United States
Department of
Agriculture

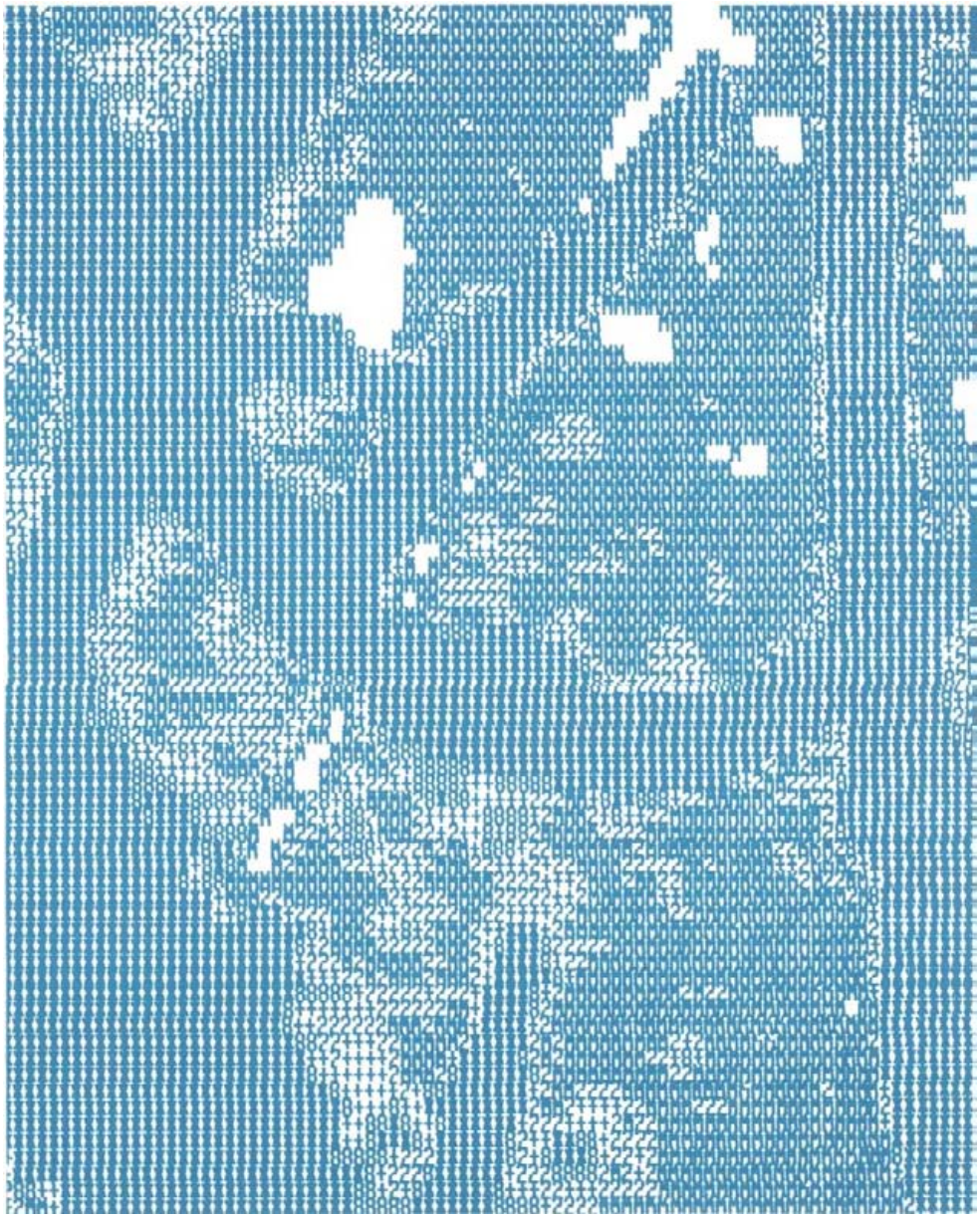
Forest Service

**Pacific Southwest
Forest and Range
Experiment Station**

General Technical
Report PSW- 53

SCANIT: centralized digitizing of forest resource maps or photographs

Elliot L. Amidon E. Joyce Dye



The Authors:

were formerly assigned to the Station's research unit investigating measurement and analysis techniques for management planning, with headquarters in Berkeley, Calif. **ELLIOT L. AMIDON** is now assigned to the Station's unit at Arcata, Calif., that is studying the management of Pacific coastal forests on unstable lands. He earned a bachelor's degree in forest management at Colorado State University (1954) and a master's degree in agricultural economics at the University of California, Berkeley (1961). **E. JOYCE DYE** is now a computer systems analyst with the Station's statistics and computer services group, at Berkeley. She earned a bachelor's degree in geography (1973) at the University of California, Berkeley.

Publisher:

Pacific Southwest Forest and Range Experiment Station
P.O. Box 245, Berkeley, California 94701

June 1981

SCANIT: centralized digitizing of forest resource maps or photographs

Elliot L. Amidon E. Joyce Dye

CONTENTS

SCANIT is a computerized technique for encoding in digital form the spatial data present on wildland resource maps or aerial photographs. These data can be processed further by a geographic information ("computer mapping") system (Amidon 1978). The basic documents are sent by users having diverse applications to a central location and automatically raster-scanned on a microdensitometer. In accordance with user specifications, digital data are summarized, edited or modified, and returned on magnetic tape. SCANTT can provide a partial processing or preprocessing facility for scanner-based systems with large throughput requirements, such as the Wildland Resource Information System (WRIS) or the RID*POLY Geographic Information System. These systems use maps photographically reduced onto a negative for automatic digitizing (Russell and others 1975, Deschene 1981). Another major application, that of remote sensing, requires the acquisition of densities for processing by statistical programs to discriminate between vegetative types. The raster scanning service is similar to photofinishing in that the cost of a central facility can be amoritized over many users.

- **SCANIT can scan either maps or aerial photographs**

Each of the two applications mentioned earlier has its own processing requirements. The primary purpose of SCANIT is to acquire input for a geographic information system. Typically, the negative is of a map containing lines, such as polygon boundaries, to be sliced at one density level into lines and background. The search may be more complicated than in the case of multidensities because the desired threshold is not obvious. All densities below a specified threshold (on a negative) constitute a line. In the second, typical application—remote sensing—the objective is some form of pattern recognition. The user wants multiple density levels on an aerial photograph to be scanned and classified by integer intervals.

- **SCANIT allows work to be performed at a central facility to vary by user needs**

The amount of work done at a central facility may vary by user needs. Users with minicomputers may choose to process the raw scanner output themselves. Another alternative is to provide the central facility with guidelines and let an experienced scanner operator exercise judgment.

The amount of work accomplished by the facility is governed by varying the arrangement of three main programs. All three programs could be used on the requestor's minicomputer, which would virtually eliminate processing instructions to a central operator. Program SLICE groups densities in various ways for later analysis, FREAK provides a frequency distribution and various statistics, and LOOKED provides for display and editing of the scan data.

- **SCANIT's interactive system minimizes the need for a user manual**

In a typical batch processing environment, forms are encoded and keypunched to provide data processing control, usually at a remote site. Comparable control is achieved more easily with the interactive system provided. The computer provides both instructions and queries as the situation demands. The keystroke responses are often binary, either yes or no, developing in a logical manner (*fig. 1*). This direct approach is possible because work experience obtained during development is embedded in the computer programs that control the flow of work.

- **SCANIT's program maintenance manual minimizes search time**

Although little written information is needed to operate the system, a substantial amount is required to maintain it. The program maintenance manual is provided in a rigid format to minimize search time. The approach chosen is that of "Guidelines for Documentation of Computer Programs and Automated Data Systems," Federal Information Processing Standards Publication 38 (U.S. Dep. of Commerce 1976). One of a series by the National Bureau of Standards, the guidelines are distributed under the provisions of Part 6 of Title 15, Code of Federal Regulations.

In addition to general information, the manual provides detailed descriptions of the 22 routines (main and subprograms) developed. Subprograms are related to each other and to their main programs. Finally, variables are defined and referenced to the programs that use them.

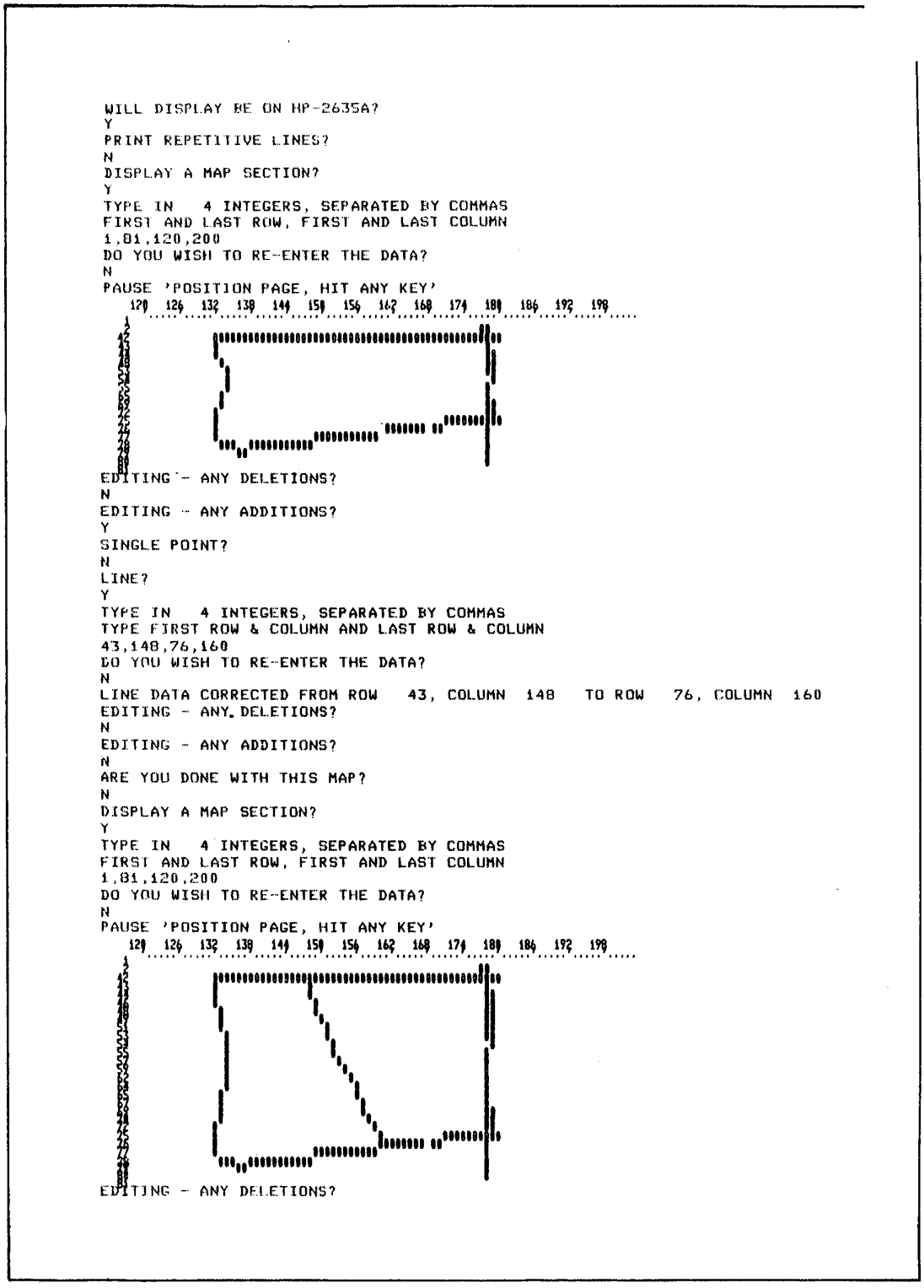


Figure 1—Neither portion of binary map data are from the original scan. The upper "picture" is compressed, in comparison with the original, because repetitive scan lines have been deleted. After inserting a line, the data are redisplayed. Note the "stretching" effect because fewer scan lines are repetitive than before.

1. GENERAL INFORMATION

The automated digitizing process may start with a negative or positive photo-reduction of a map, or a negative or diapositive of an aerial photograph. A scanning microdensitometer will record thousands of densities from the input image. Because millions of densities can be produced each day, a minicomputer needs efficient utility routines for data transfer and manipulation.

The user may simply want all 256 density levels recorded, or may have them thresholded into binary, zero-one categories. Alternatively, the densities may be divided into groups for the user to apply some pattern recognition algorithm. This manual provides the means of maintaining a centralized digitizing and partial processing (or preprocessing) service.

HARDWARE

The system consists of a scanner, a minicomputer, a magnetic tape unit, a fixed disc drive, a demountable disc drive, and a terminal. Although specific brand names are identified, alternatives for each item are available from other manufacturers.[1]

Our current scanner is a Joyce Loebl Scandig Model 3 drum type microdensitometer. The system's central processing unit is an Eclipse S/ 130, manufactured by Data General (DG). We selected a 32K word memory (16-bit words). The DG disc 14234H has one fixed and one removable disc cartridge for 10 MM bytes of total storage and with a data transfer rate of 250K bytes per second. The DG magnetic tape unit 6021/6023 accommodates an 800-BPI, 9-track, 2400-foot reel, and has a transfer rate of 60K bytes per second. Terminals include a printer-plotter, a cathode ray tube (CRT), and a tri-mode printer, but almost any keyboard-display combination may be used.

SUPPORT SOFTWARE

Our programs use the manufacturer's I/O routines, bit accessing, and standard FORTRAN functions controlled by the Real Time Disk Operating System (RDOS). Modular program structure facilitates conversion to other operating systems. A disc operating system is required along with loaders, assemblers, and FORTRAN compiler and libraries. A text editing capability is desirable. The SCANIT computer programs developed by us will be copied on a magnetic tape, supplied by the requestor.

The computer programs described in this publication are available on request with the understanding that the U.S. Department of Agriculture cannot assure their accuracy, completeness, reliability, or suitability for any other purpose than that reported. The recipient may not assert any proprietary rights thereto nor represent them to anyone as other than Government-produced computer programs.

[1] Trade names and commercial enterprises or products are mentioned solely for information. No endorsement by the U.S. Department of Agriculture is implied.

ADDITIONAL INFORMATION

The additional options available to a user can be specified by completing the service request form. Permissible combinations of options and constraints are controlled by the software and become evident during interactive use of the system.

The maintenance programmer will find the glossary essential. More exact information is available in the program listings. The extensive cross-referencing of programs and variables will help trace decision paths through the complex system structure. The 22 programs are listed alphabetically and each is described according to the guidelines provided in the outline for maintenance manuals cited earlier (U.S. Dep. of Commerce 1976, p. 46, 47).

The processing times for the most frequently encountered operations are not significant for planning purposes. A run involving a large number of frequency groups for an entire image could consume 2 hours or more. Usually, a small portion of the negative can be selected to provide sufficient information. Program SLICE was timed on an image of a quarter-million pixels, divided into 10 density groups. This representative case required one-half hour to process.

1.1. Service Request Form

A sample service request form can be modified to suit an individual requestor's need (*fig. 2*). Complex processing problems may require attachment of supplementary instructions.

1.2. System Routines Used

Twenty-two routines were developed for this system and 23 others are part of the Data General library (*fig. 3*).

1.3. Glossary of Variables

The variables used are defined in the glossary (*fig. 4*).

1.4. Cross-referencing of Programs and Variables

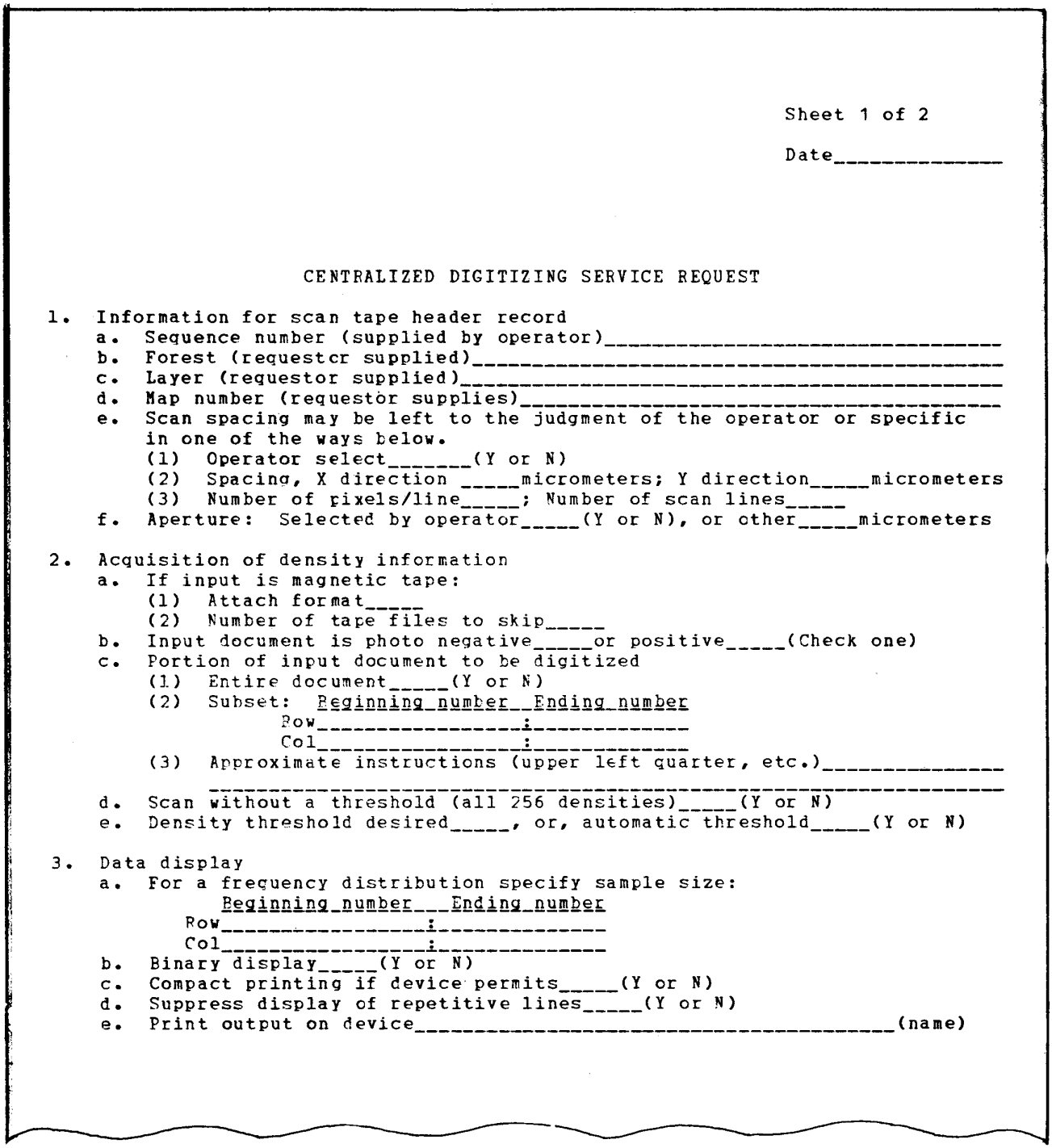
Tracing program execution through more than 40 subprograms is facilitated by a cross-referencing tool (*fig. 5*).

If the glossary definition of a variable is inadequate, refer to its use in the program code. A cross-reference table indicating the variable is provided (*fig. 6*).

1.5. System Modification and Conversion

The system described can be adapted to function on another minicomputer. To assist conversion with nonstandard features and extensions, specialized information is supplied on the FORTRAN used and on the operating system routines.

a. COMPILER NOSTACK. This option causes the compiler to allocate local variables in fixed locations rather than on



Sheet 1 of 2

Date_____

CENTRALIZED DIGITIZING SERVICE REQUEST

1. Information for scan tape header record
   a. Sequence number (supplied by operator)_____
   b. Forest (requestor supplied)_____
   c. Layer (requestor supplied)_____
   d. Map number (requestor supplies)_____
   e. Scan spacing may be left to the judgment of the operator or specific in one of the ways below.
      (1) Operator select_____(Y or N)
      (2) Spacing, X direction _____micrometers; Y direction_____micrometers
      (3) Number of pixels/line_____; Number of scan lines_____
   f. Aperture: Selected by operator_____(Y or N), or other_____micrometers

2. Acquisition of density information
   a. If input is magnetic tape:
      (1) Attach format_____
      (2) Number of tape files to skip_____
   b. Input document is photo negative_____or positive_____(Check one)
   c. Portion of input document to be digitized
      (1) Entire document_____(Y or N)
      (2) Subset: Beginning number Ending number
          Row_____:_____
          Col_____:_____
      (3) Approximate instructions (upper left quarter, etc.)_____
      _____
   d. Scan without a threshold (all 256 densities)_____(Y or N)
   e. Density threshold desired_____, or, automatic threshold_____(Y or N)

3. Data display
   a. For a frequency distribution specify sample size:
      Beginning number Ending number
      Row_____:_____
      Col_____:_____
   b. Binary display_____(Y or N)
   c. Compact printing if device permits_____(Y or N)
   d. Suppress display of repetitive lines_____(Y or N)
   e. Print output on device_____(name)

Figure 2—Scanning and manipulation of density data are initiated by a single document.

Sheet 2 of 2

Date_____

f. Multilevel display
(1) Equal-sized groups
a) Low end of range (density)_____, or default_____
b) High end of range (density)_____, or default_____
c) Number of groups_____, or default_____
d) Default group symbols_____(Y or N), or specific symbols for the number of groups specified in c. above_____

(2) Unequal-sized groups
a) Low end of range (density)_____
b) High end of range (density)_____
c) Number of groups_____
d) Specify up to 64 pairs of group edges--(use a continuation sheet if necessary).

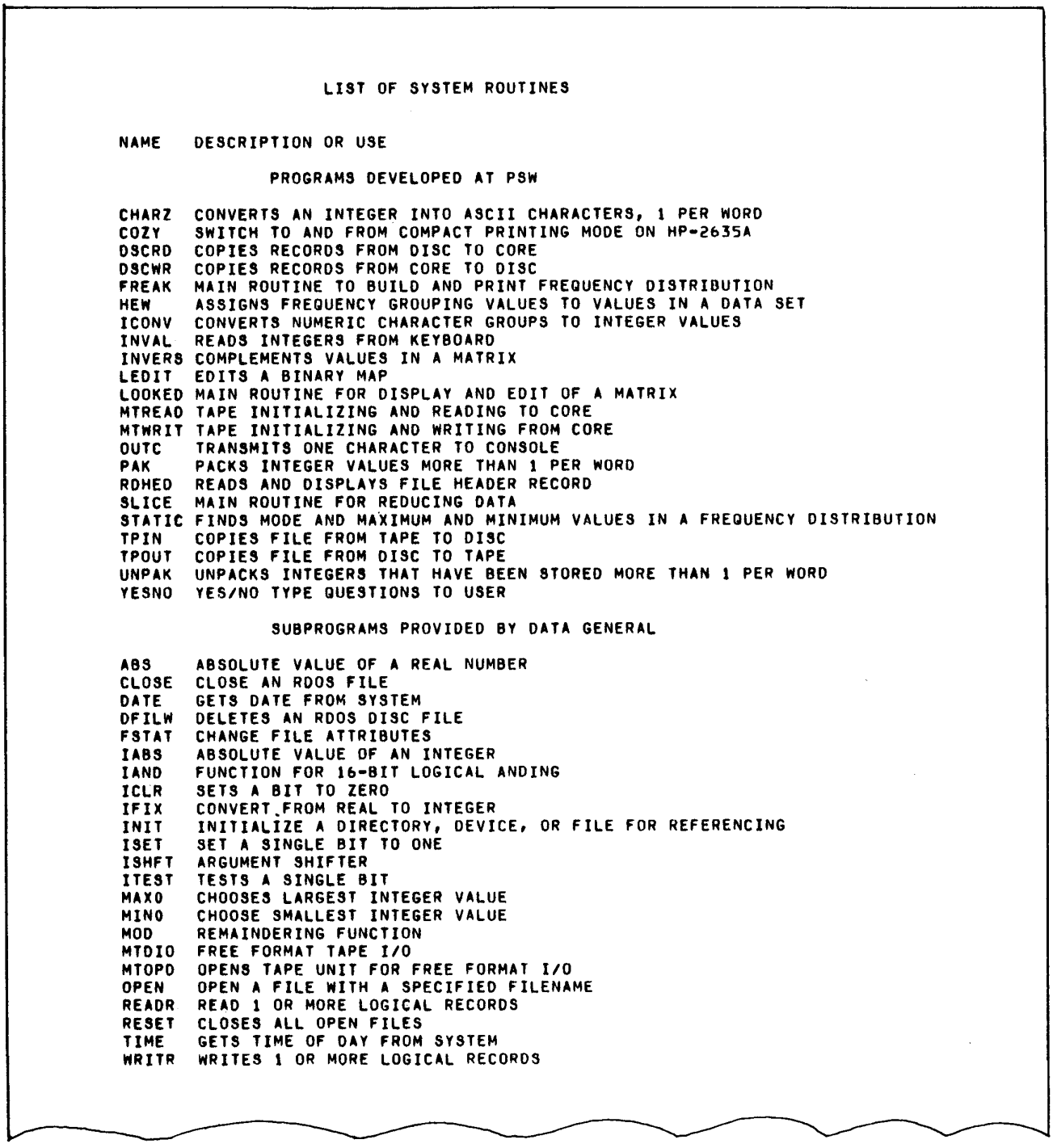
	1	2	3	4	5	6	7	8	9	10
Low edge for each group										
Symbols for the above										

4. Editing: Attach data editing guidelines unless the work is to be done by the requestor.

5. Disposition of output
a. Return original document to sender_____(Y or N)
b. Return input negative to sender_____(Y or N)
c. Return input tape to sender_____(Y or N)
d. Send output tape to requestor with format or other description_____ (Y or N)
e. Name and address_____
_____
_____

Telephone_____or_____
Date required_____
Insurance, special mail instructions_____
_____

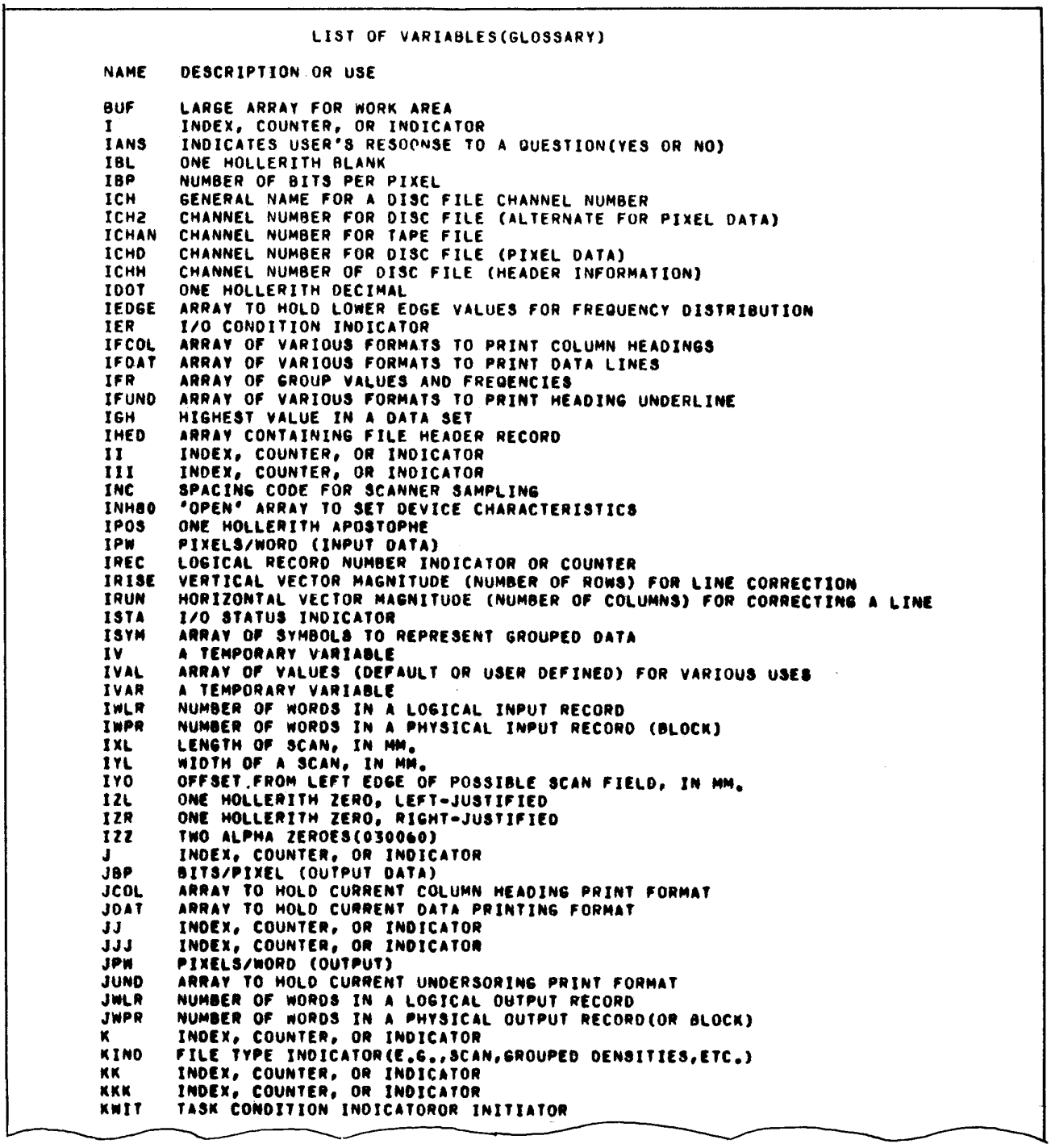
Figure 2—Scanning and manipulation of density data are initiated by a single document—Continued.



LIST OF SYSTEM ROUTINES

NAME	DESCRIPTION OR USE
	PROGRAMS DEVELOPED AT PSW
CHARZ	CONVERTS AN INTEGER INTO ASCII CHARACTERS, 1 PER WORD
COZY	SWITCH TO AND FROM COMPACT PRINTING MODE ON HP-2635A
DSCRD	COPIES RECORDS FROM DISC TO CORE
DSCWR	COPIES RECORDS FROM CORE TO DISC
FREAK	MAIN ROUTINE TO BUILD AND PRINT FREQUENCY DISTRIBUTION
HEW	ASSIGNS FREQUENCY GROUPING VALUES TO VALUES IN A DATA SET
ICONV	CONVERTS NUMERIC CHARACTER GROUPS TO INTEGER VALUES
INVAL	READS INTEGERS FROM KEYBOARD
INVERS	COMPLEMENTS VALUES IN A MATRIX
LEDIT	EDITS A BINARY MAP
LOOKED	MAIN ROUTINE FOR DISPLAY AND EDIT OF A MATRIX
MTREAD	TAPE INITIALIZING AND READING TO CORE
MTWRIT	TAPE INITIALIZING AND WRITING FROM CORE
OUTC	TRANSMITS ONE CHARACTER TO CONSOLE
PAK	PACKS INTEGER VALUES MORE THAN 1 PER WORD
RDHED	READS AND DISPLAYS FILE HEADER RECORD
SLICE	MAIN ROUTINE FOR REDUCING DATA
STATIC	FINDS MODE AND MAXIMUM AND MINIMUM VALUES IN A FREQUENCY DISTRIBUTION
TPIN	COPIES FILE FROM TAPE TO DISC
TPOUT	COPIES FILE FROM DISC TO TAPE
UNPAK	UNPACKS INTEGERS THAT HAVE BEEN STORED MORE THAN 1 PER WORD
YESNO	YES/NO TYPE QUESTIONS TO USER
	SUBPROGRAMS PROVIDED BY DATA GENERAL
ABS	ABSOLUTE VALUE OF A REAL NUMBER
CLOSE	CLOSE AN RDOS FILE
DATE	GETS DATE FROM SYSTEM
DFILW	DELETES AN RDOS DISC FILE
FSTAT	CHANGE FILE ATTRIBUTES
IABS	ABSOLUTE VALUE OF AN INTEGER
IAND	FUNCTION FOR 16-BIT LOGICAL ANDING
ICLR	SETS A BIT TO ZERO
IFIX	CONVERT FROM REAL TO INTEGER
INIT	INITIALIZE A DIRECTORY, DEVICE, OR FILE FOR REFERENCING
ISET	SET A SINGLE BIT TO ONE
ISHFT	ARGUMENT SHIFTER
ITEST	TESTS A SINGLE BIT
MAX0	CHOOSES LARGEST INTEGER VALUE
MIN0	CHOOSE SMALLEST INTEGER VALUE
MOD	REMAINDERING FUNCTION
MTDIO	FREE FORMAT TAPE I/O
MTOPD	OPENS TAPE UNIT FOR FREE FORMAT I/O
OPEN	OPEN A FILE WITH A SPECIFIED FILENAME
READR	READ 1 OR MORE LOGICAL RECORDS
RESET	CLOSES ALL OPEN FILES
TIME	GETS TIME OF DAY FROM SYSTEM
WRITR	WRITES 1 OR MORE LOGICAL RECORDS

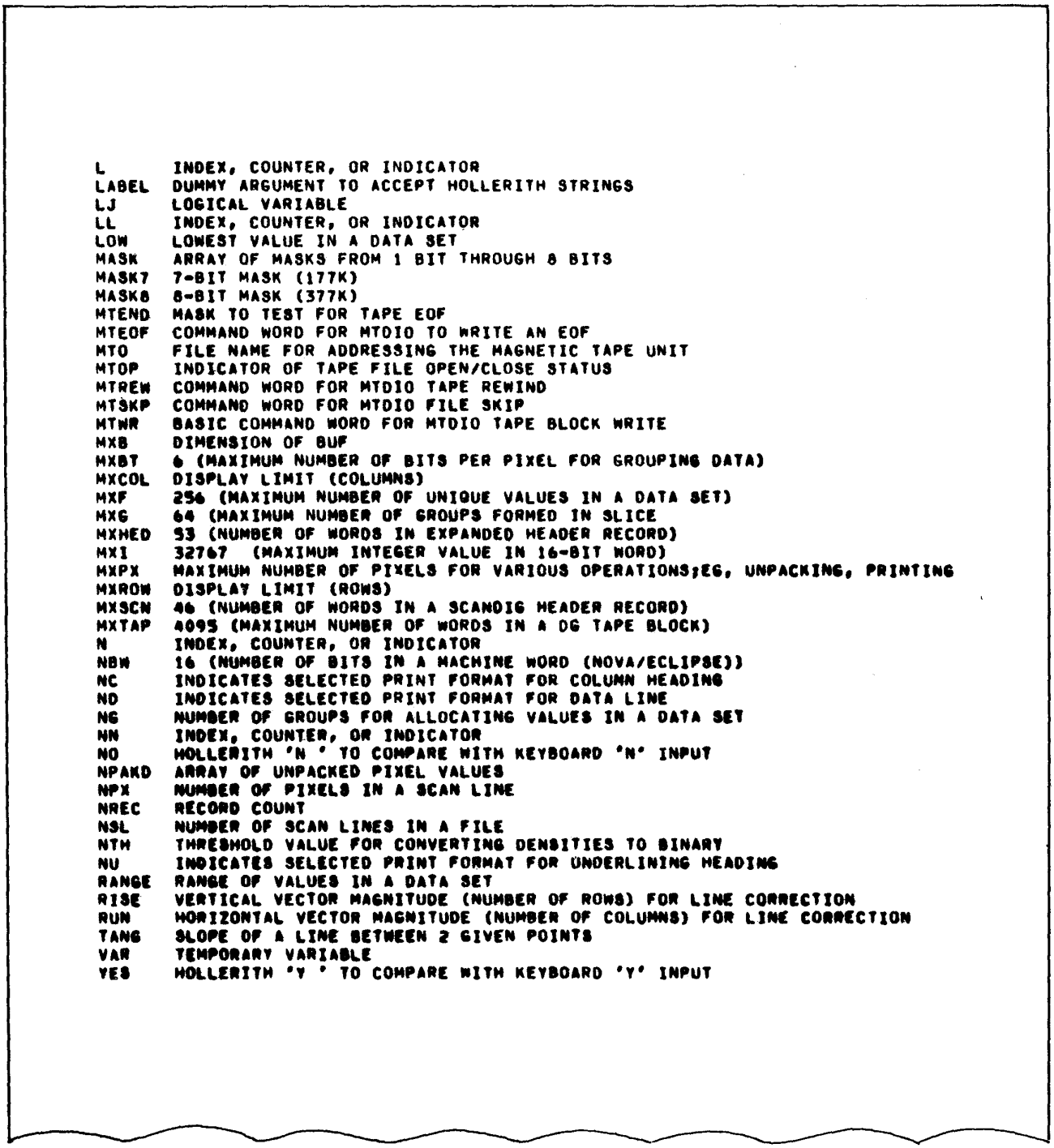
Figure 3—The system requires 45 routines of which one-half were developed expressly for the digitizing service.



LIST OF VARIABLES(GLOSSARY)

NAME	DESCRIPTION OR USE
BUF	LARGE ARRAY FOR WORK AREA
I	INDEX, COUNTER, OR INDICATOR
IANS	INDICATES USER'S RESOONSE TO A QUESTION(YES OR NO)
IBL	ONE HOLLERITH BLANK
IBP	NUMBER OF BITS PER PIXEL
ICH	GENERAL NAME FOR A DISC FILE CHANNEL NUMBER
ICH2	CHANNEL NUMBER FOR DISC FILE (ALTERNATE FOR PIXEL DATA)
ICHAN	CHANNEL NUMBER FOR TAPE FILE
ICHD	CHANNEL NUMBER FOR DISC FILE (PIXEL DATA)
ICHH	CHANNEL NUMBER OF DISC FILE (HEADER INFORMATION)
IDOT	ONE HOLLERITH DECIMAL
IEDGE	ARRAY TO HOLD LOWER EDGE VALUES FOR FREQUENCY DISTRIBUTION
IER	I/O CONDITION INDICATOR
IFCOL	ARRAY OF VARIOUS FORMATS TO PRINT COLUMN HEADINGS
IFDAT	ARRAY OF VARIOUS FORMATS TO PRINT DATA LINES
IFR	ARRAY OF GROUP VALUES AND FREQENCIES
IFUND	ARRAY OF VARIOUS FORMATS TO PRINT HEADING UNDERLINE
IGH	HIGHEST VALUE IN A DATA SET
IHED	ARRAY CONTAINING FILE HEADER RECORD
II	INDEX, COUNTER, OR INDICATOR
III	INDEX, COUNTER, OR INDICATOR
INC	SPACING CODE FOR SCANNER SAMPLING
INH80	'OPEN' ARRAY TO SET DEVICE CHARACTERISTICS
IPOS	ONE HOLLERITH APOSTOPHE
IPW	PIXELS/WORD (INPUT DATA)
IREC	LOGICAL RECORD NUMBER INDICATOR OR COUNTER
IRISE	VERTICAL VECTOR MAGNITUDE (NUMBER OF ROWS) FOR LINE CORRECTION
IRUN	HORIZONTAL VECTOR MAGNITUDE (NUMBER OF COLUMNS) FOR CORRECTING A LINE
ISTA	I/O STATUS INDICATOR
ISYM	ARRAY OF SYMBOLS TO REPRESENT GROUPED DATA
IV	A TEMPORARY VARIABLE
IVAL	ARRAY OF VALUES (DEFAULT OR USER DEFINED) FOR VARIOUS USES
IVAR	A TEMPORARY VARIABLE
IWLR	NUMBER OF WORDS IN A LOGICAL INPUT RECORD
IWPR	NUMBER OF WORDS IN A PHYSICAL INPUT RECORD (BLOCK)
IXL	LENGTH OF SCAN, IN MM.
IYL	WIDTH OF A SCAN, IN MM.
IYO	OFFSET FROM LEFT EDGE OF POSSIBLE SCAN FIELD, IN MM.
IZL	ONE HOLLERITH ZERO, LEFT-JUSTIFIED
IZR	ONE HOLLERITH ZERO, RIGHT-JUSTIFIED
IZZ	TWO ALPHA ZEROES(030060)
J	INDEX, COUNTER, OR INDICATOR
JBP	BITS/PIXEL (OUTPUT DATA)
JCOL	ARRAY TO HOLD CURRENT COLUMN HEADING PRINT FORMAT
JDAT	ARRAY TO HOLD CURRENT DATA PRINTING FORMAT
JJ	INDEX, COUNTER, OR INDICATOR
JJJ	INDEX, COUNTER, OR INDICATOR
JPW	PIXELS/WORD (OUTPUT)
JUND	ARRAY TO HOLD CURRENT UNDERSORING PRINT FORMAT
JWLR	NUMBER OF WORDS IN A LOGICAL OUTPUT RECORD
JWPR	NUMBER OF WORDS IN A PHYSICAL OUTPUT RECORD(OR BLOCK)
K	INDEX, COUNTER, OR INDICATOR
KIND	FILE TYPE INDICATOR(E.G.,SCAN,GROUPED DENSITIES,ETC.)
KK	INDEX, COUNTER, OR INDICATOR
KKK	INDEX, COUNTER, OR INDICATOR
KWIT	TASK CONDITION INDICATOROR INITIATOR

Figure 4—A glossary of variables used in the system.



```
L       INDEX, COUNTER, OR INDICATOR
LABEL   DUMMY ARGUMENT TO ACCEPT HOLLERITH STRINGS
LJ      LOGICAL VARIABLE
LL      INDEX, COUNTER, OR INDICATOR
LOW     LOWEST VALUE IN A DATA SET
MASK    ARRAY OF MASKS FROM 1 BIT THROUGH 8 BITS
MASK7   7-BIT MASK (177K)
MASK8   8-BIT MASK (377K)
MTEND   MASK TO TEST FOR TAPE EOF
MTEOF   COMMAND WORD FOR MTDIO TO WRITE AN EOF
MTO     FILE NAME FOR ADDRESSING THE MAGNETIC TAPE UNIT
MTOP    INDICATOR OF TAPE FILE OPEN/CLOSE STATUS
MTREW   COMMAND WORD FOR MTDIO TAPE REWIND
MTSKP   COMMAND WORD FOR MTDIO FILE SKIP
MTWR    BASIC COMMAND WORD FOR MTDIO TAPE BLOCK WRITE
MXB     DIMENSION OF BUF
MXBT    6 (MAXIMUM NUMBER OF BITS PER PIXEL FOR GROUPING DATA)
MXCOL   DISPLAY LIMIT (COLUMNS)
MXF     256 (MAXIMUM NUMBER OF UNIQUE VALUES IN A DATA SET)
MXG     64 (MAXIMUM NUMBER OF GROUPS FORMED IN SLICE
MXHED   53 (NUMBER OF WORDS IN EXPANDED HEADER RECORD)
MXI     32767  (MAXIMUM INTEGER VALUE IN 16-BIT WORD)
MXPX    MAXIMUM NUMBER OF PIXELS FOR VARIOUS OPERATIONS;EG, UNPACKING, PRINTING
MXROW   DISPLAY LIMIT (ROWS)
MXSCN   46 (NUMBER OF WORDS IN A SCANDIG HEADER RECORD)
MXTAP   4095 (MAXIMUM NUMBER OF WORDS IN A DG TAPE BLOCK)
N       INDEX, COUNTER, OR INDICATOR
NBW     16 (NUMBER OF BITS IN A MACHINE WORD (NOVA/ECLIPSE))
NC      INDICATES SELECTED PRINT FORMAT FOR COLUMN HEADING
ND      INDICATES SELECTED PRINT FORMAT FOR DATA LINE
NG      NUMBER OF GROUPS FOR ALLOCATING VALUES IN A DATA SET
NN      INDEX, COUNTER, OR INDICATOR
NO      HOLLERITH 'N ' TO COMPARE WITH KEYBOARD 'N' INPUT
NPAKD   ARRAY OF UNPACKED PIXEL VALUES
NPX     NUMBER OF PIXELS IN A SCAN LINE
NREC    RECORD COUNT
NSL     NUMBER OF SCAN LINES IN A FILE
NTH     THRESHOLD VALUE FOR CONVERTING DENSITIES TO BINARY
NU      INDICATES SELECTED PRINT FORMAT FOR UNDERLINING HEADING
RANGE   RANGE OF VALUES IN A DATA SET
RISE    VERTICAL VECTOR MAGNITUDE (NUMBER OF ROWS) FOR LINE CORRECTION
RUN     HORIZONTAL VECTOR MAGNITUDE (NUMBER OF COLUMNS) FOR LINE CORRECTION
TANG    SLOPE OF A LINE BETWEEN 2 GIVEN POINTS
VAR     TEMPORARY VARIABLE
YES     HOLLERITH 'Y ' TO COMPARE WITH KEYBOARD 'Y' INPUT
```

Figure 4—A glossary of variables used in the system—Continued.

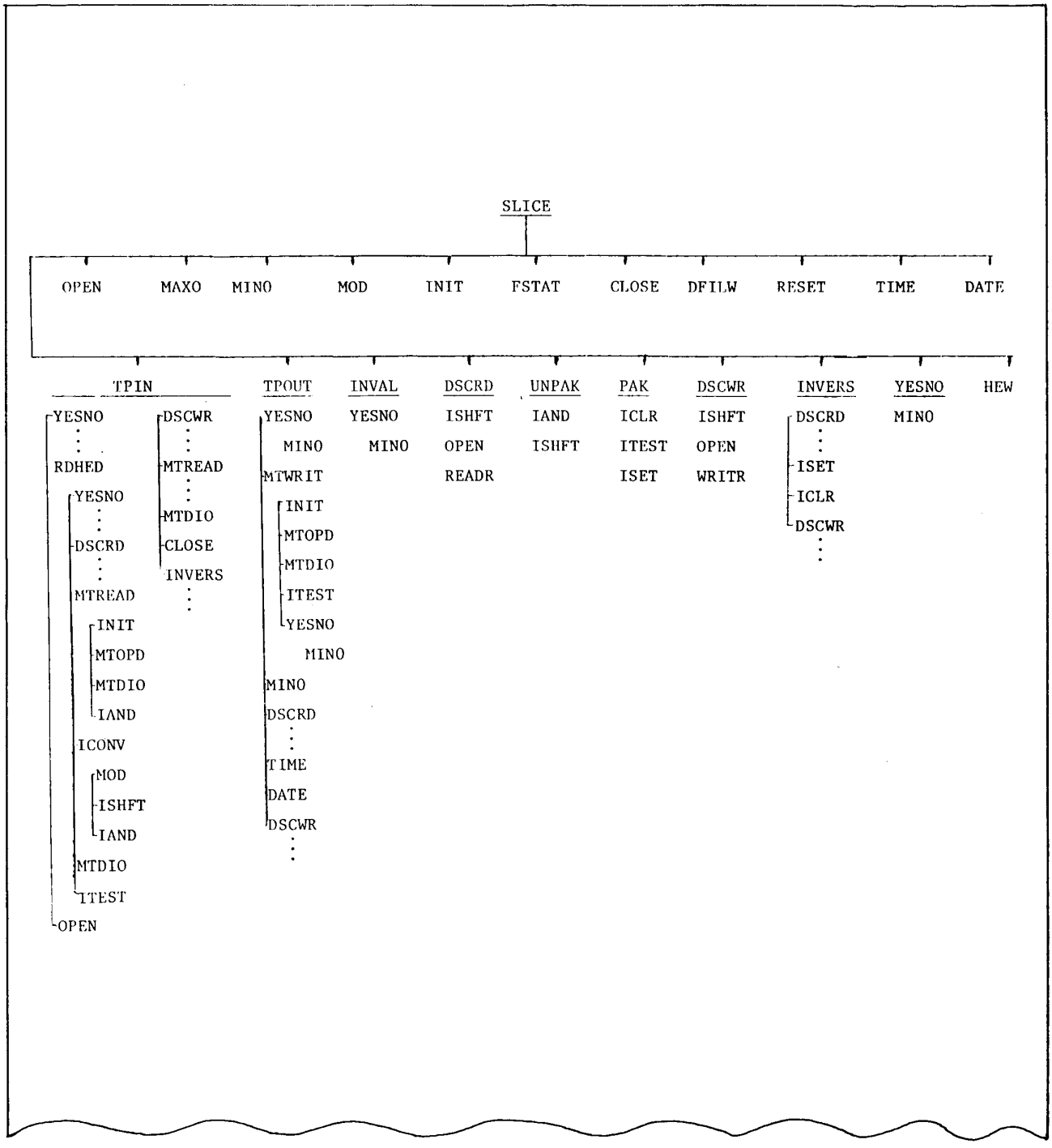


Figure 5—The three main programs call subprograms in a complex manner.

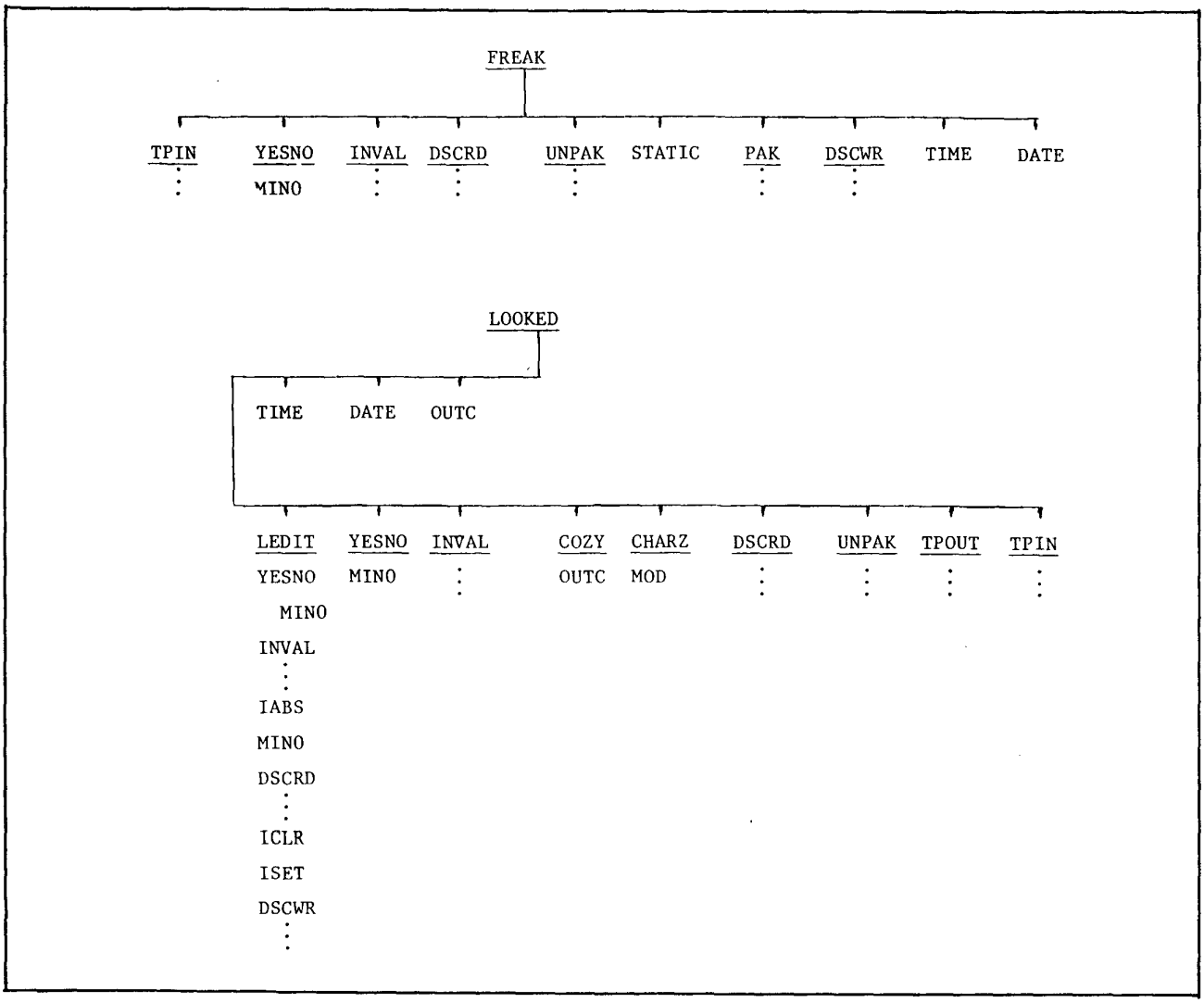


Figure 5—The three main programs call subprograms in a complex manner—Continued.

```
                    LOCATION OF VARIABLES AND COMMON

NAME    SUBPROGRAM OR COMMON

BUF     CHARZ,DSCRD,DSCWR,FREAK,ICONV,INVERS,LEDIT,LOOKED,MTREAD,
           MTWRIT,PAK,RDHED,SLICE,TPIN,TPOUT,UNPAK
I
IANS    FREAK,INVAL,LEDIT,LOOKED,MTWRIT,RDHED,SLICE,TPIN,TPOUT,YESNO
IBL     LOOKED
IBP     COMMON/H/
ICH     DSCRD,DSCWR,FREAK,RDHED,TPIN
ICH2    SLICE
ICHAN   FREAK,LOOKED,MTREAD,MTWRIT,RDHED,SLICE,TPIN,TPOUT
ICHD    DSCRD,DSCWR,FREAK,INVERS,LEDIT,LOOKED,SLICE,TPIN,TPOUT
ICHH    FREAK,TPIN,TPOUT
IDOT    LOOKED
IEDGE   HEW,SLICE
IER     DSCRD,DSCWR,FREAK,LOOKED,MTREAD,MTWRIT,RDHED,SLICE,TPIN,TPOUT
IFCOL   LOOKED
IFDAT   LOOKED
IFR     FREAK,STATIC
IFUND   LOOKED
IGH     FREAK,SLICE,STATIC
IHED    FREAK,ICONV,LOOKED,RDHED,SLICE,TPIN,TPOUT
II
III
INC     COMMON/H/
INH80   COMMON/A/ (FREAK,LOOKED,SLICE)
IPOS    LOOKED
IPW     FREAK,LOOKED,MTREAD,RDHED,SLICE
IREC    DSCRD,DSCWR,FREAK,INVERS,RDHED,SLICE,TPIN,TPOUT
IRISE   LEDIT
IRUN    LEDIT
ISTA    MTREAD,MTWRIT,RDHED,TPOUT
ISYM    SLICE
IV      DSCRD,DSCWR,FREAK,ICONV,INVERS,LEDIT,LOOKED,MTWRIT,PAK,
           RDHED,SLICE,STATIC,TPOUT,UNPAK
IVAL    INVAL,LEDIT,LOOKED
IVAR    CHARZ,DSCRD,DSCWR,FREAK,ICONV,INVERS,LEDIT,LOOKED,RDHED,SLICE
IWLR    DSCRD,FREAK,INVERS,LEDIT,LOOKED,SLICE,TPIN
IWPR    MTREAD,RDHED,TPIN
IXL     COMMON/H/
IYL     COMMON/H/
IYO     COMMON/H/
IZL     LOOKED
IZR     ICONV
IZZ     RDHED
J
JBP     PAK,SLICE
JCOL    LOOKED
JDAT    LOOKED
JJ
JJJ
JPW     SLICE
JUND    LOOKED
JWLR    DSCWR,SLICE,TPOUT
JWPR    MTWRIT,TPOUT
K
KIND    COMMON/H/
KK
KKK
KWIT    DSCRD,DSCWR,FREAK,ICONV,INVERS,LEDIT,LOOKED,MTREAD,MTWRIT,
           RDHED,SLICE,TPIN,TPOUT
L
LABEL   ICONV,INVAL,YESNO
LJ      MTWRIT,PAK,RDHED
LL      ICONV
```

Figure 6—A cross-reference of programs and variables.

```
LOW     FREAK,SLICE,STATIC
MASK    UNPAK
MASK7   ICONV
MASK8   SLICE
MTEND   MTREAD
MTEOF   MTWRIT
MTO     MTREAD,MTWRIT
MTOP    MTREAD
MTREW   MTWRIT
MTSKP   RDHED,MTWRIT
MTWR    MTWRIT
MXB     CHARZ,DSCRD,DSCWR,FREAK,INVERS,LEDIT,LOOKED,MTREAD,MTWRIT,
           SLICE,TPIN,TPOUT
MXBT    SLICE
MXCOL   LOOKED
MXF     FREAK
MXG     SLICE
MXHED   ICONV,FREAK,LOOKED,RDHED,SLICE,TPIN,TPOUT
MXI     FREAK
MXPX    FREAK,LEDIT,LOOKED,SLICE
MXROW   LOOKED
MXSCN   FREAK,RDHED,TPOUT
MXTAP   TPOUT
N
NBW     FREAK,INVERS,LOOKED,SLICE,TPIN
NC      LOOKED
ND      LOOKED
NG      NEW,SLICE
NN      ICONV,RDHED
NO      YESNO
NPAKD   NEW,LOOKED,PAK,UNPAK
NPX     COMMON/H/,NEW
NREC    DSCRD,DSCWR,FREAK,INVERS,LEDIT,LOOKED,RDHED,SLICE,TPIN,TPOUT
NSL     COMMON/H/
NTH     COMMON/H/
NU      LOOKED
RANGE   SLICE
RISE    LEDIT
RUN     LEDIT
TANG    LEDIT
VAR     FREAK,INVERS,LEDIT,SLICE
YES     YESNO

COMMON/H/    FREAK,INVERS,LEDIT,LOOKED,RDHED,SLICE,TPIN,TPOUT
COMMON/A/    FREAK,LOOKED,SLICE
```

Figure 6—A cross-reference of programs and variables—Continued.

a shared stack. With this option, local variables in a called routine retain their values between calls.

b. PARAMETER. This statement assigns a numeric or logical value to a variable name. The value cannot be altered accidentally during program execution.

c. X in column one. An X in column 1 of a FORTRAN statement allows optional compilation of the statement. A special option of the FORTRAN compiler causes the statement to be included. Otherwise the statement is treated as a comment.

d. nnnk. This form denotes the octal value of an integer constant; for example, 137K = octal 137.

e. Channel numbers. These numbers are equivalent to FORTRAN logical unit numbers. Each file or device to be accessed must be associated with a channel, by way of a call to OPEN. There are default channel numbers for several of the common I/O devices. The SCANIT system uses the system defaults of channel 11 for keyboard input and channel 10 for CRT or other display unit.

f. PAUSE. This statement interrupts the program, usually to allow the user time to perform some task. To resume program execution, the user presses any console key.

g. TYPE and ACCEPT. These statements allow unformatted I/O on the console.

The special FORTRAN library and operating system routines can be replaced with equivalent routines for another system. Each of the subprograms furnished by Data General are described to the extent necessary for conversion.

a. ABS. This is a standard FORTRAN function.

b. CLOSE (channel, error). This routine closes the file associated with the specific channel number. The value re-

turned in "error" indicates if the call was successful or, if not, what condition caused failure.

c. DATE (array, error). The month, day, and year are returned in the three-word integer array. The error value indicates success or failure.

d. DFILW (filename, error). This subprogram deletes a disk file (filename). The error value indicates success or failure.

e. FSTAT (channel, attributes, error). This subprogram is used to alter the "attributes" of the file currently associated with the specified channel. File attributes include read and write protection, file *type,* and others. Each attribute is determined by the on-off state of its assigned bit within the integer attributes. Thus, there cannot be more than 16 attributes in this system. The error value indicates success or failure.

f. IABS. This is a standard FORTRAN function.

g. LAND (arg1, arg2). This function compares the bit patterns of the two arguments. The resulting value of the function is an integer with a one-bit where both arguments contained one-bits, and a zero bit where one or both arguments contained zero bits.

h. ICLR (word, position). This subroutine stores a zero in the specified bit position of the given word. The value for position can range from zero (for least significant at the right end) to 15 (for most significant at the left end).

i. IFIX. This is a standard FORTRAN function.

j. INIT (name, type, error). This subroutine acquaints the system with name, which may be a device or a directory of files. Type specifies full or partial initialization, error indicates success or failure.

k. ISET (word, position). This subroutine stores a one in the specified bit position of the given word. The value for position can range from zero (for least significant at the right end) to 15 (for most significant at the left end).

1. ISHFT (word, number). This subprogram can be used to shift the bit pattern of the word to the right or left, according to the number of bit positions specified. A negative number causes a right shift; a positive number causes a left shift. Vacated bit positions are zero-filled.

m. ITEST (word, position). This logical function tests the value of the bit in the specified position of the given word. Test is true (-1) if the bit is one and false (zero) if the bit is zero. The value for position can range from zero (for least significant at the right end) to 15 (for most significant at the left end).

n. MAX0. This is a standard FORTRAN function.

o. MIN0. This is a standard FORTRAN function.

p. MOD. This is a standard FORTRAN function.

q. MTDIO (channel, command, array, status, error, count). This subroutine handles free format tape I/O for the device currently associated with the specified channel. Selected bits of the "command" word indicate the I/O action (for example, read, write, move, and others). The command word may also be used to indicate word or record count. "Array" is the core area used for receiving or providing data for the I/O action. Each bit of the status word provides information to calling routines for aid in case of errors. The error flag is returned to caller and indicates success or failure. The count argument is optional and returns the actual number of words (or records).

r. MTOPD (channel, name, mask, error). This subroutine opens the named device and associates it with the specified channel. Each bit of the mask word can be set to alter the appearance of some characteristic of the device. The SCANIT system does not use this facility. Error is returned and indicates success or failure.

s. OPEN (channel, name, mode, error). This subroutine opens disk file "name" and associates it with the specified channel. Mode controls the type of I/O allowed on the file (1 = read only, 3 = write by user only but read by all users, any other value = read and write by any user). Error is returned and indicates success or failure.

t. READR (channel, rec1, array, nrec, error). This subroutine reads "nrec" logical records from the file opened on the specified channel into "array. " The read operation starts at record number rec1. Error, returned, indicates success or failure.

u. RESET. This subroutine has no arguments and closes all files currently open to the job.

v. TIME (array, error). This subroutine returns hours, minutes, and seconds in the array. The returned error code indicates success or failure..

w. WRITR (channel, rec1, array, nrec, error). This subroutine writes "nrec" logical records from "array" into the file associated with specified channel. Writing starts with record number rec1. Error, returned, indicates success or failure.

2. PROGRAM DESCRIPTIONS AND LISTINGS

2.1. FORTRAN Subroutine CHARZ

2.1.1. Problem and Solution Method

Each digit of an integer is to be converted to the ASCII character code for that digit. Solution method is to use modulo ten, division by 10, and addition of ASCII zero. The resulting characters are stored one per word in an array.

2.1.3. Processing

a. Processing logic. N is set equal to IVAR for use in processing. The right-hand digit of N is isolated by MOD (N,10), converted to its ASCII character code by adding 48 (decimal) and stored in the high-order element of array BUF. N is then divided by 10 to place the next digit in position for a MOD (N, 10), conversion and storing in the next lower word of BUF. After each division, N's value is tested. If N > 0, the procedure is repeated. If N = 0, blanks are stored in unused words of BUF, and control returns to calling routine. IVAR's value is not changed.

b. Linkages. CHARZ is called by LOOKED and uses the MOD function.

c. Variables and constants. N is set equal to input integer WAR at entry and is used for the actual processing, J is a subscript for BUF, 48 is the value used to convert integer values to ASCII characters. Refer to glossary for others.

e. Error handling provisions. If there are more digits in the input integer than there are elements in the output array, an error message is issued and processing stops with no return.

f. Restrictions and limitations. Number of digits in input value cannot exceed dimension (MXB) of output array.

2.1.5. Interfaces

Array BUF contains an ASCII character code for a value from zero through nine in the right-hand byte of each element. WAR is the integer value provided by the calling routine. MXB, the dimension of BUF, is specified by the calling routine.

```
                    CHARZ
      COMPILER NOSTACK
      SUBROUTINE CHARZ(IVAR,BUF,MXB)
C TO CONVERT A POSITIVE INTEGER (IVAR) TO MXB CHARACTERS, STORED ONE PER WORD,
C RIGHT-JUSTIFIED, IN BUF.
C BLANKS REPLACE ZEROES IN UNUSED WORDS OF BUF.
      INTEGER BUF(MXB)
      N=IVAR
      J=MXB
      DO 20 I=1,MXB
      K=MOD(N,10)
      BUF(J)=K+48
      N=N/10
      J=J-1
      IF(N.LE.0) GO TO 25
   20 CONTINUE
      IF(N.GT.0) GO TO 80
   25 DO 30 I=1,J
      BUF(I)=32
   30 CONTINUE
      GO TO 90
C
C ERROR.
C
   80 WRITE(10,1000)IVAR,MXB
 1000 FORMAT('0INTEGER ',I6,' HAS TOO MANY DIGITS TO STORE IN STRING OF
     I2,' CHARACTERS.')
      STOP
   90 RETURN
      END
```

2.2. FORTRAN Subroutine COZY

2.2.1. Problem and Solution Method

Provision must be made to change from normal to compact printing when a variable-mode printer is available. This routine handles mode changes for an HP-2635A.

2.2.3. Processing

a. Processing logic. A series of control characters are output to the printer.

b. Linkages. This routine is called by LOOKED. Assembly routine OUTC passes the control characters to the console.

c. Variables and constants. Constants for the control characters are determined by the HP-2635A manual. The argument L is a switch variable specifying either normal or compact printing.

f. Restrictions and limitations. This is a special-purpose routine limited to the HP-2635A.

2.2.4. Output

The output consists of a string of control characters directed to the console.

2.2.5. Interfaces

Output characters must be recognizable and usable by the console. In the event that a different output device is used, an empty routine should be substituted unless references to COZY are removed from LOOKED and the load list.

```
                    COZY
      SUBROUTINE COZY(L)
C FOR USE ONLY WITH HP-2635A, TO SWITCH TO AND FROM COMPACT
C PRINTING MODE.
C     IF L = 1, PRINT MODE IS SET TO COMPACT.
C     IF L = 2, PRINT MODE IS SET TO NORMAL.
      GO TO (10,20),L
   10 CALL OUTC(33K)
      CALL OUTC(46K)
      CALL OUTC(153K)
```

```
      CALL OUTC(62K)
      CALL OUTC(123K)
      CALL OUTC(33K)
      CALL OUTC(46K)
      CALL OUTC(154K)
      CALL OUTC(60K)
      CALL OUTC(104K)
      GO TO 90
   20 CALL OUTC(33K)
      CALL OUTC(46K)
      CALL OUTC(153K)
      CALL OUTC(60K)
      CALL OUTC(123K)
      CALL OUTC(33K)
      CALL OUTC(46K)
      CALL OUTC(154K)
      CALL OUTC(66K)
      CALL OUTC(104K)
   90 RETURN
      END
```

2.3. FORTRAN Subroutine DSCRD

2.3.1. Problem and Solution Method

A variable number of logical records previously stored on disc are to be copied to core memory. A free-format Data General disc reading routine provides efficient transfer.

2.3.2. Input

Input consists of logical records of uniform length in a randomly organized file.

2.3.3. Processing

a. Processing logic. The routine uses the channel number to construct a file name and open the file if it is not already open. The routine reads a series of records into core, checking for error conditions.

b. Linkage. The subprogram is called by FREAK, SLICE, INVERS, LEDIT, RDHED, and TPOUT, and calls DG routines READR, ISHFT, and OPEN.

c. Variables and constants. N contains ASCII characters zero and blank (030040). This word (N) can be added to the channel number to give a file name (IV) if a file has not been opened and associated with the channel already.

IVAR—This is the number of bytes per record.

e. Error handling provisions. One argument (KWIT) returns the status of a disc read, whether successful (0), or not (1).

2.3.4. Output

In the event of a disc read error, a message is printed on console.

2.3.5. Interfaces

BUF is an array of length MXB. IREC is the logical record number for the first record to be copied. ICH defines the relative channel number for the disc file. NREC is the number of successive logical records to be read.

```
                    DSCRD
      COMPILER NOSTACK
      SUBROUTINE DSCRD(ICH,IREC,NREC,BUF,MXB,IWLR,KWIT)
C COPIES NREC LOGICAL RECORDS (LENGTH IWLR WORDS) FROM DISC TO CORE, BEGINNING
C AT RECORD NUMBER IREC.
C BUF, DIMENSIONED MXB, IS THE ARRAY TO RECEIVE THE DATA.
C MXB CANNOT BE LESS THAN NREC * IWLR.
C ICH IS THE RELATIVE CHANNEL NUMBER FOR THE DISC FILE.
C KWIT RETURNS THE STATUS OF DISC READ:
C    0 = SUCCESSFUL READ
C    1 = ERROR
C
      INTEGER BUF(MXB)
X     WRITE(10,6666)
X6666 FORMAT('0ENTERING DSCRD')
      KWIT=0
C CONSTRUCT A FILE NAME USING ASCII CHARACTER REPRESENTATION
C OF THE CHANNEL NUMBER.
      N=30040K
      IV=ISHFT(ICH,8)+N
      IVAR=IWLR*2
      CALL OPEN(ICH,IV,2,IER,IVAR)
      IF(IER.NE.1.AND.IER.NE.40.AND.IER.NE.20) GO TO 801
      CALL READR(ICH,IREC,BUF,NREC,IER)
      IF(IER.EQ.1) RETURN
```

```
 801 KWIT=1
     WRITE(10,8000) IER
8000 FORMAT('0DISC READ ERROR',I4)
     RETURN
     END
```

2.4. FORTRAN Subroutine DSC WR

2.4.1. Problem and Solution Method

An efficient method is needed to rapidly transfer data from core to disc. This subprogram provides the necessary preliminaries to call a DG system routine that allows control of data flow at nearly the assembly language level.

2.4.3. Processing

a. Processing logic. This routine may be used with more than one channel number. Because the number of channels cannot be anticipated, every call to DSCWR initiates an attempt to open the file. A unique file name is created by converting the input, relative channel number to an ASCII character, after the RDOS naming convention. The number of bytes in an output record is provided to the opening routine, if the file is not already open. After opening, a DG routine provides rapid transfer of NREC records from core to disc starting from relative position IREC.

b. Linkages. The program is called by FREAK, SLICE, TPIN, INVERS, TPOUT, and LEDIT, and calls DG routines OPEN, WRITR, AND ISHFT.

c. Variables and constants. Refer to the glossary for terms not defined here. The value IV—030040 octal—is added to the integer relative channel number to give an ASCII character file name. WAR represents the length of a randomly organized file in bytes per record.

e. Error handling provisions. Repeated attempts to open the same file generate error codes. These are distinguished from actual errors before writing proceeds. Opening or writing errors prompt a combined general message.

f. Restrictions and limitations. Channel numbers, from which file names are created, must be provided in accordance with system specifications.

2.4.4. Output

The data in the array BUF are written in a random-access disc file as logical records of uniform length.

2.4.5. Interfaces

BUF contains data to be copied to disc. KWIT returns a message on disc write status. The remaining arguments are defined in the glossary.

2.4.7. Run description

Locate subprogram with calling routines and FORTRAN library. Follow RDOS conventions in selection of multiple channel names.

```
                    DSCWR
      COMPILER NOSTACK
      SUBROUTINE DSCWR(ICH,IREC,NREC,BUF,MXB,JWLR,KWIT)
C COPIES RECORDS FROM CORE TO DISC FILE, OPENING THE FILE FIRST,
C IF NECESSARY.
C BUF = ARRAY FROM WHICH TO WRITE DATA, LENGTH MXB.
C MXB CANNOT BE LESS THAN NREC * JWLR.
C ICH = RELATIVE CHANNEL NUMBER.
C IREC = RELATIVE POSITION OF FIRST RECORD TO BE WRITTEN.
C NREC = NUMBER OF LOGICAL RECORDS TO BE WRITTEN.
C JWLR = NUMBER OF WORDS IN A LOGICAL RECORD.
C KWIT RETURNS STATUS OF DISC WRITE:
C   0 = SUCCESSFUL WRITE
C   1 = DISC ERROR
C
      INTEGER BUF(MXB)
X     WRITE(10,6666)ICH
X6666 FORMAT('0ENTERING DSCWR, ICH',I3)
      KWIT=0
C GENERATE ALPHA FORM OF NUMERIC CHANNEL NO.;USE AS FILE NAME.
      IV=30040K
       BUF(1)=ISHFT(ICH,8)+IV
      IVAR=JWLR*2
      CALL OPEN(ICH,BUF,2,IER,IVAR)
      IF(IER.NE.1.AND .IER.NE.40.AND.IER.NE.20) GO TO 804
      GO TO 990
      CALL WRITR(ICH,IREC,BUF,NREC,IER)
      IF(IER.EQ.1) GO TO 990
  804 KWIT=1
      WRITE(10,8004) IER
 8004 FORMAT('0DISC ERROR',I4)
  990 RETURN
      END
```

2.5. FORTRAN Main Program FREAK

2.5.1. Problem and Solution Method

User needs to look at a frequency distribution of part or all of his data. Data elements may be from one to eight bits per pixel; therefore, the number of groups may vary from two to 256. This program obtains needed information from the header record to construct a frequency distribution.

2.5.2. Input

The header record supplies the number of bits per pixel and the information needed to compute the number of pixels per record. Logical records of data are on disc. User types in directions on sample size. User can specify beginning and ending row and column or ask for all data to be included.

2.5.3. Processing

a. Processing logic. From information in the header, the actual number of data pixels can be calculated, thus excluding the border and padding pixels added by the Scandig controller. User is asked to type directions as to sample size. Each record to be included is read from disc, data is unpacked, and the corresponding groups are incremented. If the count for any group exceeds 32767, the count is zeroed out and starts again.

b. Linkages. This program calls PSW routines TPIN, YESNO, INVAL, DSCRD, UNPAK, STATIC, PAK, and DSCWR. It makes use of DG routines OPEN, TIME, DATE, MAX0, MIN0, MOD, and RESET.

c. Variables and constants. KKK and IVAR are the beginning and ending record numbers to be included in the sample. KK and IV are the beginning and ending pixel numbers of data to be included from each record. IV is used later in the program in converting the mode to character format and also for indicating the number of print lines in the frequency distribution. The remaining variables and constants are defined in the glossary or are used as indexes or subscripts.

e. Error handling provisions. Input data (tape and console) are checked for reasonableness. I/O status is checked after each operation. The count for each group is not allowed to exceed 32767 (MXI). At the time the count would exceed MXI, a message is printed and the count for that group is reset to zero. It is evident that this arbitrary procedure could affect the mode, but the user is given enough information to adjust any decision based on the mode.

f. Restrictions and limitations. The number of bits per pixel cannot be greater than eight. The number of groups in the frequency distribution cannot be greater than 256. The maximum number of pixels per scan line is 4500 (MXPX).

2.5.4. Output

The mode of the frequency distribution is added to the header. The frequency distribution table is printed, using a maximum of 80 print positions. The date and time are also printed.

```
                    FREAK
      COMPILER NOSTACK
C CONSTRUCTS AND PRINTS FREQENCY TABLE FROM PACKED DATA ON TAPE.
C NUMBER OF GROUPS MAY VARY FROM 2 TO 256.  BITS/PIXEL MAY VARY FROM
C 1 TO 8.
C PIXELS OF ALL 1-BITS WERE ADDED FOR BORDERING AND PADDING AT SCAN TIME BY
C THE SCANDIG CONTROL PROGRAM.
C THE NUMBER OF PIXELS SO ADDED WILL NOT BE INCLUDED IN THE COUNT,SO THAT
C  ONLY ACTUAL DENSITY DATA WILL BE  CONSIDERED IN THE DISTRIBUTION.
      PARAMETER ICHAN=0,ICHD=1,ICHH=2
      PARAMETER MXHED=53,MXSCN=46,NBW=16
      PARAMETER MXB=3000,MXF=256, MXPX=4500
      INTEGER BUF(MXPX),IHED(MXHED),IFR(MXF,2)
      COMMON/H/IBP,NPX,NSL,NTH,INC, IXL,IYO,IYL,KIND
      COMMON/A/INH80(3)
      DATA INH80/-1,2,40000K/
      DATA MXI/32767/
C
C
      CALL OPEN(10,'$TTO',INH80,IER)
      CALL TIME(IHED(48),IER)
      CALL DATE(IHED(51),IER)
      WRITE(10,5000)(IHED(I),I=48,53)
 5000 FORMAT('0TIME=',3I3,6X,'DATE=',I3,':',I2,':',I2)
      CALL TPIN(BUF,MXB,IHED,MXHED,IWLR,ICHAN,ICHD,KWIT)
      IF(KWIT.NE.0) GO TO 990
      LL=2**IBP
      IF(LL.GT.MXF) GO TO 800
C INITIALIZE ARRAY IFR.
      DO 5 I=1,LL
      IFR(I,1)=I-1
    5 IFR(I,2)=0
      IPW=NBW/IBP
C CALCULATE NUMBER OF REAL DATA PIXELS IN EACH LINE.
```

```
C THESE CALCULATIONS ARE BASED ON USE OF SCANDIG SPACING WHEEL VARYING
C FROM 12.5 TO 100 MICRONS.
      VAR=(2**INC)*12.5
      IV=(1000./VAR)*IYL+1
      IVAR=NSL-1
      KK=2
      KKK=2
      NREC=1
      CALL YESNO(IANS,'USE ALL DATA FOR FREQUENCY TABLE',32)
      IF(IANS.EQ.1) GO TO 12
   10 I=4
      CALL INVAL('FIRST AND LAST ROW, FIRST AND LAST COLUMN',41,BUF,I)
      KKK=MAX0(BUF(1),KKK)
      IVAR=MIN0(BUF(2),IVAR)
      KK=MAX0(BUF(3),KK)
      IV=MIN0(BUF(4),IV)
      IF(KKK.GT.IVAR) GO TO 803
      IF(KK.GT.IV) GO TO 803
X     WRITE(10,7000) IBP,NPX,NSL,NTH,INC,IXL,IYO,IYL
X7000 FORMAT('0/H/',8I6)
X     WRITE(10,7001)KKK,IVAR,KK,IV
X7001 FORMAT('0',4I6)
   12 II=IV-KK+1
      IF(II.GT.MXPX) GO TO 804
      JJ=MXPX-IWLR+1
      DO 20 I=KKK,IVAR
      IREC=I-1
C


      CALL DSCRD(ICHD,IREC,NREC,BUF(JJ),MXB,IWLR,KWIT)
      IF(KWIT.GT.0) GO TO 990
      CALL UNPAK(IBP,BUF(JJ),BUF(1),KK,II,IPW)
      DO 18 K=1,II
      J=BUF(K)+1
      IF(IFR(J,2).LT.MXI) GO TO 17
      WRITE(10,1017)BUF(K)
 1017 FORMAT('0COUNT REDUCED BY 32767 FOR VALUE',I4,'XXXXXXXXXXXXXXXXXXXXX')
      IFR(J,2)=0
   17 IFR(J,2)=IFR(J,2)+1
   18 CONTINUE
   20 CONTINUE
      CALL STATIC(IFR(1,2),LL,IGH,LOW,K)
C CONVERT MODE (K) TO ASCII CHARACTERS AND STORE IN IHED FOR POSSIBLE LATER USE.
      NTH=K
      IV=K
      II=2
      K=4
      DO 30 I=II,K
      J=6-I
      BUF(J)=MOD(IV,10)+48
      IV=IV/10
   30 CONTINUE
      BUF(1)=48
      J=8
      I=1
      JJ=0
      CALL PAK(J,IHED(37),BUF,I,K,II)
      CALL DSCWR(ICHH,JJ,I,IHED,MXHED,MXHED,KWIT)
      IF(KWIT.NE.0) GO TO 990
      IV=(IGH-LOW+4)/4
      IGH=IGH+1
      WRITE(10,1001)IV
 1001 FORMAT('0FREQUENCY DISTRIBUTION TABLE NEEDS',I3,' LINES')
      PAUSE' POSITION PAGE, THEN HIT ANY KEY'
      DO 53 J=1,IV
      LOW=LOW+1
      WRITE(10,1002)(IFR(L,2),IFR(L,1),L=LOW,IGH,IV)
 1002 FORMAT(4(I14,' (',I3,')'))
   53 CONTINUE
      WRITE(10,1003)(IHED(I),I=48,53)
 1003 FORMAT(////,I3,':',I2,':',I2,2X,3I3)
      GO TO 990
C ERROR CONDITIONS.
  800 WRITE(10,8000)IBP
 8000 FORMAT('0UNREASONABLE INPUT TAPE, BITS PER PIXEL=',I2)
```

```
      GO TO 990
  803 WRITE(10,8003)
 8003 FORMAT('0ERROR IN SAMPLE SELECTION')
      GO TO 900
  804 TYPE 'TOO MANY PIXELS FOR AN UNPACKED LINE, =',II
  900 CALL YESNO(IANS,'RE-ENTER DATA',13)
      IF(IANS.EQ.1) GO TO 10
  990 CALL RESET
      WRITE(10,1000)
 1000 FORMAT(//////)
      STOP
      END
```

2.6. FORTRAN Subroutine HEW

2.6.1. Problem and Solution Method

A frequency distribution is needed for a collection of data, such as scan density values. Each element in the collection must be assigned a group value (0, 1, 2. . . .) according to divisions supplied to this routine.

2.6.3. Processing

a. Processing logic. The data collected to be grouped is unsorted and certain values may not be represented within the range of the data. Each group is distinguished by an "edge"—the lowest value in a group. Starting with zero, edges are stored in ascending order for comparison with the data collection. The result of the comparison is that every element of the data collection is replaced by a group number of zero or more. The group of highest value will contain all data element values greater than or equal to the highest group edge,

b. Linkages. HEW is called by the main program SLICE.

2.6.5. Interfaces

NPAKD is the data collection. NPX is the number of elements in NPAKD. LEDGE is the array of group edges. NG is the number of groups.

```
                  HEW
      COMPILER NOSTACK
      SUBROUTINE HEW(NPAKD,NPX,NG,IEDGE)
C ASSIGNS A GROUP NUMBER TO EACH OF NPX VALUES IN ARRAY NPAKD.
C USES LOWER EDGE VALUES,IN ARRAY IEDGE, TO DETERMINE PROPER GROUP.
C THE VALUE IN THE FIRST ELEMENT OF IEDGE IS ZERO (THE LOWEST EDGE).
C
      INTEGER NPAKD(NPX),IEDGE(NG)
      DO 20 I=1,NPX
      DO 16 J=2,NG
      K=J-1
      IF(NPAKD(I).LT.IEDGE(J)) GO TO 18
   16 CONTINUE
      K=NG
   18 NPAKD(I)=K-1
   20 CONTINUE
      RETURN
      END
```

2.7. FORTRAN Subroutine ICONV

2.7.1. Problem and Solution Method

A group of ASCII characters, assumed to represent the positive integers 0 to 9, must be converted into an integer value. The task is done by using necessary byte operations without benefit of assembly language.

2.7.3. Processing

a. Processing logic.

1. General logic. A 16-bit word contains two bytes with an ASCII, 7-bit character in each byte. Each half of the word must be masked off in turn during the conversion. The numbers obtained represent decimal integers which, when multiplied by a power of 10 and added together, return the desired value.

2. Detailed logic. Although the subroutine is more general, usually four ASCII characters will be input in two 2-byte words. The character in the right-hand byte of each word is obtained by a logical AND with 177K mask. An 8-bit right shift operation brings the left byte in each word into position for the logical .AND. operation. A right shift alone obtains the left byte given zero left-fill, but masking in addition adds generality. The result of the .AND. is a 7-bit ASCII character representation of an integer. To obtain an integer, an ASCII zero must be subtracted from the character form. For example, a decimal 9 (11 octal) is the result of subtracting ASCII zero (60 octal) from ASCII 9 (71 octal). The one-to-NN-digit decimal numbers can be developed by multiplying 1-digit integers by powers of 10, e.g., 1003 = (1*10**3)+3.

b. Linkages. This routine is called by RDHED and uses the functions LAND, ISHFT, and MOD.

c. Variables and constants. Most constants and variables are defined centrally in a glossary to reduce redundancy, however, local variables are described here.

1. Constants: IZR is ASCII zero (000060K). MASK7 is 177 octal or 127 decimal or seven 1-bits in binary (0 000 000 001 111 111).

2. Variables: IV is the binary value of a byte (temporary). LL is the byte counter (subscript). IVAR stores the integer value obtained.

e. Error handling provisions. The variable IV is tested to see if it falls within the range 0 to 9. Any non-numeric character causes an immediate message and an error indicator is returned to the calling routine for subsequent decision as to appropriate action.

f. Restrictions and limitations. The converted value returned by the subroutine is limited to the maximum integer value allowed by the system, e.g., 2**15-1.

2.7.5. Interfaces

IHED (dimension MXHD) contains the ASCII characters. NN is the number of characters. WAR returns the integer value to the calling routine. KWIT indicates success or error on return. LABEL is a definition of the value in IVAR.

2.7.7. Run Description

Subprogram to be located with calling routines and FORTRAN Library.

```
                ICONV
      COMPILER NOSTACK
      SUBROUTINE ICONV(IHED,MXHED,IVAR,NN,KWIT,LABEL)
C CONVERTS NN ASCII CHARACTERS TO AN INTEGER VALUE
C KWIT IS UNCHANGED IF NO ERROR OCCURS.
C KWIT IS INCREMENTED BY 1 IF ERROR OCCURS.
      INTEGER IHED(MXHED),LABEL(6)
      DATA IZR/60K/,MASK7/177K/
      IVAR=0
      DO 100 I=1,NN
      LL=(I-1)/2+1
C ODD I = LEFT CHARACTER,  EVEN I = RIGHT CHARACTER OF WORD.
      IF(MOD(I,2).EQ.0) GO TO 1
C LEFT CHARACTER.
      IV=IAND(ISHFT(IHED(LL),-8),MASK7)-IZR
      GO TO 2
C RIGHTHAND CHARACTER.
    1 IV=IAND(IHED(LL),MASK7)-IZR
    2 IF(IV.LT.0.OR.IV.GT.9) GO TO 90
      IF(IVAR-3276) 100,3,92
    3 IF(IVAR+IV.GT.3283) GO TO 92
  100 IVAR=IVAR*10+IV
      GO TO 99
C
C ERROR - NON-NUMERIC CHARACTER.
   90 LL=NN/2
      WRITE(10,1000)I,(IHED(K),K=1,LL)
 1000 FORMAT('0NON-NUMERIC CHARACTER, POSITION #',I3,' IN',1X,20A2)
      IVAR=0
      KWIT=KWIT+1
       RETURN
   92 LL=NN/2
      WRITE(10,1001),IVAR,IV,(IHED(I),I=1,LL)
 1001 FORMAT('0INTEGER VALUE WILL EXCEED CAPACITY')
      RETURN
C
C DISPLAY DESCRIPTION AND VALUE.
   99 WRITE(10,2000)LABEL,IVAR
 2000 FORMAT(1X,6A2,I5)
      RETURN
      END
```

2.8. FORTRAN Subroutine INVAL

2.8.1. Problem and Solution Method

An interactive system implies that data will be input during execution. The solution method is to perform integer input handling through one routine.

2.8.2. Input

Keyboard input consists of one or more integers, each except the last followed by a comma. A carriage return follows the last value entered.

2.8.3. Processing

a. Processing logic. A message to the console tells the operator how many integers to enter and may also identify the data expected. After data are keyed in, a message asking for a "yes" or "no" response gives the operator a chance to reenter the data, if desired. Otherwise, control returns to the calling routine.

b. Linkages. INVAL is called by LOOKED, LEDIT, FREAK, and SLICE. INVAL calls YESNO and uses the RDOS ACCEPT statement, allowing unformatted input.

e. Error handling provisions. The program provides the opportunity to reenter data if an error is detected. For example, if the CR is hit before all values are keyed in, the system will wait. The operator can enter enough commas to complete a dummy list, then reenter data correctly.

2.8.4. Output

Prompting messages are output to console.

2.8.5. Interfaces

A string array (LABEL dimensioned L) identifying the input data is provided by the caller. LABEL may be empty (L = 0). Calling routine also specifies K, the number of integer values to be keyed in. The input values are returned in array IVAL, one per element.

```
                INVAL
      COMPILER NOSTACK
      SUBROUTINE INVAL(LABEL,L,IVAL,K)
C ROUTINE TO ACCEPT K INTEGERS TYPED IN AT EXECUTION TIME.
C L IS NUMBER OF CHARACTERS IN MESSAGE.
C IF L=0, NO MESSAGE IS PRINTED.
      INTEGER YES,NO
      INTEGER LABEL(1),IVAL(K)
      J=(L+1)/2
   15 IF(K.GT.1) WRITE(10,1002)K
 1002 FORMAT(' TYPE IN ',I3,1X,'INTEGERS, SEPARATED BY COMMAS')
      IF(K.EQ.1) WRITE(10,1003)
 1003 FORMAT(' TYPE IN 1 INTEGER')
      IF(J.GT.0) WRITE(10,1000)(LABEL(I),I=1,J)
 1000 FORMAT('0',39A2)
      DO 19 I=1,K
   19 IVAL(I)=0
      ACCEPT (IVAL(I),I=1,K)
      CALL YESNO(IANS,'DO YOU WISH TO RE-ENTER THE DATA',32)
      IF(IANS.EQ.1) GO TO 15
      RETURN
      END
```

2.9. FORTRAN Subroutine INVERS

2.9.1. Problem and Solution Method

If a positive transparency was scanned instead of a negative, the scan image can be reversed by taking the 1's complement of each pixel. Additional processing is optional for the padding pixels added to each record of a Scandig data file.

2.9.3. Processing

a. Processing logic. In DG equipment, the negative of a number is its 2's complement. To obtain the 1's complement, subtract 1 from the 2's complement. This process is applied to each word containing one or more pixels to be inverted. If all pixels are to be complemented, this is all the processing necessary for each record. If the padding pixels are excluded from the complementing, their values will be set to all 1-bits for densities, or to all 0-bits for thresholded data.

b. Linkages. INVERS is called by TPIN and SLICE. INVERS calls DSCRD, ISET, ICLR, and DSCWR.

c. Variables and constants. KK and IVAR are the first and last records to be processed. K and IV indicate bit processing positions within each record.

d. Error handling provisions. I/O status is checked after each operation. Header record values are checked for reasonable data format.

e. Restrictions and limitations. There must be a multiple of 32 bits per record (2*16*no. of words).

2.9.4. Output

Inverted image replaces original on disc.

2.9.5. Interfaces

BUF, dimensioned MXB, is the I/O buffer. ICHD is the channel number for data file. I WLR is the logical record length in words. JJ, at calling, is a switch indicating how to process the padding pixels. On return, JJ indicates if task was done.

```
                INVERS
      COMPILER NOSTACK
      PARAMETER NBW=16
      SUBROUTINE INVERS(BUF,MXB,ICHD,IWLR,JJ)
C SPECIALIZED ROUTINE FOR PROGRAMS PROCESSING SCANDIG DATA
C FOR SUBSEQUENT USE IN THE UNIVAC 1108  SYSTEM.
C COMPLEMENTS VALUES IN A MATRIX.
C SET JJ TO 1 OR 2 BEFORE CALLING INVERS.
C    JJ = 1 IS A REQUEST TO COMPLEMENT ALL BUT THE PADDING PIXELS.
C    JJ = 2 IS A REQUEST TO COMPLEMENT ALL PIXEL VALUES.
```

```
C PADDING PIXELS ARE THE BORDER AND PADDING PIXELS ADDED DURING SCANNING
C ON THE SCANDIG.
C IF TASK IS DONE,JJ IS RETURNED AS ZERO.
C IF TASK IS NOT DONE, JJ IS RETURNED AS A NON-ZERO.
C CAUTION: THIS ROUTINE EXPECTS AN EVEN MULTIPLE OF 16-BIT WORDS
C IN EACH LOGICAL RECORD.  NBW IS USED AS A CONSTANT AND AS A WARNING.
      INTEGER BUF(MXB)
      COMMON/H/IBP,NPX,NSL,NTH,INC,IXL,IYO,IYL,KIND
      NN=JJ
      NREC=1
C SET DEFAULTS FOR COMPLEMENTING ALL VALUES.
      K=IWLR
      IVAR=NSL
      KK=1
      IF(NN.EQ.2) GO TO 201
C CALCULATE NUMBER OF PADDING PIXELS AT END OF EACH LINE.
C RESET DEFAULTS TO EXCLUDE WORDS WITH ONLY PADDING PIXELS.
      VAR=(2**INC)*12.5
      IV=(1000./VAR)*IYL
      IV=(NPX-IV-1)*IBP
X     WRITE(10,6232)IV,NPX,IWLR,K,IYL,INC,NSL,VAR
X6232 FORMAT('0INVERS',7I6,F6.1)
      IF(IV.GE.32) GO TO 801
      IF(IV.LE.16) GO TO 101
      K=IWLR-1
      IV=IV-16
  101 IVAR=NSL-1
      KK=2
C PROCESS EACH ROW(1 ROW PER RECORD).
  201 DO 103 I=KK,IVAR
      IREC=I-1
      CALL DSCRD(ICHD,IREC,NREC,BUF,MXB,IWLR,KWIT)
      IF(KWIT.NE.0) GO TO 990
C REVERSE ALL WORDS CONTAINING ANY NON-PADDING PIXELS.
      DO 102 J=1,K
C TWO'S COMPLEMENT, SO - - -
      BUF(J)=-(BUF(J))
C SUBTRACT ONE TO OBTAIN ONE'S COMPLEMENT.
      BUF(J)=BUF(J)-1
  102 CONTINUE
      IF(NN.EQ.2) GO TO 203
        IF(IBP.EQ.1) GO TO 500
C RESTORE ONES IN PADDING PIXELS BECAUSE PADDING IS SUPPOSED
C TO BE 1'S IF IBP EXCEEDS 1 (TO SIMULATE DENSE BACKGROUND).
      DO 402 J=1,IBP
      CALL ISET(BUF(1),NBW-J)
  402 CONTINUE
      DO 403 J=1,IV
      CALL ISET(BUF(K),J-1)
  403 CONTINUE
      GO TO 203
  STORE ZEROES IN PADDING PIXELS BECAUSE PADDING SHOULD BE ZEROES
C IF IBP EQUALS 1 (TO SIMULATE THRESHOLDED DENSITIES).
  500 CALL ICLR(BUF(1),15)
      DO 202 J=1,IV
      CALL ICLR(BUF(K),J-1)
  202 CONTINUE
  203 CALL DSCWR(ICHD,IREC,NREC,BUF,MXB,IWLR,KWIT)
      IF(KWIT.NE.0) GO TO 990
  103 CONTINUE
      GO TO 990
  801 WRITE(10,8001)IV
 8001 FORMAT('0',I4,' BITS PADDING AT END OF SCAN LINE, TOO MUCH!')
  990 JJ=KWIT
      RETURN
      END
```

2.10. FORTRAN Subroutine LEDIT

2.10.1. Problem and Solution Method

A binary map may require editing to add or delete points or lines. For a single point, the user types in the row and column. The record (from row number) is read into core and the appropriate bit (from column number) is cleared (deleted) or set (added). The corrected record is written back in place on disc. For a line, the user types in the row and column for each of the two end points. It does not matter which point is typed in first. An array of points is constructed to approximate the line.

If the line is a true vertical or horizontal line, that is, the end points have the same row or column, the point array will be obvious to the user. If the line has a slope other than vertical or horizontal, the array of points is based on the slope and direction of the line. The resulting line, therefore, may not duplicate exactly an existing line to be deleted. This may not be important if the resulting data will be processed further in the WRIS system. A gap will enable the WRIS line-thinning routine to eliminate the line.

2.10.2. Input

The user responds to console prompts for yes or no, row or column input.

2.10.3. Processing

a. Processing logic. The x and y values of the end points provide the slope and direction of the line. The stepping value is set to plus- or minus-1 accordingly. The slope value (< 1) is used to increment or decrement the other coordinate. This process results in the array of coordinate pairs to describe a line with end points as given.

b. Linkages. LEDIT is called by LOOKED and calls YESNO, INVAL, MIN0, DSCRD, ICLR, ISET, and DSCWR.

c. Variables and constants. KK = 1 for deleting, 2 for adding point(s). JJ = 2 or 4, the number of integers expected for one or two points in INVAL. IRISE and IRUN are the vertical and horizontal vector magnitudes in integer format. RISE and RUN are the same values in floating-point format. TANG is the slope of the line. JJJ = column indicator (x value). KKK = row indicator (y value). N = number of correcting points in the array IVAL. VAR and J are used in applying the slope value to obtain the next x or y in the constructed line. J is also used with K to control disk I/O. K, IV, and L are used to calculate the word and the bit within the word to be corrected.

e. Error handling provisions. Point coordinates are checked to make sure they fall within the range of data. If data file is not a binary, map, an error message is printed and control returns to calling routine.

f. Restrictions and limitations. Data file must be a binary map (1 bit per pixel). MXPX is the maximum number of points in a constructed line. The maximum of (RISE, RUN), therefore, cannot exceed MXPX.

2.10.4. Output

Corrected data is written in place on disc. Console message confirms line correction.

2.10.5. Interfaces

IVAL, dimensioned MXPX, is the array to receive the x, y coordinates describing a point or line. BUF, dimensioned MXB, is used for I/O. IWLR is the logical record length. ICHD is channel number for the data file.

```
                  LEDIT
      COMPILER NOSTACK
      SUBROUTINE LEDIT(IVAL,MXPX,BUF,MXB,IWLR,ICHD)
C ALLOWS USER TO EDIT A BINARY MAP BY SPECIFYING ROW AND COLUMN OF POINT TO
C BE CHANGED.  LINES CAN BE SPECIFED BY ENTERING ONLY THE ROW AND COLUMN FOR
C EACH OF THE 2 END POINTS.
C
      INTEGER IVAL(2,MXPX),BUF(MXB)
      COMMON/H/IBP,NPX,NSL,NTH, INC,IXL,IYO,IYL,KIND
      IF(IBP.NE.1) GO TO 80
    5 CALL YESNO(IANS,'EDITTING - ANY DELETIONS',24)
      IF(IANS.NE.1) GO TO 10
      KK=1
      GO TO 20
   10 CALL YESNO(IANS,'EDITTING - ANY ADDITIONS',24)
      IF(IANS.NE.1) GO TO 90
      KK=2
   20 CALL YESNO(IANS,'SINGLE POINT',12)
      IF(IANS.NE.1) GO TO 25
      JJ=2
      CALL INVAL('TYPE ROW AND COLUMN',19,IVAL,JJ)
      N=1
      GO TO 27
   25 CALL YESNO(IANS,'LINE',4)
      IF(IANS.NE.1) GO TO 5
   26 JJ=4
      CALL INVAL('TYPE FIRST ROW & COLUMN AND LAST ROW & COLUMN',45,IVAL,JJ)
   27 K=JJ/2
      DO 29 J=1,K
      DO 28 I=1,2
      IF(IVAL(I,J).LT.1) GO TO 82
   28 CONTINUE
      IF(IVAL(1,J).GT.NSL) GO TO 82
      IF(IVAL(2,J).GT.NPX) GO TO 82
   29 CONTINUE
      IF(JJ.EQ.2) GO TO 61
```

```
C BUILD ARRAY OF INTERMEDIATE POINTS.
      IRISE=(IVAL(1,2)-IVAL(1,1))+1
      IRUN=(IVAL(2,2)-IVAL(2,1))+1
      IF(IRISE.LT.1) IRISE=IRISE-2
      IF(IRUN.LT.1) IRUN=IRUN-2
      RISE=IRISE
      RUN=IRUN
      K=1
      JJJ=IVAL(2,1)
      KKK=IVAL(1,1)
X     WRITE (10,7032)IRISE,IRUN,JJJ,KKK
X7032 FORMAT('0IRISE,IRUN,COL,ROW',4I7)
      IF(IABS(IRISE).GT.IABS(IRUN)) GO TO 35
      N=MIN0(IABS(IRUN),MXPX)
      IF(IRUN.LT.0) K=-1
      TANG=RISE/ABS(RUN)
      DO 30 I=1,N
      VAR=I-1
      J=VAR*TANG
      J=KKK+J
      IVAL(1,I)=J
      IVAL(2,I)=JJJ
      JJJ=JJJ+K
   30 CONTINUE
      GO TO 61
   35 N=MIN0(IABS(IRISE),MXPX)
      IF(IRISE.LT.0) K=-1
      TANG=RUN/ABS(RISE)
      DO 37 I=1,N
      VAR=I-1
      J=VAR*TANG
      J=JJJ+J
      IVAL(2,I)=J
      IVAL(1,I)=KKK
      KKK=KKK+K
   37 CONTINUE
C APPLY CORRECTION(S) TO DATA FILE.
   61 I=0
      J=-1
   70 I=I+1
      K=IVAL(1,I)-1
      IF(J.EQ.K) GO TO 71
      J=K
      NREC=1
      CALL DSCRD(ICHD,J,NREC,BUF,MXB,IWLR,KWIT)
      IF(KWIT.NE.0) GO TO 90
X     WRITE(10,7000)I,J,K
X7000 FORMAT('0I,J,K',3I6)
   71 K=IVAL(2,I)
      IV=(K+15)/16
      L=(IV*16)-K
      IVAR=BUF(IV)
      GO TO (72,74),KK
   72 CALL ICLR(IVAR,L)
      GO TO 79
   74 CALL ISET(IVAR,L)
   79 BUF(IV)=IVAR
      CALL DSCWR(ICHD,J,NREC,BUF,MXB,IWLR,KWIT)
      IF(I.LT.N) GO TO 70
      IF(N.GT.1) WRITE(10,1000)(IVAL(I,1),I=1,2),(IVAL(I,N),I=1,2)
 1000 FORMAT('0LINE DATA CORRECTED FROM ROW',I5,', COLUMN',I5,3X,'TO ROW',
        I5,', COLUMN',I5)
      GO TO 5
   80 TYPE 'THIS IS NOT A BINARY MAP; DO NOT TRY TO EDIT'
      GO TO 90
   82 TYPE 'EDIT PARAMETERS OUTSIDE OF MAP LIMITS'
      GO TO 26
   90 RETURN
      END
```

2.11. FORTRAN Main Program LOOKED

2.11.1. Problem and Solution Method

User needs a method of viewing selected portions of data and of editing data if necessary. This routine processes print requests and calls an editing subprogram.

2.11.2. Input

Program accepts yes or no responses and integers describing the map piece to be printed (row and column limits).

2.11.3. Processing

a. Processing logic. Print area controls are set according to the output device indicated by the user. Print formats are set at run time for the type of data to be printed. Lines of data exactly duplicating the previous print line can be ignored at user's option. Compact printing is available for data that are represented by I symbol per pixel. Column headings are printed at beginning of each map piece printed. Row numbers are printed at left edge of output line.

b. Linkages. LOOKED calls OPEN, TIME, DATE, TPIN, YESNO, INVAL, MAX0, MIN0, COZY, OUTC, CHARZ, DSCRD, UNPAK, LEDIT, and TPOUT.

c. Variables and constants. III = 0 if repetitive lines are ignored, 1 if repetitive lines are to be printed. KKK is the incremental parameter for printing the column headings. IV is the remaining number of rows to be printed in one piece. IVAR is the width, in columns, of the piece to be printed. K is the beginning column. KK is the end column. It is also used to indicate position in BUF. JJJ is the maximum number of records that can be contained in BUF. NN and LL indicate which half of array NPAKD is to be used. JJ and L count 1-bits in a print line. The remaining variables are in the glossary.

e. Error handling provisions. I/O status is checked; if error, processing stops. First and last row and column values are compared for reasonableness. User is notified of error and prompted for new values. Width and length of print piece are limited to device capacity, even if user's request is greater.

f. Restrictions and limitations. Program sets print limits for CRT (72 columns, 22 rows), wide-carriage typewriter terminal (120 columns, unlimited rows), and HP-2635A (216 columns for compact printing, 120 columns for standard printing, unlimited rows). Logical records cannot exceed 3000 words (MXB).

2.11.4. Output

Prompts are typed for yes, no, or integer supplied by user. User-selected map piece is printed on output device. Edited output is discussed in *2.10.* , the section on LEDIT.

2.11.5. Interfaces

Common block /H/ is shared with several subprograms.

```
                LOOKED
      COMPILER NOSTACK
      PARAMETER MXB=3000,ICHD=1,MXHED=53,ICHAN=0
      PARAMETER MXPX=216,NBW=16
C PRINTS USER-SELECTED PIECES OF A MATRIX.  USER CAN ALSO EDIT MATRIX IF THE
C PIXELS ARE 1 BIT EACH.  PIECE WIDTH IS NOT ALLOWED TO EXCEED PRINT LINE WIDTH
C OF OUTPUT DEVICE (MXCOL).  PIECE LENGTH (MXROW) IS NOT LIMITED UNLESS THE
C DISPLAY DEVICE IS A CRT.
C MXCOL IS ALSO USED AS A SWITCH TO SELECT STANDARD OR COMPACT PRINTING.
C PRINTING CAN BE SUPPRESSED IF LINE IS IDENTICAL TO THE PREVIOUS LINE.
      INTEGER JCOL(5),JUND(7),JDAT(8)
      INTEGER IFCOL(5,3),IFUND(7,3),IFDAT(8,4)
      INTEGER IVAL(4),NPAKD(MXPX,2),BUF(MXB)
      COMMON/A/INH80(3)
      INTEGER IHED(MXHED)
      COMMON/H/IBP,NPX,NSL,NTH,INC,IXL,IYO,IYL,KIND
      DATA IBL/' '/,IZL/'0'/
      DATA INH80/-1,2,40000K/
      DATA IPOS/1H'/,IDOT/1H./
C FORMATS FOR COLUMN HEADINGS.
      DATA IFCOL(1,1)/'(10I12)'/,IFCOL(1,2)/'(3X,15I8)'/
      DATA IFCOL(1,3)/'(2X,20I6)'/
C FORMATS FOR HEADING UNDERLINES.
      DATA IFUND(1,1)/'(9X,40(2X,A1))'/,IFUND(1,2)/'(7X,30(3X,A1))'/
      DATA IFUND(1,3)/'( 7X,  120A1)'/
C FORMATS FOR DATA LINES.
      DATA IFDAT(1,1)/'(I5,4X,40I3)'/,IFDAT(1,2)/'(I5,2X,30I4)'/
      DATA IFDAT(1,3)/'(I5,2X,120A1)'/,IFDAT(1,4)/'(I5,2X,120I1)'/
C
      CALL OPEN(10,'$TTO',INH80,IER)
      CALL TIME(IHED(48),IER)
      CALL DATE(IHED(51),IER)
      WRITE(10,5000)(IHED(I),I=48,53)
 5000 FORMAT('0TIME=',3I3,6X,'DATE=',I3,'-',I2,'-',I2)
      CALL TPIN(BUF,MXB,IHED,MXHED,IWLR,ICHAN,ICHD,KWIT)
      IF(KWIT.NE.0) GO TO 99
```

```
      IPW=NBW/IBP
      MXROW=0
      MXCOL=MXPX
      CALL YESNO(IANS,'WILL DISPLAY BE ON HP-2635A',27)
      IF(IANS.NE.1) GO TO 4
      IF(KIND.EQ.2) MXCOL=120
      GO TO 5
    4 MXCOL=120
      CALL YESNO(IANS,'WILL DISPLAY BE ON CRT',22)
      IF(IANS.NE.1) GO TO 5
      MXROW=22
      MXCOL=72
    5 III=0
      CALL YESNO(IANS,'PRINT REPETITIVE LINES',22)
      IF(IANS.EQ.1) III=1
      GO TO (10,20,10),KIND
C BINARY (OR ONE DIGIT OR SYMBOL PER PIXEL SPACE).
   10 ND=3
   11 NC=3
      NU=3
      KKK=6
      GO TO 30
C GROUPED, NUMERIC VALUES, UP TO 3 DIGITS PER PIXEL SPACE.
   20 ND=4
      IF(IBP.LT.4) GO TO 11
      IF(IBP.GT.6) GO TO 25
      NC=1
      NU=1
      ND=1
      KKK=4
      MXCOL=MXCOL/3
      GO TO 30
   25 KKK=2
      MXCOL=MXCOL/4
      NC=2
      NU=2
      ND=2
   30 IF(MXCOL.GT.132) GO TO 50
C SET UP FORMAT STATEMENTS FOR STANDARD PRINT LINES.
      DO 45 I=1,5
   45 JCOL(I)=IFCOL(I,NC)
      DO 46 I=1,7
   46 JUND(I)=IFUND(I,NU)
      DO 47 I=1,8
   47 JDAT(I)=IFDAT(I,ND)
   50 CALL YESNO(IANS,'DISPLAY A MAP SECTION',21)
      IF(IANS.NE.1) GO TO 70
      J=4
      CALL INVAL('FIRST AND LAST ROW, FIRST AND LAST COLUMN', 41,IVAL,J)
      IVAL(1)=MAX0(IVAL(1),1)
      IVAL(3)=MAX0(IVAL(3),1)
      IVAL(2)=MIN0(IVAL(2),NSL)
      IVAL(4)=MIN0(IVAL(4),NPX)
      IVAL(1)=IVAL(1)-1
      IV=IVAL(2)-IVAL(1)
      IF(IV.LT.1) GO TO 80
      IF(MXROW.GT.0) IV=MIN0(IV,MXROW)
      IVAR=IVAL(4)-IVAL(3)+1
      K=IVAL(3)
      IVAR=IVAR+KKK-MOD(IVAR,KKK)
      IVAR=MIN0(IVAR,MXCOL)
      IF(IVAR.LT.1) GO TO 80
      L=IVAR/KKK
      JJJ=KKK-1
      KK=IVAL(3)+IVAR-1
      NN=1
      LL=0
      IVAR=MIN0(IVAR,NPX)
      DO 150 I=1,IVAR
  150 NPAKD(I,NN)=0
      PAUSE 'POSITION PAGE, HIT ANY KEY'
C PRINT COLUMN HEADINGS AND UNDERLINE.
      IF(MXCOL.GT.132) GO TO 51
```

```
C STANDARD SPACING.
      WRITE(10,JCOL)(I,I=K,KK,KKK)
      WRITE(10,JUND)(IPOS,(IDOT,I=1,JJJ),J=1,L)
      GO TO 55
C COMPACT SPACING
   51 J=6
X     WRITE(10,6000)J,K,KK,KKK
X6000 FORMAT('0J,K,KK,KKK',4I6)
      CALL COZY(1)
      CALL OUTC(40K)
      CALL OUTC(40K)
      DO 52 I=K,KK,KK
      CALL CHARZ(I,BUF,J)
      DO 52 II=1,J
      CALL OUTC(BUF(II))
   52 CONTINUE
      CALL OUTC(15K)
      CALL OUTC(12K)
X     WRITE(10,6001)(BUF(I),I=1,6)
X6001 FORMAT('0BUF FROM CHARZ',6OI7)
C TO PRINT UNDERLINE TO COLUMN HEADINGS, IN COMPACTED FORMAT.
      DO 152 I=1,7
  152 CALL OUTC(40K)
      DO 53 J=1,L
      CALL OUTC(47K)
      DO 53 I=1,JJJ
      CALL OUTC(56K)
   53 CONTINUE
      CALL OUTC(15K)
      CALL OUTC(12K)
      CALL COZY(2)
C PRINT DATA LINES.
   55 JJJ=MXB/IWLR
      NREC=MIN0(JJJ,IV)
      I=IVAL(1)
      J=ICHD
      CALL DSCRD(ICHD,IVAL(1),NREC,BUF,MXB,IWLR,KWIT)
      IF(KWIT.NE.0) GO TO 90
X     WRITE(10,7000)(BUF(I),I=1,10)
X7000 FORMAT('0BUF FROM DISC',10OI8)
      DO 60 I=1,NREC
      NN=3-NN
      KK=(I-1)*IWLR+1
X     WRITE(10,7001)I,KK,IBP,K,IVAR
X7001 FORMAT('0I,KK,IBP,K,IVAR',5I7)
      CALL UNPAK(IBP,BUF(KK),NPAKD(1,NN),K,IVAR,IPW)
      IVAL(1)=IVAL(1)+1
      IF(KIND.NE.1) GO TO 258
      JJ=0
      DO 58 II=1,IVAR
      JJ=JJ+NPAKD(II,NN)
      IF(NPAKD(II,NN).EQ.0) NPAKD(II,NN)=IBL
      IF(NPAKD(II,NN).EQ.1) NPAKD(II,NN)=IZL
   58 CONTINUE
      IF(JJ.NE.L) GO TO 59
  258 IF(III.EQ.1) GO TO 59
      LL=3-NN
      DO 158 II=1,IVAR
      IF(NPAKD(II,NN).NE.NPAKD(II,LL)) GO TO 59
  158 CONTINUE
      GO TO 60
   59 IF(MXCOL.GT.132) GO TO 159
C PRINT LINE IN STANDARD FORMAT.
      WRITE(10,JDAT)IVAL(1),(NPAKD(II,NN),II=1,IVAR)
      GO TO 259
C TO PRINT DATA LINE IN COMPACTED FORMAT.
  159 J=5
      CALL COZY(1)
      CALL CHARZ(IVAL(1),BUF,J)
      DO 160 II=1,J
      CALL OUTC(BUF(II))
  160 CONTINUE
      CALL OUTC(40K)
      CALL OUTC(40K)
```

```
      L=0
      DO 161 II=1,IVAR
      IF(L.GE.JJ) GO TO 261
      IF(NPAKD(II,NN).EQ.IZL) GO TO 260
      CALL OUTC(40K)
      GO TO 161
  260 CALL OUTC(60K)
      L=L+1
  161 CONTINUE
  261 CALL OUTC(15K)
      CALL OUTC(12K)
      CALL COZY(?)
  259 L=JJ
   60 CONTINUE
      IV=IV-NREC
      IF(IV.GT.0) GO TO 55
C CALL ROUTINE TO EDIT BINARY MAP
   70 I=(MXB-IWLR)/2
      CALL LEDIT(BUF(IWLR+1),I,BUF,MXB,IWLR,ICHD)
      CALL YESNO(IANS,'ARE YOU DONE WITH THIS MAP',26)
      GO TO (90,50),IANS
C
C ERRORS.
   80 TYPE'ERROR IN MAP PIECE PARAMETERS'
      GO TO 50
   90 CALL TPOUT(BUF,MXB,IWLR,ICHAN,ICHD,IHED,MXHED,KWIT)
   99 WRITE(10,9900)
 9900 FORMAT(///////)
      STOP
      END
```

2.12. FORTRAN Subroutine MTREAD

2.12.1. Problem and Solution Method

The various activities and checks required for tape reading must be concentrated into one general-purpose program. Several Data General system routines intended for fast, direct reading are conveniently grouped into a utility program.

2.12.2. Input

Magnetic tape files are read using RDOS direct, free-format tape I/O programs.

2.12.3. Processing

a. Processing logic.

1. General logic. A detailed opening procedure is called on the first use of the subprogram. On subsequent calls, the direct reading proceeds until stopped by an end-of-file or an error condition.

2. Detailed logic. On the first call to the subprogram, the DG routine INIT performs partial initialization, required before file reference. A device name and an argument for indicating error conditions are supplied. The opening procedure is accomplished with a call to another DG routine, MTOPD. After opening and on subsequent calls, the DG free-format tape reading routine MTDIO is called to read data. It uses the designated channel, and reads a tape file until the end of a record is met. The number of words read into the array receiving the data (BUF) is returned as the last argument (N).

b. Linkages. The subprogram is called by RDHED and TPIN. It calls DG tape-handling routines INIT, MTOPD, and MTDIO. It uses the DG LAND function.

c. Variables and constants. With the exceptions listed below, constants and variables are defined once in the glossary to avoid redundancy. BUF is an array for I/O. Its dimension, MXB, should be at least 4095, the maximum DG tape record length. ICHAN = 0, a tape channel selected for this system. KWIT = 0 if no problem, 1 for tape EOF, 2 for tape opening or read error. MTOP = 0 if file is not open, 1 if file is open.

e. Error handling provisions. An error may be detected in any one of three DG routines called. An error during directory initialization (INIT) or opening a magnetic tape unit will cause the message "MTOPD error" to be printed with the error code. An error after calling the third, free-format reading routine (MTDIO), causes one of two actions. A test is made for an end-of-file and if found, file is closed and control returns to the calling program. Otherwise, a tape reading error message is printed with a status code, then control returns to caller.

f. Restrictions and limitations. The name of the directory file and the magnetic tape channel number may vary at each installation. DG tape records cannot exceed 4095 words.

2.12.4. Output

Error messages are output to the console.

2.12.5. Interfaces

This routine returns a tape record in BUF. It receives status and error messages from DG routines. The program also returns I/O status via the argument KWIT.

2.12.7. Run Description

Subprogram is to be located with calling routines and FORTRAN library. In the event of errors, codes will permit reference to explanations in Data General manuals.

```
                MTREAD
          COMPILER NOSTACK
         SUBROUTINE MTREAD(ICHAN,BUF,MXB,KWIT,N)
C  RDOS DIRECT BLOCK TAPE READ
C  MT 9 DEC 77, REVISED FEB 78 (J.DYE)
C
C  BUF = ARRAY TO RECEIVE DATA, LENGTH MXB
C  N RETURNS THE NUMBER OF WORDS ACTUALLY READ.
C  KWIT IS RETURNED AS:
C        0 - ALL OK
C        1 - TAPE EOF
C        2 - TAPE READ OR OPEN ERR
C
C  USES CHANNEL ICHAN, OPENS TO FIRST FILE ON TAPE
         INTEGER BUF(MXB)
C MASK TO CHECK FOR EOF.
         DATA MTEND/400K/
C INDICATOR FOR TAPE CHANNEL OPEN.
         DATA MTOP/0/
X        WRITE(10,6666)
X6666 FORMAT('0ENTERING MTREAD')
         KWIT=0
           IF(MTOP .GT. 0) GOTO 1
C  OPEN (ON FIRST CALL)
           CALL INIT("MT0",0,IER)
           CALL MTOPD(ICHAN,"MT0:0",0,IER)
           IF(IER .NE. 1) GOTO 99
         MTOP=1
C
C  READ
    1    CALL MTDIO(ICHAN,MXB,BUF,ISTA,IER,N)
X        WRITE(10,8889)ISTA,IER
X8889 FORMAT('0ISTA,IER',OI8,I4)
         IF(IER.EQ.1) RETURN
C TEST FOR EOF.
         IF(IAND(ISTA,MTEND).EQ.0) GO TO 8
         KWIT=1
         RETURN
C
C ERROR CONDITIONS.
C
    8 KWIT=2
           WRITE(10,10) ISTA
   10    FORMAT(/" **Tape read err, status =",OI6/)
         WRITE(10,1000)IER
 1000 FORMAT(' IER=',I4)
         RETURN
C
C OPEN ERROR.
   99 WRITE(10,11)IER
   11 FORMAT(/" **MTOPD error ",I3/)
         KWIT=2
         RETURN
         END
```

2.13. FORTRAN Subroutine MT WRIT

2.13.1. Problem and Solution Method

Data in a buffer are to be written on magnetic tape. This procedure provides the controls and status checks to write each record on tape. User types an integer in response to prompt for number of tape files to skip. User also types "yes" or "no" to signal that tape is ready.

2.13.3. Processing

a. Processing logic. The output file is initialized by DG routine INIT and opened by DG routine MTOPD. KWIT (furnished by caller) indicates whether to call INIT and MTOPD. The file name and data channel are defined in the program. Data files already on the tape can be skipped. Each record is written rapidly by using DG routine MTDIO for free-format I/O. When all records have been copied, caller can initiate two EOF's and tape rewind.

b. Linkages. The subprogram is called by TPOUT. It calls DG subroutines INIT, MTOPD, MTDIO, and ITEST. It calls PSW subroutine YESNO.

c. Variables and constants. Refer to the glossary for variables not described here. L = number of tape files to skip. LF= condition of a specified bit in the I/O status word. IV is the command word for writing the record on tape.

e. Error handling provisions. Error conditions can arise during the use of the DG routines INIT, MTOPD, and MTDIO. The error is not described in detail but a code is

printed for which a definition can be found in the system manuals.

f. Restrictions and limitations. Directory name and tape channel number may vary by system installation. JWPR cannot exceed 4095 words on DG.

2.13.4. Output

Outputs JWPR words of data in a physical record to magnetic tape at each call. Furnishes I/O error messages on console.

2.13.5. Interfaces

Receives I/O status and error messages from DG routines. KWIT, supplied by caller, is set to 1 for using INIT and MTOPD, 0 for writing data on tape, 2 to write 2 EOF's, and rewind tape. KWIT, returned by MTWRIT, indicates successful write (= 0) or error (= 1).

2.13.7. Run Description

Locate this subprogram in the appropriate directory along with FORTRAN library routines.

```
                MTWRIT
      COMPILER NOSTACK
      SUBROUTINE MTWRIT(ICHAN,BUF,MXB,JWPR,KWIT)
C WRITES A RECORD ON TAPE, OPENING FILE FIRST IF NECESSARY.
C ALSO WRITES DOUBLE EOF, REWINDS, AND CLOSES FILE.
C JWPR WORDS OF DATA ARE COPIED TO TAPE FROM ARRAY BUF.
C USER SETS KWIT BEFORE CALLING MTWRIT:
C   KWIT = -1 ON FIRST CALL TO OPEN FILE IF NECESSARY.
C   KWIT = 0 ON ALL CALLS TO WRITE DATA ON TAPE.
C   KWIT = -2 ON CALL TO WRITE EOF'S AND CLOSE FILE.
C AFTER EACH CALL, KWIT IS RETURNED: 0 = SUCCESSFUL CALL
C                                    1 = ERROR
C
C
      INTEGER BUF(MXB)
      LOGICAL LJ, ITEST
      DATA MTEOF,MTREW,MTWR,MTSKP/60000K,10000K,50000K,30000K/
X     WRITE(10,6666)
X6666 FORMAT('0ENTERING MTWRIT')
      IF(KWIT.GE.0) GO TO 19
      IF(KWIT.EQ.-2) GO TO 900
C INIT AND OPEN TAPE FILE.
    5 CALL INIT('MT0',0,IER)
      IF(IER.NE.1.AND.IER.NE.40) GO TO 800
      CALL MTOPD(ICHAN,'MT0:0',0,IER)
      IF(IER.NE.1) GO TO 801
      ACCEPT 'SKIP HOW MANY FILES (99 WILL ABORT JOB)',L
      IF(L.LE.0) GO TO 18
      IF(L.EQ.99) STOP
      DO 8 I=1,L
      II=I-1
      CALL MTDIO(ICHAN,MTSKP,BUF,ISTA,IER)
      LJ=ITEST(ISTA,15)
      IF(LJ.EQ.0) GO TO 8
      LJ=ITEST(ISTA,8)
      IF(LJ.GE.0) GO TO 802
    8 CONTINUE
C
C WRITE A RECORD.
   18 CALL YESNO(IANS,'READY',5)
      IF(IANS.NE.1) GO TO 5
   19 IV=JWPR+MTWR
      CALL MTDIO(ICHAN,IV,BUF,ISTA,IER)
      IF(IER.NE.1) GO TO 803
      KWIT=0
      RETURN
C
C WRITE EOF'S, REWIND, CLOSE FILE.
  900 CALL MTDIO(ICHAN,MTEOF,BUF,ISTA,IER)
      IF(IER.NE.1) GO TO 803
      CALL MTDIO(ICHAN,MTEOF,BUF,ISTA,IER)
      IF(IER.NE.1) GO TO 803
      CALL MTDIO(ICHAN, MTREW,BUF,ISTA,IER)
      IF(IER.NE.1) GO TO 803
      KWIT=0
      RETURN
C
C ERROR CONDITIONS.
  800 WRITE(10,8000)IER
 8000 FORMAT('0TAPE INIT ERROR',I6)
      GO TO 890
```

```
  801 WRITE(10,8001)IER
 8001 FORMAT('0MTOPD ERROR',I6)
      GO TO 890
  802 TYPE 'ERROR AFTER FILE NO',II
      GO TO 890
  803 WRITE(10,8003)IER
 8003 FORMAT('0MTDIO ERROR',I5)
  890 KWIT=1
      RETURN
      END
```

2.14. Assembly Subroutine OUTC

2.14.1. Problem and Solution Method

An essential feature of a system that includes data display is the ability to transmit individual characters to the output device. This routine interfaces with DG systems to output a character.

2.14.3. Processing

b. Linkages. OUTC is called by COZY and LOOKED.

c. Variables and constants. CHAR = stack location of first (and only) argument.

2.14.4. Output

One character is transmitted to the output device. It can be a character to be printed or a control character, such as backspace or linefeed.

2.14.5. Interfaces

The only argument is the character in ASCII code for the DG system.

```
                 OUTC
        .TITL   OUTC
        .ENT    OUTC
; SUBROUTINE TO OUTPUT ONE CHARATER TO CONSOLE
; THE FORM OF THE FORTRAN CALL IS 'CALL OUTC(CHAR)'.
        .EXTD   .FRET,.CPYL
        .TXTM   1
        .NREL
A0=0
A1=1
A2=2
A3=3
;
CHAR=-167       ;STACK LOC OF FIRST ARG
;
; ENTRY
        1       ;STACK SIZE
OUTC:   JSR     @.CPYL  ;FETCH ARG ADDRESSES TO LOCAL STACK
        STA     A3,SA3  ;SAVE OUR STACK PTR
        LDA     A0,@CHAR,A3   ;FETCH CHARACTER
        .SYSTM
        .PCHAR  ;WRITE CHARACTER TO CONSOLE
        JMP     .+1     ;IGNORE ERROR RETURN
        LDA     A3,SA3  ;RESTORE A3 (NEED?)
        JSR     @.FRET  ;RETURN TO CALLER
;
SA3:    0       ;SAVE OUR STACK PTR
;
        .END
```

2.15. FORTRAN Subroutine PAK

2.15.1. Problem and Solution Method

Whenever practical, data should be stored in packed form. This routine packs integer numbers (representing pixels) into an array according to the number of bits per pixel.

2.15.3. Processing

a. Processing logic. Caller gives (J) the relative position to start in the packed array, (KK) the number of elements per packed word, and (JBP) the number of bits per element. From these values, the subroutine calculates the beginning word position in the packed array (BUF), and the beginning bit position within that word. An outer DO-loop is based on N, the number of elements to be packed. Within this loop the packed word position is incremented as needed and the starting bit position is initialized. An inner DO-loop, based on JBP, tests each bit of the incoming data element and sets or clears the bit accordingly.

b. Linkages. PAK is called by SLICE and FREAK and uses DG subprograms ITEST, ICLR, and ISET.

c. Variables and constants. IV indicates the word of the packed array currently receiving data. K indicates the starting bit position for the current packed data element. JJ indicates the current bit being set or cleared in the packed array. II indicates the current bit being tested in the incoming element.

2.15.5. Interfaces

Refer to *section 2.15.3.a.*

```
                PAK
      COMPILER NOSTACK
      SUBROUTINE PAK(JBP,BUF,NPAKD,J,N,KK)
C PACKS N INTEGER VALUES OF JBP BITS EACH INTO ARRAY BUF, STARTING AT
C THE JTH POSITION IN BUF.  VALUES ARE TAKEN FROM ARRAY NPAKD (ONE PER WORD).
C KK = THE NUMBER OF VALUES PER WORD OF ARRAY BUF.
C ANY UNUSED BITS IN EACH WORD OF BUF ARE THE LEFT (HIGH-ORDER) BITS.
      INTEGER BUF(1),NPAKD(1)
      LOGICAL ITEST,LJ
      IV=(J+KK-1)/KK
      K=JBP*(IV*KK-J)
      DO 40 I=1,N
      JJ=K
      DO 31 L=1,JBP
      II=L-1
      CALL ICLR(BUF(IV),JJ)
      LJ=(ITEST(NPAKD(I),II))
      IF(LJ.EQ.-1) CALL ISET(BUF(IV),JJ)
      JJ=JJ+1
   31 CONTINUE
      IF(K.GT.0) GO TO 35
      K=KK*JBP
      IV=IV+1
   35 K=K-JBP
   40 CONTINUE
      RETURN
      END
```

2.16. FORTRAN Subroutine RDHED

2.16.1. Problem and Solution Method

a. Problem. The Scandig system can produce multiple files of raster-scanned data on magnetic tape. An interactive search procedure is needed to find the desired file for further processing. Parameters describing the file are also needed.

b. Solution method. Each scanner tape file is identified by a descriptive header record. The user finds the desired file interactively, reading header records and skipping over unneeded files.

c. Alternative solution. Another choice is to read the header from disc if the desired data are available there.

2.16.2. Input

User supplies an integer value for the number of files to skip. Value can be zero.

2.16.3. Processing.

a. Processing logic. This routine controls the reading of a header record from tape or disc. The information contained therein is deciphered and displayed to the user with a prompt. If tape files, the user can specify the necessary number of tape files to space forward to reach the desired data.

b. Linkages. The routine is called by TPIN and, in turn, calls YESNO, DSCRD, MTREAD, MTDIO, ICONV, and logical function ITEST.

c. Variables and constants. WAR = record length expected from a tape or disc read. NN = number of ASCII character positions assigned to each integer value in the header record. IV = number of tape files to skip.

e. Error handling provisions. In addition to the normal check of I/O error status, several likely mistakes are anticipated and corrective action defined. These errors are in tape header length, type of file, or positioning resulting from skipping too many files.

f. Restrictions and limitations. Tape files are skipped over in a forward direction only. Backspacing is judged unnecessary. The user can restart if the wrong number of skips is made.

2.16.4. Output

The contents of the desired file header record are printed.

2.16.5. Interfaces

ICH is furnished by caller. Common block /H/, array IHED, and KWIT are passed to caller. KWIT is returned from calls to DSCRD, MTREAD, and ICONV.

2.16.7. Run Description

This subroutine returns to calling program for run control.

```
                RDHED
      COMPILER NOSTACK
    PARAMETER MXSCN=46, ICHAN=0
      SUBROUTINE RDHED(ICH,IHED,MXHED,KWIT)
C  FROM TAPE OR DISC, READS HEADER RECORD AND DISPLAYS CONTENTS.
```

```
C  IF INPUT IS FROM TAPE, SKIPS FILES AS DIRECTED.
C  CONVERTS HEADER INFO AND STORES IN COMMON/H/.
C   *mt*  12 DEC 77, REVISED MARCH, JULY, AND SEPT. 1978, BY J. DYE
C
      LOGICAL LJ,ITEST
      INTEGER IHED(MXHED)
      COMMON/H/IBP,NPX,NSL,NTH,INC,IXL,IYO,IYL,KIND
      DATA MTSKP/30000K/
        DATA IZZ/2H00/
C
      KIND=0
C  READ HEADER
      IVAR=MXHED
      IF(ICH.GT.ICHAN) GO TO 4
      CALL YESNO(IANS,'EXPANDED HEADER RECORD',22)
      IF(IANS.NE.1) IVAR=MXSCN
       GO TO 5
    4 IREC=0
      NREC=1
      CALL DSCRD(ICH,IREC,NREC,IHED,MXHED,IVAR,KWIT)
      IF(KWIT.NE.0) GO TO 899
      KIND=IHED(47)
      GO TO 10
    5 CALL MTREAD(ICH,IHED,IVAR,KWIT,IWPR)
      L=MXHED
X     WRITE(10,5005)L,ICH,KWIT,IWPR,(IHED(I),I=1,MXHED)
X5005 FORMAT('0RDHED',4I6,6(//9(0I7)))
        IF(KWIT .NE. 0) GO TO 899
C
CCHECK BEGINS "0000"
        IF(IHED(1) .NE. IZZ .OR. IHED(2) .NE. IZZ) GOTO 802
   10 WRITE(10,1011)
 1011   FORMAT(//" Scandig tape header record"/
     $   " --------------------------")
C  NOW DECODE AND DISPLAY FIELDS
      NN=4
      KWIT=0
        CALL ICONV(IHED(3),MXHED,ISEQ,NN,KWIT,"Sequence     ")
        WRITE(10,1012)(IHED(K),K=5,28)

CC
 1012   FORMAT(" Forest: ",12A2/" Layer: ",12A2)
        CALL ICONV(IHED(29),MXHED,IMAP,NN,KWIT,"Map          ",
        CALL ICONV(IHED(31),MXHED,IBP,NN,KWIT,"Bits/Pixel  ")
        CALL ICONV(IHED(33),MXHED,NPX,NN,KWIT,"Pixels/line ")
        CALL ICONV(IHED(35),MXHED,NSL,NN,KWIT,"Scan lines  ")
        CALL ICONV(IHED(37),MXHED,NTH,NN,KWIT,"Threshold   ")
        CALL ICONV(IHED(39),MXHED,INC,NN,KWIT,"Increment   ")
        CALL ICONV(IHED(41),MXHED,IXL,NN,KWIT,"X Length mm ")
        CALL ICONV(IHED(43),MXHED,IYO,NN,KWIT,"Y Offset mm ")
        CALL ICONV(IHED(45),MXHED,IYL,NN,KWIT,"Y Length mm ")
      WRITE(10,1014)
 1014 FORMAT("--------------------------"//)
      IF(KIND.EQ.0) GO TO 11
      TYPE 'KIND OF FILE = ',KIND
      WRITE (10,2000)(IHED(I),I=48,53)
 2000 FORMAT('0WRITTEN',I3,':',I2,':',I2,4X,3I3)

      GO TO 12
   11 KIND=1
      IF(IBP.GT.1) KIND=2
      IF(NTH.EQ.0.AND.IBP.EQ.8) KIND=3
C IS THIS THE DESIRED TAPE AND/OR FILE?
   12 IF(ICH.NE.0) GO TO 990
      ACCEPT'SKIP HOW MANY TAPE FILES? (99 WILL ABORT JOB): ',IV
      IF(IV.EQ.0) GO TO 990
      IF(IV.EQ.99) STOP
      TYPE 'ICH',ICH
      DO 15 I=1,IV
      CALL MTDIO(ICH,MTSKP,IHED,ISTA,IER)
      LJ=ITEST(ISTA,15)
      IF(LJ.EQ.0) GO TO 15
      LJ=ITEST(ISTA,8)
      IF(LJ.GE.0) GO TO 803
   15 CONTINUE
      GO TO 5
```

```
C
C
C ERROR CONDITIONS.
C
C
C TAPE HEADER RECORD ERROR.
  802 WRITE(10,8002)IHED(1),IHED(2),IWPR
 8002 FORMAT('0HEADER BEGINS (OCTAL)',2OI8,', CONTAINS',I6,' WORDS')
      GO TO 899
C TAPE POSITION ERROR.
  803 WRITE(10,8003)ISTA,IER
 8003 FORMAT('0TAPE POSITIONING ERROR, STATUS AND ERROR=',OI8,I4)
  899 KWIT=1
  990 RETURN
        END
```

2.17. FORTRAN Main Program SLICE

2.17.1. Problem and Solution Method

Density values in a Scandig-produced file may range from 0 through 255. Data is combined into fewer groups according to user specifications of group edges and number.

2.17.2. Input

Optional input from console includes low and high observations, number of groups, group lower edge values, and symbols for representing the groups for graphic display.

2.17.3. Processing

a. Processing logic. Header record and data are copied to disc if not already there. Header record contains parameters for calculating default values for group edges (LEDGE) and high observation (IGH). Defaults are also set for low observation (LOW), the number of groups (NG), group low edge values, and group symbols (ISYM). User is prompted for optional values to replace the defaults of any of these parameters. For only two groups of data, the low edge of the second group is either the default—Mode (NTH) divided by 2—or user-specified. For more than two groups, low edges are based on range and number of groups. For graphic display, data can be represented by group number or a symbol representing the group. If the number of groups, NG, is less than eleven, the default symbols are single digits from zero to NG. If the number of groups exceeds ten, default symbols are assigned sequentially, starting at ASCII code 32 and ending at ASCII code 95, for a maximum of 64. The maximum number of bits per data element (JBP) is calculated from the number of groups. In turn, the number of data elements per word (JPW), the output logical record length (JWLR), and the bytes per record (IV) are calculated. A new disc file is initialized and opened to receive the reduced data. Each data record is copied from disc to core and unpacked, I data element per word. Each data element is replaced by the group number corresponding to its value or by a symbol if one is to be used. Data is repacked, padded if necessary to a multiple of 32 bits in the record, and copied to disc. Header record is corrected to show any change in the number of data elements per record (NPX). The original file is closed so it can be reopened with the new record length. If the reduced data are binary, bit values are complemented so lines are represented by 1-bits. The reduced data are copied to the original file area for use by another program, or for copy to tape.

b. Linkages. SLICE calls OPEN, TIME, DATE, TPIN, YESNO, INVAL, INIT, DSCRD, UNPAK, HEW, PAK, DSCWR, FSTAT, CLOSE, DFILW, INVERS, TPOUT, and RESET. It uses subfunctions MAX0, MIN0, and MOD.

c. Variables and constants. VAR = width of one group (RANGE/NG). IVAR is the ASCII code or character used for possible padding of records. The remainder are defined in *section 2.17.3a* or the glossary.

e. Error handling provisions. I/O errors result in program termination. If input file is a binary map file, an error message is printed and program stops. If input record length exceeds program array, error message is printed and program stops. If user-furnished grouping parameters exceed array capacity or are inconsistent, error message is printed, and user is given choice of stopping or furnishing new parameters. LOW is not allowed to be less than zero; IGH is not allowed to be greater than (2**IBP)-l. NG cannot exceed 2**MXBT (currently 64).

f. **Restrictions and limitations.** Array dimensions may need to be increased for larger records.

2.17.4. Output

Numerous prompting and error messages may be printed on console. Reduced data are placed in original file for further use.

```
                SLICE
C MAIN ROUTINE FOR DATA REDUCTION ACCORDING TO DEFAULT OR USER-SELECTED
C PARAMETERS.  NUMBER OF GROUPS (NG) CAN VARY FROM 2 TO MXG (64).  DEFAULT
C IS 10.  LOW VALUE DEFAULT IS 0, HIGH VALUE DEFAULT IS (2**IBP)-1.
C NG CANNOT EXCEED RANGE.
      COMPILER NOSTACK
      PARAMETER MXHED=53,ICHAN=0
      PARAMETER MXG=64,MXBT=6,ICHD=1,ICH2=5,MXPX=4200
      PARAMETER MXB=3000,NBW=16
```

```
      INTEGER IHED(MXHED),IEDGE(MXG),ISYM(MXG)
      INTEGER BUF(MXPX)
      COMMON/A/INH80(3)
      COMMON/H/IBP,NPX,NSL,NTH,INC,IXL,IYO,IYL,KIND
      DATA INH80/-1,2,40000K/
      DATA MASK8/377K/
C
      CALL OPEN(10,'$TTO',INH80,IER)
      CALL TIME(IHED(48),IER)
      CALL DATE(IHED(51),IER)
      WRITE(10,5000)(IHED(I),I=48,53)
 5000 FORMAT('0TIME=',3I3,6X,'DATE=',I3,':',I2,':',I2)
      CALL TPIN(BUF,MXB,IHED,MXHED,IWLR,ICHAN,ICHD,KWIT)
      IF(KWIT.NE.0) GO TO 990
      IF(IBP.EQ.1) GO TO 800
      IF(NPX.GT.MXPX) GO TO 801
C SET DEFAULTS FOR NUMBER OF GROUPS, HIGH AND LOW VALUES.
    8 LOW=0
      IGH=(2**IBP)-1
      NG=10
      CALL YESNO(IANS,'USE DEFAULT LOW AND HIGH OBSERVATIONS',37)
      IF(IANS.EQ.1) GO TO 9
      J=0
      N=2
      CALL INVAL(J,J,BUF,N)
      LOW=MAX0(LOW,BUF(1))
      IGH=MIN0(IGH,BUF(2))
    9 RANGE=IGH-LOW+1
      CALL YESNO(IANS,'USE DEFAULT FOR NUMBER OF GROUPS',33)
      IF(IANS.EQ.1) GO TO 11
      N=1
      J=0
      CALL INVAL(J,J,NG,N)
   11 IF(NG.GT.MXG) GO TO 803
      IF(NG.GT.IFIX(RANGE)) GO TO 803
      IF(NG.LE.1) GO TO 803
C SET GROUP EDGES.
      IEDGE(1)=LOW
      IEDGE(2)=NTH/2
      CALL YESNO(IANS,'USE DEFAULT GROUP EDGES',23)
      IF(IANS.EQ.1) GO TO 12
      J=37
      N=NG-1
      CALL INVAL('LOWER EDGE OF EACH GROUP EXCEPT FIRST',J,IEDGE(2),N)
      GO TO 14
   12 VAR=RANGE/NG
      J=2
      IF(NG.LE.J) GO TO 14
      DO 13 I=2,NG
      IEDGE(I)=IEDGE(1)+(I-1)*VAR
   13 CONTINUE
   14 WRITE(10,2000)LOW,IGH,NG

 2000 FORMAT('0LOW=',I4,3X,'HIGH=',I4,3X,'NO.GROUPS=',I3)
      WRITE(10,2001)(IEDGE(I),I=1,NG)
 2001 FORMAT('0GROUP EDGES:',4(/,10X,16I5))
      IF(IEDGE(2).GT.IGH) GO TO 803
      IF(IEDGE(NG).LE.LOW) GO TO 803
      DO 213 I=2,NG
      IF(IEDGE(I).LE.IEDGE(I-1)) GO TO 803
  213 CONTINUE
      CALL YESNO(IANS,'PARAMETERS OKAY',15)
      IF(IANS.NE.1) GO TO 8
      CALL YESNO(IANS,'USE GROUP SYMBOLS',17)
      IF(IANS.NE.1) GO TO 114
      J=47
      IF(NG.GT.10) J=31
      DO 111 I=1,NG
      ISYM(I)=I+J
  111 CONTINUE
      CALL YESNO(IANS,'USE DEFAULT SYMBOLS',19)
      IF(IANS.EQ.1) GO TO 113
      WRITE(10,1014) NG
 1014 FORMAT('0TYPE',I3,' SYMBOLS, WITH NO SPACES OR PUNCTUATION')
      READ(11,1015)(ISYM(I),I=1,NG)
 1015 FORMAT(64A1)
```

```
C SET OUTPUT PACKING PARAMETERS (BITS/VALUE, VALUES/WORD).
  113 JBP=8
      IVAR=40K
      GO TO 20
  114 DO 15 I=1,MXBT
      IF(NG.GT.2**I) GO TO 15
      JBP=I
      IVAR=(2**I)-1
      GO TO 19
   15 CONTINUE
      GO TO 803
   19 IF(JBP.GE.IBP) GO TO 803
   20 JPW=NBW/JBP
      JWLR=(NPX+JPW-1)/JPW
      IF(MOD(JWLR,2).NE.0) JWLR=JWLR+1
      IF(JWLR.GT.MXB) GO TO 801
C ESTABLISH DISC FILE FOR WRITING OUTPUT LOGICAL RECORDS.
      IV=JWLR*2
      CALL INIT('DP0F',0,IER)
      TYPE 'INIT DP0F',IER
      CALL OPEN(ICH2,'DP0F:5',6,IER,IV)
      TYPE 'OPEN DP0F',IER
      IF(KWIT.GT.0) GO TO 990
C GROUP DATA.
      IPW=NBW/IBP
      NREC=1
      II=(JWLR*JPW)-NPX
      IV=1
      LLL=MXPX-II+1
      KKK=MXPX-IWLR+1
      CALL TIME(BUF,IER)
      IHOUR=BUF(1)
       BUF(1)=BUF(3)+BUF(2)*60
       TYPE 'BEGIN SLICING',BUF(1)
      DO 40 I=1,NSL
      IREC=I-1
      CALL DSCRD(ICHD,IREC,NREC,BUF(KKK),IWLR,IWLR,KWIT)


      IF(KWIT.GT.0) GO TO 990
      CALL UNPAK(IBP,BUF(KKK),BUF(2),IV,NPX,IPW)
      CALL HEW(BUF(2),NPX,NG,IEDGE)
      IF(JBP.LE.6) GO TO 34
      DO 31 J=1,NPX
      L=J+1
      K=BUF(L)+1
      BUF(L)=ISYM(K)
   31 CONTINUE
   34 CALL PAK(JBP,BUF,BUF(2),IV,NPX,JPW)
C TO PAD OUTPUT RECORD FOR POSSIBLE 1108 USE.
      IF(II.LE.0) GO TO 39
      JJ=NPX+1
      DO 36 J=LLL,MXPX
      BUF(J)=IVAR
   36 CONTINUE
      CALL PAK(JBP,BUF,BUF(LLL),JJ,II,JPW)
   39 CALL DSCWR(ICH2,IREC,NREC,BUF,MXB,JWLR,KWIT)
   40 CONTINUE
      CALL TIME(BUF,IER)
      BUF(1)=BUF(1)-IHOUR
      BUF(1)=BUF(1)*3600+BUF(2)*60+BUF(3)
       TYPE 'END SLICING',BUF(1)
      IF(II.LE.0) GO TO 46
C CORRECT NPX IN HEADER RECORD TO REFLECT ADDITIONAL PADDING.
      NPX=NPX+II
      IV=NPX
      DO 42 I=1,4
      J=5-I
      BUF(J)=MOD(IV,10)+48
      IV=IV/10
   42 CONTINUE
      J=8
      I=1
      K=4
      II=2
      CALL PAK(J,IHED(33),BUF,I,K,II)
```

```
   46 IHED(47)=2
      IF(JBP.EQ.1) IHED(47)=1
      IF(JBP.EQ.8) IHED(47)=3
      KIND=IHED(47)
      IHED(32)=IHED(32)-IBP+JBP
      IBP=JBP
C COPY DATA TO REGULAR OUTPUT FILE AREA.
      CALL FSTAT(ICHD,0,IER)
      TYPE 'FSTAT ICHD',IER
      CALL CLOSE(ICHD,IER)
      TYPE 'CLOSE ICHD',IER
      CALL DFILW('1',IER)
      TYPE 'DFILW 1',IER
      JJ=2*JWLR
      CALL OPEN(ICHD,'1',2,IER,JJ)
      TYPE 'REOPEN ICHD',IER
      NREC=1
       TYPE 'BEGIN COPY'
      DO 60 I=1,NSL
      IREC=I-1
      CALL DSCRD(ICH2,IREC,NREC,BUF,MXB,JWLR,KWIT)
      IF(KWIT.NE.0) GO TO 990
      CALL DSCWR(ICHD,IREC,NREC,BUF,MXB,JWLR,KWIT)
      IF(KWIT.NE.0) GO TO 990
   60 CONTINUE
      TYPE 'END COPY'
      IF(JBP.NE.1) GO TO 65
C IF A BINARY MAP, REVERSE THE IMAGE SO ONE-BITS REPRESENT LINES (LOW
CDENSITY.
      JJ=2
       TYPE 'BEGIN INVERS'
      CALL INVERS(BUF,MXB,ICHD,JWLR,JJ)
       TYPE 'END INVERS'
      IF(JJ.NE.0) GO TO 990
   65 CALL TPOUT(BUF,MXB,JWLR,ICHAN,ICHD,IHED,MXHED,KWIT)
      GO TO 990
  800 WRITE(10,8000)IBP
 8000 FORMAT('0WRONG INPUT FILE, BITS PER PIXEL=',I2)
      GO TO 990
  801 TYPE 'TOO MANY PIXELS/ROW TO PROCESS'
      GO TO 990
  803 CALL YESNO(IANS,'ERROR IN PARAMETERS, ABANDON JOB',32)
      IF(IANS.NE.1) GO TO 8
  990 CALL RESET
      WRITE(10,1000)
 1000 FORMAT(//////)
      STOP
      END
```

2.18. FORTRAN Subroutine STATIC

2.18.1 Problem and Solution Method

Given a frequency distribution, the user may need more details about it. This routine finds the mode and range (high and low observations).

2.18.3. Processing

a. Processing logic. The mode is found by setting a variable (IV) equal to the count of the first group, and K equal to the group value, then comparing the count of each successive group with IV. If any count exceeds IV, then IV is reset to that count, and K is reset to that group number. This routine is part of a specialized system and group values are assumed to be from 0 through LL-1, where LL is the number of groups. The minimum observation (LOW) is the first non-zero count encountered in a loop starting at group one. Similarly, the maximum observation is the first non-zero count encountered in a loop starting at the highest group and working down.

b. Linkages. STATIC is called by FREAK.

f. Restrictions and limitations. Group values are assumed to be consecutive integers starting with zero.

2.18.4. Output

The low and high observations and the mode are printed.

2.18.5. Interfaces

IFR, dimensioned LL, is the array of group counts. K returns the value of the mode. IGH and LOW return the extreme observations.

```
                STATIC
      COMPILER NOSTACK
      SUBROUTINE STATIC(IFR,LL,IGH,LOW,K)
C FINDS MODE (K), MINIMUM (LOW) AND MAXIMUM (IGH) VALUES IN A FREQUENCY
C TABLE (IFR).
      INTEGER IFR(LL)
      K=0
      IV=IFR(1)
C FIND MODE.
      DO 10 I=2,LL
      IF(IFR(I).LT.IV) GO TO 10
      IV=IFR(I)
      K=I-1
   10 CONTINUE
C FIND MINIMUM VALUE.
      DO 20 I=1,LL
      IF(IFR(I).EQ.0) GO TO 20
      LOW=I-1
      GO TO 26
   20 CONTINUE
      LOW=LL-1
C FIND MAXIMUM VALUE.
   26 DO 30 I=1,LL
      J=LL+1-I
      IF(IFR(J).LE.0) GO TO 30
      IGH=J-1
      GO TO 40
   30 CONTINUE
      IGH=0
   40 WRITE(10,1000) LOW,IGH,K
 1000 FORMAT('0RANGE=',I3,' TO',I4,5X,'MODE=',I3)
      RETURN
      END
```

2.19. FORTRAN Subroutine TPIN

2.19.1. Problem and Solution Method

A control routine is useful for several main processors to call for preparing files and initializing. This routine manages tape-to-disc copy or verifies files' readiness.

2.19.2. Input

Header record and data file are copied from tape unless already on disc, in which case the header is copied from disc. Keyboard response from user indicates location of data.

2.19.3. Processing

a. Processing logic. Message on console asks for input data location. Channel number is set according to tape or disc input. Header record is obtained. If header came from disc, process ing skips tape copying. If not, the disc files are opened and the header record copied to disc. A loop controls tape reading and disc writing of the data. Tape is rewound and closed. User is given option of reversing image.

b. Linkages. TPIN is called by FREAK, LOOKED, and SLICE. It calls YESNO, RDHED, OPEN, DSCWR, MTREAD, MTDIO, CLOSE, and INVERS.

e. Error handling provisions. I/O operations return a status code to be checked.

2.19.4. Output

The console displays messages, including one notifying user how many records were copied to disc.

2.19.5. Interfaces

All parameters are included in the glossary.

```
                TPIN
      COMPILER NOSTACK
      PARAMETER NBW=16,ICHH=2
      SUBROUTINE TPIN(BUF,MXB,IHED,MXHED,IWLR,ICHAN,ICHD,KWIT)
C COPIES TAPE FILE ONTO DISC, UNLESS DIRECTED TO USE FILES ALREADY ON DISC.
C BUF IS AN ARRAY LARGE ENOUGH TO HOLD MXB WORDS.  MXB NEED NOT BE
C MORE THAN 4095 FOR NOVA COMPUTERS.  MXB SHOULD BE 3000 WHEN READING RECORDS
C FROM A TAPE WRITTEN BY THE SCANDIG.
C IWLR INDICATES NUMBER OF WORDS PER SCAN LINE;IE,A LOGICAL RECORD.
C ICHD IS THE RELATIVE CHANNEL NUMBER FOR THE DISC DATA FILE.
C ICHAN IS THE RELATIVE CHANNEL NUMBER FOR THE TAPE UNIT.
C KWIT (RETURNED) INDICATES ACCOMPLISHMENT:
C       0 = SUCCESSFUL COPY
C       1 = ERROR
       INTEGER BUF(MXB),IHED(MXHED)
      COMMON/H/IBP,NPX,NSL,NTH,INC,IXL,IYO,IYL,KIND
```

```
X     WRITE(10,6666)
X6666 FORMAT('0ENTERING TAPIN')
      KWIT=0
      IREC=0
      ICH=ICHAN
      CALL YESNO(IANS,'USE DISC FILE',13)
      IF(IANS.NE.1) GO TO 7
      ICH=ICHH
C READ HEADER RECORD.
    7 CALL RDHED(ICH,IHED,MXHED,KWIT)
      IF(KWIT.NE.0) GO TO 990
      I=NBW/IBP
      IWLR=(NPX+I-1)/I
      IF(ICH.NE.ICHAN) GO TO 980
      ICH=ICHH
      I=MXHED*2
      CALL OPEN(ICH,'2',3,IER,I)
      TYPE 'OPEN ICH',IER
      I=2*IWLR
      CALL OPEN(ICHD,'1',2,IER,I)
      TYPE 'OPEN ICHD',IER
      IREC=0
      NREC=1
      CALL DSCWR(ICH,IREC,NREC,IHED,MXHED,MXHED,KWIT)
      IF(KWIT.NE.0) GO TO 801
      I=MXB/IWLR
      I=I*IWLR
C COPY DATA RECORDS FROM TAPE TO DISC.
   10 CALL MTREAD(ICHAN,BUF,I,KWIT,IWPR)
      IF(KWIT.GT.0) GO TO 800
      NREC=IWPR/IWLR
      CALL DSCWR(ICHD,IREC,NREC,BUF,MXB,IWLR,KWIT)
      IF(KWIT.NE.0) GO TO 990
      IREC=IREC+NREC
      GO TO 10
C
C ALERTS AND/OR ERROR CONDITIONS.
  800 KWIT=KWIT-1
  801 IF( KWIT.GT.1) KWIT=1
      WRITE(10,9000) IREC
 9000 FORMAT('0',I8,' RECORDS COPIED FROM TAPE TO DISC')
      NSL=NSL+1
      IHED(36)=IHED(36)+1
C REWIND TAPE AND CLOSE FILE.
      CALL MTDIO(ICHAN,10000K,BUF,ISTA,IER)
      IF(IER.NE.1) WRITE(10,8006) IER
 8006 FORMAT('0TAPE REWIND ERROR',I4)
      CALL CLOSE(0,IER)
C ALLOW USER TO REVERSE IMAGE IF NECESSARY.
  980 CALL YESNO(IANS,'DOES DATA REPRESENT A PHOTO NEGATIVE',36)
      IF (IANS.EQ.1) GO TO 990
      KWIT=1
        CALL INVERS(BUF,MXB,ICHD,IWLR,KWIT)
  990 RETURN
      END
```

2.20. FORTRAN Subroutine TPOUT

2.20.1. Problem and Solution Method

At the end of a processing program, the need may arise to update, rewrite files, or both. This subprogram provides control of lower-level routines to handle these tasks.

2.20.3. Processing

a. Processing logic.

1. General logic. Program allows interactive direction from the user or operator. If output tape is specified, blocks of records are copied from core to magnetic tape. The tape file is opened if necessary, written, terminated with a double EOF, rewound, and closed. The subprograms DSCRD and MTWRIT provide rapid data transfer by calling specialized DG subroutines.

2. Detailed logic. The first step is to determine the maximum physical tape record length (which is 4095 words [MXTAP] in the present configuration). The next step is to calculate the number of logical records that will fit in MXB or 4095 words, whichever is less. The process is more easily understood with a numerical example: suppose the maximum number of words for a physical output tape record (LL) is 28. If one divides this by 9 (the number of words in an output logical record) the result is 3, or the number of logical records that will fit in a physical record. Once these parameters are established,

the total number of records to be copied, NSL, can be read from the disc in groups of 3 into core, then passed to tape in maximum-sized blocks.

b. Linkages. TPOUT is called by FREAK, LOOKED, and SLICE. It uses subprograms YESNO, DSCWR, MTWRIT, MIN0, DSCRD, TIME, and DATE.

c. Constants and variables. Only three temporary variables are defined here, the rest are defined in the glossary of variables to avoid redundancy. JJ = maximum number of logical records in a physical record. IV = running total of logical records. LL = maximum physical tape record.

e. Error handling provisions. Various error status indicators associated with I/O subroutines are tested after use.

f. **Restrictions and limitations.** Buffer sizes must be set correctly by calling routine. Buffer must be at least as large as a logical record. Ideally, it is several times as large.

i. Optimization. Because large amounts of data will normally be handled by this routine, efficiency is important. The subprograms called also, in turn, call DG subroutines for precise I/O control.

2.20.4. Output

Error and status messages are printed on console. Logical records copied from disc to tape are the final product. Data file and header record remain available on disc also.

2.20.7. Run Description

Locate the calling routines and the FORTRAN library in the same directory. The header record can be the standard, 46-word header as written by the Scandig routine or expanded to 53 words. Either type can be read by TPIN. When the output tape is intended for use with the WRIS routines, the header must be in the 46-word format.

```
          TPOUT
      COMPILER NOSTACK
      PARAMETER ICHH=2,MXSCN=46,MXTAP=4095
      SUBROUTINE TPOUT(BUF,MXB,JWLR,ICHAN,ICHD,IHED,MXHED,KWIT)
C INITIALIZES AND OPENS OUTPUT TAPE FILE, IF NECESSARY.
C COPIES BLOCKS OF RECORDS FROM DISC TO TAPE.
C WRITES DOUBLE EOF, REWINDS TAPE, AND CLOSES FILE.
C WRITES UPDATED HEADER RECORD ON DISC.
C BUF IS AN ARRAY LARGE ENOUGH TO HOLD MXB WORDS.
C MXB NEED NOT BE MORE THAN MXTAP (4095) FOR DG SYSTEMS.
C JWLR = NUMBER OF WORDS IN A LOGICAL TAPE RECORD.  JWLR CANNOT
C BE LARGER THAN MXB.  PHYSICAL TAPE RECORD IS A MULTIPLE OF JWLR BUT
C NOT GREATER THAN MXTAP (4095).
C ICHD INDICATES THE CHANNEL NUMBER FOR THE DISC FILE.
C KWIT (RETURNED) INDICATES ACCOMPLISHMENT:
C       0 = SUCCESSFUL COPY
C       1 = DISC READ ERROR
C       2 = OUTPUT TAPE ERROR
C
      INTEGER BUF(MXB),IHED(MXHED)
      COMMON/H/IBP,NPX,NSL,NTH,INC,IXL,IYO,IYL,KIND

C
X     WRITE(10,6666)
X6666 FORMAT('0ENTERING TAPOUT')
      CALL TIME(IHED(48),IER)
      CALL DATE(IHED(51),IER)
      IHED(47)=KIND
C INITIALIZE VARIABLES.
      KWIT=0
      IV=0
C SET ISTA FOR FIRST CALL TO MTWRIT.
      ISTA=-1
```

2.21. FORTRAN Subroutine UNPAK

2.21.1. Problem and Solution Method

Individual elements of a data collection packed two or more per word must be made available for processing. This routine unpacks a subset of the data and stores the data values one per element of an array.

2.21.3. Processing

a. Processing logic. Calculation of IV, the word in BUF containing the initial value (J) to be unpacked, is based on J and on KK, the number of packed values per word. K, the bit shift indicator, is initially calculated from IBP times the relative position of the initial value in word number IV. A loop, controlled by the number of unpacked values (N), isolates each value by ANDing a mask of IBP bits with the shifted (by K bits) packed word BUF (IV). IV is incremented and K is decremented or initialized as necessary, as processing continues.

b. Linkages. UNPAK is called by SLICE, FREAK, and LOOKED. It uses DG functions IAND and ISHFT.

c. Variables and constants. MASK is an array of words containing 1-bits in the low order positions. The number of bits varies from 1 through 8. High order bits are zeros.

f. Restrictions and limitations. NPAKD must be dimensioned to at least N in the calling routine.

2.21.5. Interfaces

BUF is the input array of packed data. NPAKD is the output array of individual values. J is the first packed value needed from BUF. N is the number of values to unpack. KK is the number of packed values per word of BUF.

```
            UNPAK
      COMPILER NOSTACK
      SUBROUTINE UNPAK(IBP,BUF,NPAKD,J,N,KK)
C UNPACKS N CONSECUTIVE INTEGER VALUES OF IBP BITS EACH FROM ARRAY BUF,
C STARTING WITH THE JTH VALUE.  STORES VALUES, ONE PER WORD, IN ARRAY NPAKD.
C ANY UNUSED BITS IN EACH WORD OF BUF ARE ASSUMED TO BE THE LEFT (HIGH-
C ORDER) BITS.  KK IS THE NUMBER OF PACKED VALUES PER WORD IN BUF.
      INTEGER BUF(1),NPAKD(1),MASK(8)
      DATA MASK/1K,3K,7K,17K,37K,77K,177K,377K/
      IV=(J+KK-1)/KK
      K=IBP*(IV*KK-J)
      DO 40 I=1,N
      NPAKD(I)=IAND(MASK(IBP),ISHFT(BUF(IV),-K))
      IF(K.GT.0) GO TO 35
      K=KK*IBP
      IV=IV+1
   35 K=K-IBP
   40 CONTINUE
      RETURN
      END


      CALL YESNO(IANS,'WRITE OUTPUT TAPE',17)
      IF(IANS.NE.1) GO TO 910
      CALL YESNO(IANS,'IS OUTPUT TAPE READY',20)
      IF(IANS.NE.1) GO TO 9
      L=MXHED
      CALL YESNO(IANS,'EXPANDED HEADER RECORD',22)
      IF(IANS.NE.1) L=MXSCN
      CALL MTWRIT(ICHAN,IHED,MXHED,L,ISTA)
      IF(ISTA.NE.0) GO TO 804
C MAXIMUM PHYSICAL RECORD LENGTH IS MXTAP (4095) OR DIMENSION OF BUF.
      LL=MIN0(MXTAP,MXB)
C FIGURE NUMBER OF LOGICAL RECORDS IN MAXIMUM TAPE RECORD.
      JJ=LL/JWLR
      IF(JJ.LT.1) GO TO 802
C COPY RECORDS.
      DO 20 I=1,NSL,JJ
      IREC=I-1
      NREC=MIN0(JJ,NSL-IREC)
      CALL DSCRD(ICHD,IREC,NREC,BUF,MXB,JWLR,KWIT)
      IF(KWIT.NE.0) GO TO 990
      JWPR=NREC*JWLR
      CALL MTWRIT(ICHAN,BUF,MXB,JWPR,ISTA)
      IF(ISTA.NE.0) GO TO 804
      IV=IV+NREC
   20 CONTINUE
      GO TO 900

  ERROR CONDITIONS.

  802 WRITE(10,8002)JWLR,LL
 8002 FORMAT('0LOGICAL RECORD TOO LARGE, =',I6,2X,'PHYSICAL RECORD LENGTH=',I4)
  804 KWIT=2
      GO TO 990
C
C DONE WITH TAPE.
C
  900 ISTA=-2
      CALL MTWRIT(ICHAN,BUF,MXB,JWPR,ISTA)
C REWRITE HEADER ON DISC.
  910 L=0
      J=1
      CALL DSCWR(ICHH,L,J,IHED,MXHED,MXHED,KWIT)
      IF(KWIT.NE.0) TYPE 'ERROR IN REWRITING HEADER RECORD ON DISC',KWIT
  990 WRITE (10,1000)IV
```

```
 1000 FORMAT('0',I8,' LOGICAL RECORDS COPIED FROM DISC TO TAPE')
      RETURN
      END
R
```

2.22. FORTRAN Subroutine YESNO

2.22.1. Problem and Solution Method

In an interactive system, there are numerous cases of yes-or-no questions to the user. This routine handles the I/O with the console.

2.22.2. Input

User responds with a typed Y or N to each question.

2.22.3. Processing

a. Processing logic. The calling routine passes a character string containing the question. The string is printed on the console. User types in a Y or an N. If the response is not valid, the routine prompts user for Y or N. If the response is valid, IANS is set accordingly.

b. Linkages. YESNO is called by FREAK, LOOKED, SLICE, RDHED, INVAL, TPOUT, TPIN, MTWRIT, and LEDIT. It uses DG function MIN0.

c. Variables and constants. L is the maximum number of words in the question. J is the actual number of words in the question. II is a Hollerith question mark, added to print line for each question.

f. Restrictions and limitations. Question (in LABEL) is limited to 76 characters (38 words on DG).

2.22.4. Output

Questions on console and a prompt for a 'Y' or 'N'.

2.22.5. Interfaces

LABEL is a character array containing a question from calling routine. N is the number of characters. TANS is returned.

```
                YESNO
      COMPILER NOSTACK
      SUBROUTINE YESNO(IANS,LABEL,N)
C  LABEL IS A STRING ARRAY CONTAINING A QUESTION TO BE TYPED ON THE OUTPUT
C  DEVICE.  N IS THE NUMBER OF CHARACTERS IN LABEL.  A MAXIMUM OF 76 WILL BE
C  PRINTED.
C  RETURNS IANS = 1 FOR YES
C          IANS = 2 FOR NO
      INTEGER YES,LABEL(1)
      DATA II/'? '/
      DATA YES/'Y '/,NO/'N '/
      IANS=0
      J=(N+1)/2
      L=38
      J=MIN0(L,J)
      WRITE(10,1000)(LABEL(I),I=1,J),II
 1000 FORMAT('0',39A2)
   10 READ(11,2000)J
 2000 FORMAT(A1)
      IF(J.EQ.YES)IANS=1
      IF(J.EQ.NO) IANS=2
      IF(IANS.GT.0) GO TO 90
      TYPE 'TYPE Y OR N'
      GO TO 10
   90 RETURN
      END
```

3. LITERATURE CITED

Amidon, Elliot L.
1978. **Computer mapping systems for integrated resource inventories.** *In* Integrated Inventories of Renewable Natural Resources: Proceedings of the Workshop Jan. 8-12, 1978 (Tucson, Ariz.). Gen. Tech. Rep. RM-55. Rocky Mountain Forest and Range Exp. Sm., Fort Collins, Colo., p. 354-359.

Deschene, Wallace A.
1981. **User's manual: RID*POLY Geographic Information System.** Gen. Tech. Rep. INT- 105, 144 p. Intermountain Forest and Range Exp. Stn., Forest Serv., U.S. Dep. Agric., Ogden, Utah.

Russell, Robert M., David A. Sharpnack, and Elliot L. Amidon.
1975. **Wildland Resource Information System: user's guide.** USDA Forest Serv. Gen. Tech. Rep. PSW-10, 36 p., illus. Pacific Southwest Forest and Range Exp. Stn., Berkeley, Calif.

U.S. Department of Commerce, National Bureau of Standards.
1976. Guidelines for documentation of computer programs and automated data systems. U.S. Dep. Commerce Federal Processing Standards Publ. 38. 55 p.

Amidon, Elliot L., and E. Joyce Dye.
1981. **SCANIT: centralized digitizing of forest resource maps or photographs.** Gen. Tech. Rep. PSW-53, 42 p., illus. Pacific Southwest Forest and Range Exp. Stn., Forest Serv., U.S. Dep. Agric., Berkeley, Calif.

Spatial data on wildland resource maps and aerial photographs can be analyzed by computer after digitizing. SCANIT is a computerized system for encoding such data in digital form. The system, consisting of a collection of computer programs and subroutines, provides a powerful and versatile tool for a variety of resource analyses. SCANIT also may be converted easily to another computer system. As a highly automated system, SCANIT requires a substantial investment in specialized instruments and assumes a central facility with a raster scanner and a minicomputer to serve many users. The system's designers, by adopting a highly interactive approach to the control of the data processing, have provided for persons with little or no computer experience to use the system and have minimized the need for a user manual. This report on SCANIT is designed to aid the maintenance programmer locate needed information easily and to assure an adequate level of documentation for each routine.

Retrieval Terms: aerial photographs, forest resource maps, computer programs, SCANIT, digitizers