RECFM=F,BLKSIZE=1024,DSORG=DA)
//FT12F(C1 DD DSN=ESAR.UNIT14,UNIT=AFF=FT14F(C1,
//    VOL=REF=*.FT14F(C1,DISP=OLD,
//    DCB=(RECFM=F,BLKSIZE=1(24,DSORG=DA)
//FT13F(C1 DD UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(3,2)),
//    DCB=(RECFM=F,BLKSIZE=64(0)
//SYSUDUMP DD SYSOUT=A
//FT2(F(C1 DD UNIT=TPS,DSN=ENGR.NEWOLS,DISP=OLD
//FT21F(C1 DD UNIT=AFF=FT2(F(C1,
//     DSN=ENGR.F1653966,DISP=OLD
//FT22F(C1 DD UNIT=AFF=FT2(F(C1,
//     DSN=ENGR.FE352194,DISP=OLD
//FT23F(C1 DD UNIT=AFF=FT2(F(C1,
//     DSN=ENGR.F3196024,DISP=OLD
//FT24F(C1 DD UNIT=AFF=FT2(F(C1,
//     DSN=ENGR.F73(7461,DISP=OLD
//FT25F(C1 DD UNIT=AFF=FT2(F(C1,
//     DSN=ENGR.F2460312,DISP=OLD
//FT26F(C1 DD UNIT=AFF=FT2(F(C1,
//     DSN=ENGR.F4280703,DISP=OLD
//FT27F(C1 DD UNIT=AFF=FT2(F(C1,
//     DSN=ENGR.F6045285,DISP=OLD
//
```

EXAMPLE 2 - TO EXECUTE THE FILE CREATION PROGRAM WITH DATA TRANSFER FILE (DTF) FORMAT DATA INPUT. IN THIS CASE, THE DATA INPUT IS A DISC FILE THAT HAS BEEN PREVIOUSLY CREATED. NOTICE THAT THE RECORD FORMAT FOR THE TEMPORARY SEQUENTIAL SCRATCH FILE (FT13F001) HAS CHANGED AND THAT AN INFO FILE GROUP OUTPUT FILE DATA SET IS DECLARED (FT09F001).

```
//ESARC9 JOB (FEABC20C,G3B,630999,DP3B,TS),'DICK  2841',
//    MSGLEVEL=1,MSGCLASS=A,CLASS=T,NOTIFY=ESAR
//STEP1 EXEC PGM=FTARC2,TIME=20
//STEPLIB DD DSN=ENGR.TEST1,DISP=SHR
//FT01F001 DD DSN=ESAR.OLSMAS,DISP=OLD
//FT05F001 DD DSN=ESAR.DTFSTUB,DISP=SHR
//FT06F001 DD SYSOUT=A
//FT09F001 DD UNIT=SYSDA,DSN=ESAR.IN171020,
//    DISP=(NEW,CATLG),SPACE=(TRK,(6,6)),
//    DCB=(RECFM=FB,BLKSIZE=3120,LRECL=80)
//FT14F001 DD UNIT=SYSDA,DSN=ESAR.UNIT14,DISP=NEW,
//  SPACE=(CYL,3),DCB=(RECFM=F,BLKSIZE=1024,DSORG=DA)
//FT12F001 DD DSN=ESAR.UNIT14,UNIT=AFF=FT14F001,
//    VOL=REF=*.FT14F001,DISP=OLD,
//    DCB=(RECFM=F,BLKSIZE=1024,DSORG=DA)
//FT13F001 DD UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(3,2)),
//  DCB=(RECFM=VBS,BLKSIZE=6400,LRECL=4084)
//SYSUDUMP DD SYSOUT=A
//FT21F001 DD DSN=ESAR.DTF.DATA,DISP=SHR
//
```

EXAMPLE 3 - TO EXECUTE THE PROCESSING PROGRAM IN A BATCH MODE. NOTICE THAT TWO DATA SETS ARE CONCATENATED TO FORM THE INFO FILE. ALSO NOTICE THAT THE SCRATCH FILES DATA SET IS ASSUMED TO EXIST ALREADY AND THAT THE SCRATCH FILES ARE DECLARED TO BE PERMANENT WITH THE PASSED PARAMETER 'PERM'.

```
//ESARSC JOB (ATAR0600,G3B,6309900,DP3B,TS),'DICK  2841',
//  MSGLEVEL=1,MSGCLASS=A,CLASS=H,NOTIFY=ESAR
// EXEC PGM=ATARC6,TIME=20,PARM='PERM'
//STEPLIB DD DSN=ENGR.TEST1,DISP=SHR
//FT12F001 DD DSN=ESAR.SCRATCH.DATA,DISP=OLD
//FT11F001 DD DSN=ESAR.SCRATCH.DATA,DISP=OLD
//FT05F001 DD DSN=ESAR.PROCESIN.DATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//FT01F001 DD DSN=ESAR.OLSMAS,DISP=SHR
//FT14F001 DD DSN=ESAR.INFOBASE,DISP=SHR
//          DD DSN=ESAR.CB1GRUPS.DATA,DISP=SHR
//FT13F001 DD DSN=ESAR.COMSEQ,DISP=SHR
//PLOTTAPE DD UNIT=(TPS,,DEFER),
//    VOLUME=PRIVATE,DCB=(DEN=3),
//    LABEL=EXPDT=98005
//
```

```
     EXAMPLE 4 - TO EXECUTE THE PROCESSING PROGRAM UNDER TSO.
     FOR THIS EXAMPLE, A TEMPORARY SCRATCH FILES DATA SET IS
     ALLOCATED AND THE PARAMETER 'TEMP' IS PASSED TO THE
     PROGRAM.  THE PLOT COPY DATA SETS ARE DECLARED AS DUMMYS
     SO THAT NO PLOT COPIES SHOULD BE ATTEMPTED USING THIS
     CLIST.

PROC 2 EDIT INFO PROG(ENGR.PROD1(ATAR03)) MFILE(ESAR.OLSMAS)
CONTROL NOMSG
FREE DA('ESAR.CLIST')
FREE F(FT01F001 FT05F001 FT06F001 FT22F001 FT23F001)
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
FREE ATTR(SCRATX)
ALLOC F(FT12F001) DA(VTEMP) SHR
ALLOC F(FT11F001) DA(VTEMP) SHR
ALLOC F(FT13F001) DA('&EDIT.') SHR
ALLOC F(FT01F001) DA('&MFILE.') SHR
ALLOC F(FT14F001) DA('&INFO.') SHR
ALLOC F(FT05F001) DA(*)
ALLOC F(FT06F001) DA(*)
ALLOC F(FT22F001) DUMMY
ALLOC F(FT23F001) DUMMY
CALL '&PROG.' 'TEMP'
FREE F(FT01F001 FT22F001 FT23F001)
FREE F(FT11F001 FT12F001 FT13F001 FT14F001)
DELETE VTEMP


     EXAMPLE 5 - TO EXECUTE THE PROCESSING PROGRAM UNDER TSO.
     IN THIS EXAMPLE, THE SCRATCH FILES ARE ASSUMED TO BE A
     PERMANENT DATA SET AND TO BEEN INITIALIZED BY A PREVIOUS
     PROCESSING PROGRAM RUN.  THE PARAMETER 'PERM' IS PASSED TO
     THE PROGRAM.  ALSO, THE PLOT COPY DATA SETS ARE ALLOCATED.

PROC 2 EDIT INFO PROG(ENGR.PROD1(ATAR03)) MFILE(ESAR.OLSMAS)
CONTROL NOMSG
FREE DA('ESAR.CLIST')
FREE F(FT01F001 FT05F001 FT06F001 FT22F001 FT23F001)
FREE F(FT11F001 FT12F001 FT13F001 FT14F001)
CONTROL MSG
ALLOC F(FT12F001) DA(VPERM) SHR
ALLOC F(FT11F001) DA(VPERM) SHR
ALLOC F(FT13F001) DA('&EDIT.') SHR
ALLOC F(FT01F001) DA('&MFILE.') SHR
ALLOC F(FT14F001) DA('&INFO.') SHR
ALLOC F(FT05F001) DA(*)
ALLOC F(FT06F001) DA(*)
ATTR VCOP RECFM(V,B,S) DSORG(PS) LRECL(16) BLKSIZE(3124)
ALLOC DA(TEMPCOP) USING(VCOP) NEW SPACE(1) CYL
ALLOC F(FT22F001) DA(TEMPCOP)
ALLOC DA(KEEPCOP) USING(VCOP) NEW SPACE(1,1) CYL
ALLOC F(FT23F001) DA(KEEPCOP)
FREE ATTR(VCOP)
CALL '&PROG' 'PERM'
FREE F(FT01F001 FT22F001 FT23F001)
FREE F(FT11F001 FT12F001 FT13F001 FT14F001)
DELETE TEMPCOP
```

EXAMPLE 6 - TO PLOT FROM THE PLOT COPY FILE THAT WAS CREATED IN EXAMPLE 5.

```
//ESAR01 JOB (PLOTCOPY,G38,63099900,DP38,TS),'DICK  2841',
//         MSGLEVEL=1,MSGCLASS=A,CLASS=G,NOTIFY=ESAR
//     EXEC PGM=FEAR28,TIME=5
//STEPLIB DD DSN=ENGR.TEST1,DISP=SHR
//FT23F001  DD  DSN=ESAR.KEEPCOP,DISP=SHR
//FT06F001 DD SYSOUT=A
//PLOTTAPE DD UNIT=(TPS,,DEFER),
//            VOLUME=PRIVATE,LABEL=EXPDT=98005,
//            DCB=(DEN=2)
//
```

EXAMPLE 7 - TO RUN THE FILE MAINTENANCE PROGRAM UNDER TSO. THE SAVE, RESTORE AND SCRATCH FILES ARE NOT DECLARED SO THE THE 'SAVE' AND 'RESTORE' COMMANDS SHOULD NOT BE USED WITH THIS CLIST.

```
PROC 0 PROG(ENGR.TEST1(FEAR99)) MFILE(ESAR.OLSMAS)
CONTROL NOMSG
FREE DA('ESAR.CLIST')
FREE F(FT01F001 FT05F001 FT06F001)
FREE ATTR(SCRATX)
CONTROL MSG
ALLOC F(FT01F001) DA('&MFILE.') SHR
ALLOC F(FT05F001) DA(*)
ALLOC F(FT06F001) DA(*)
CALL '&PROG.'
FREE F(FT01F001)
```

EXAMPLE 8 - TO EXECUTE THE FILE MAINTENANCE PROGRAM IN A BATCH MODE. THE SAVE, RESTORE, AND SCRATCH FILES ARE DECLARED IN THIS JCL SO THAT THE SAVE AND/OR RESTORE COMMANDS COULD BE USED.

```
//ESAR20 JOB (MAINTAIN,G38,63099900,DP38,TS),'DICK  2841',
//     MSGLEVEL=1,MSGCLASS=A,CLASS=G,NOTIFY=ESAR
//   EXEC PGM=FEAR99,TIME=20
//STEPLIB DD DSN=ENGR.TEST1,DISP=SHR
//FT01F001 DD DSN=ESAR.OLSMAS,DISP=OLD
//FT05F001 DD DSN=ESAR.FMPINPUT.DATA,DISP=SHR
//FT06F001 DD SYSOUT=A
//FT07F001 DD DSN=ESAR.PARTSAVE,UNIT=TPS,DISP=OLD
//FT08F001 DD DSN=ESAR.PARTSAV1,UNIT=TPS,DISP=NEW,
//            LABEL=EXPDT=98365,
//            DCB=(RECFM=FB,BLKSIZE=8192,LRECL=1024)
//FT09F001 DD UNIT=SYSDA,DISP=NEW,SPACE=(CYL,(3,2)),
//            DCB=(RECFM=FB,BLKSIZE=4096,LRECL=1024)
//
```

EXAMPLE 9 - TO EXECUTE THE QUESTION AND ANSWER PROGRAM TO CREATE COMMAND INPUT FOR THE FILE CREATION PROGRAM (TYBTH). THIS CLIST ALLOCATES THE COMMAND INPUT FILE AS 'NEW' SO THE FILE SHOULD NOT EXIST BEFORE THIS CLIST IS EXECUTED. THE DATA SET NAME THAT SHOULD BE USED FOR THIS FILE MUST BE PASSED AS THE FIRST PARAMETER TO THE CLIST.

```
PROC 1 INPSET PROG(ENGR.PROD1(FEAR55))
CONTROL NOMSG
FREE DA('ESAR.CLIST')
FREE F(FT05F001 FT06F001)
FREE F(FT08F001)
FREE ATTR(SCRATX)
CONTROL MSG
ATTR SCRATX RECFM(F,B) LRECL(80) BLKSIZE(2480)
ALLOC DA(&INPSET.) SP(10) TRACKS USING(SCRATX)
FREE ATTR(SCRATX)
ALLOC F(FT08F001) DA(&INPSET.)
ALLOC F(FT05F001) DA(*)
ALLOC F(FT06F001) DA(*)
CALL '&PROG.'
FREE F(FT08F001)
```

EXAMPLE 10 - TO EXECUTE THE COMMAND SEQUENCE FILE INITIALIZATION PROGRAM UNDER TSO. THIS COMMAND SEQUENCE FILE DATA SET MUST BE ALLOCATED BEFORE THIS CLIST IS EXECUTED AND THE DATA SET NAME MUST BE PASSED AS THE FIRST PARAMETER TO THE CLIST.

```
PROC 1 EDITFILE
CONTROL NOMSG
FREE DA('ESAR.CLIST')
FREE F(FT01F001 FT05F001 FT06F001 FT09F001)
CONTROL MSG
ALLOC F(FT09F001) DA('&EDITFILE.') SHR
ALLOC F(FT01F001) DA('&EDITFILE.') SHR
ALLOC F(FT05F001) DA(*)
ALLOC F(FT06F001) DA(*)
CALL 'ENGR.TEST1(FEAR1)'
FREE F(FT01F001 FT09F001)
```

EXAMPLE 11 - TO COMPILE ANY SOURCE. FOR THIS EXAMPLE, THE SOURCE INPUT IS ON DATA SET 'ESAR.SOURCE.FORT' AND THE OBJECT DECK IS WRITTEN TO DATA SET 'ESAR.OBJECT.OBJ'.

```
//ESAR4   JOB (COMPIL00,G38,6309900,DP38,TS),'DICK  2841',
//        MSGLEVEL=1,NOTIFY=ESAR,MSGCLASS=A,TIME=(5,00),CLASS=D
//FORTX   EXEC PGM=IFEAAB,REGION=256K
//SYSPRINT DD SYSOUT=$
//STSTERM 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.OBJECT.OBJ
//SYSIN DD DSN=ESAR.SOURCE.FORT,DISP=(OLD,DELETE)
//
```

EXAMPLE 12 - TO LINK THE FILE CREATION PROGRAM ON TSO. THE OBJECT DECK IS ASSUMED TO BE ESAR.OBJECT.OBJ. THE LOAD MODULE IS LEFT ON THE PARTITION 'FEAR02' OF THE LIBRARY 'ENGR.PROD1'.

```
LINK OBJECT LOAD('ENGR.PROD1(FEAR02)') MAP FORTLIB LIB('ESAR.DATMAPLB'
'ENGR.FORTLIB')
```

EXAMPLE 13 - TO LINK THE BATCH VERSION OF THE PROCESSING PROGRAM ON TSO. THE MAJOR PART OF THE PROCESSING PROGRAM OBJECT IS 'ESAR.OBJECT.OBJ', WHILE THE ADDITIONAL OBJECT DECKS 'ESAR.SYMBPACK.OBJ', 'ESAR.NPLDEVCO.OBJ', AND 'ESAR.DUMMYS.OBJ' ARE USED (SEE SECTION 7). IN ADDITION, THE LIBRARIES 'ESAR.DATMAPLB' AND 'ENGR.FORTLIB' ARE USED. 'ENGR.FORTLIB' CONTAINS THE CALCOMP PLOTTING SUBROUTINES

```
LINK (OBJECT SYMBPACK NPLDEVCO DUMMYS) LOAD('ENGR.TEST1(ATAR06)') MAP
SIZE(400000,100000) FORTLIB LIB('ESAR.DATMAPLB' 'ENGR.FORTLIB')
```

EXAMPLE 14 - TO LINK THE INTERACTIVE VERSION OF THE PROCESSING PROGRAM ON TSO. THE MAJOR PART OF THE PROCESSING PROGRAM IS OBJECT IS IN 'ESAR.OBJECT.OBJ', WHILE THE ADDITIONAL OBJECT DECKS 'ESAR.SYMBPACK.OBJ', 'ESAR.NEWCPREV.OBJ', AND 'ESAR.NPLDEVTO.OBJ' ARE USED (SEE SECTION 7). IN ADDITION, THE LIBRARIES 'ESAR.DATMAPLB', 'ENGR.TCSLOAD1', AND 'ENGR.FORTLIB' ARE USED. 'ENGR.TCSLOAD1' CONTAINS THE TEKTRONIX PLOT10 LIBRARY.

```
LINK (OBJECT SYMBPACK NEWCPREV NPLDEVTO) LOAD('ENGR.PROD1(ATAR05)') MAP
SIZE(400000,100000) FORTLIB LIB('ESAR.DATMAPLB' 'ENGR.TCSLOAD1' 'ENGR.F
ORTLIB')
```


DATE
FILMED
8