2

DRA

[redacted]

(NASA-CR-120378) OPERATORS MANUAL FOR MICRODENSITOMETER CONTROL PROGRAM DENSITOMETER MODEL PDS-1010G (MODIFIED). PROGRAM TRACE VERSION (Lockheed Missiles and Space Co.) ~~27~~ 86 p HC $8.00 CSCL 14B

N74-30893

G3/14

Unclas
17093

Lockheed

MISSILES & SPACE COMPANY, INC.

A SUBSIDIARY OF LOCKHEED AIRCRAFT CORPORATION

SUNNYVALE, CA

LMSC D402244

FINAL REPORT

Section III

OPERATORS MANUAL
FOR
MICRODENSITOMETER CONTROL PROGRAM
DENSITOMETER MODEL PDS-1010G (MODIFIED)

PROGRAM TRACE

VERSION 3B

May 1974

Prepared for

George C. Marshall Space Flight Center
Huntsville, Alabama 35812

Contract No. NAS8-28018

Principal Investigator: Dr. A. M. Title

Lockheed Solar Observatory
Lockheed Palo Alto Research Laboratory
3251 Hanover Street
Palo Alto, California 94304

OPERATOR'S MANUAL FOR MICRODENSITOMETER CONTROL PROGRAM

I. Introduction

The PDS-1010G microdensitometer is run under the control of a PDP-11 program called TRACE. This program gives the operator very flexible control over the machine functions. Most commands are passed to the computer through either the Tektronix 4010 terminal or the teletype, as selected by the position of the LOCAL/LINE rocker switch above the 4010 keyboard. (LINE places the 4010 in control; LOCAL transfers control to the teletype. In general, the teletype is used when the operator desires a permanent record of the operator-computer dialogue.) A small number of control functions are requested by setting switches on the computer front panel.

II. Starting the Computer

1. Turn the key on the computer front panel clockwise from OFF to POWER.

2. Be sure that the disk LOAD/RUN switch is in the LOAD position. Move the disk OFF/ON switch to ON. Wait for the LOAD light to come on (about 10 seconds).

3. Move the disk LOAD/RUN switch to RUN. Wait for the RDY light to come on.

4. Be sure the HALT/ENABLE switch on the computer front panel is in the HALT position.

5. Set the SWITCH REGISTER (switches numbered 0-17) to octal 773100 (up is 1, down is 0).

6. Press the LOAD ADDR switch.

7. Set the SWITCH REGISTER to octal 777406.

8. Move the HALT/ENABLE switch to ENABLE.

9. Press the START switch. The system will then identify itself on the terminal.

Computer Shutdown Procedure

1. Move the HALT/ENABLE switch to HALT.

2. Move the disk LOAD/RUN switch to LOAD. You do <u>not</u> have to wait for the LOAD light.

3. Move the disk OFF/ON switch to OFF.

4. Turn the POWER key to OFF.

PRECEDING PAGE BLANK NOT FILMED

III. Loading the Program

1. After the system monitor has identified itself (DOS V08A), it prints a $. (DOS always uses $ to indicate that it is waiting for a command from the keyboard. All commands which the user types are terminated with a carriage return.)

2. Type in today's adte by typing DA , followed by the date in the format DD-MMM-YY. (Example: DA 23-JUN-73) The date must be given to the computer, because it is automatically added to the identification label on each data record.

3. After DOS prints a $, type LO N_g,N_u (where N_g,N_u is your assigned user number) to log onto the machine. DOS then prints the date (which you specified in step 2) and a meaningless time.

4. After DOS prints a $, type RU TRACE to load and run the densitometer control program. TRACE identifies itself and instructs the user to put the INCREMENT CONTROL SWITCH (on the densitometer interface panel) in the AUTO position. It then prints MONITOR on the terminal, followed by an asterisk. This indicates that it is waiting for an operator instruction.

Note: The SELECT switch on the densitometer control panel must be in AUTO and the motor power switches must be on before an attempt is made to move the carriage under computer control.

IV. Keyboard Commands

When TRACE MONITOR prints an asterisk (*) on the terminal, the operator may issue a command by typing one, or occasionally two, keys on the keyboard. TRACE then takes appropriate action. The following commands are currently available:

COMMAND		MEANING	DESCRIPTION
CTRL/C	-	Exit to monitor	IV.1
A	-	A-to-D converter test	IV.2
B	-	Beginning of tape	IV.3
C	-	Current coordinates	IV.4
D	-	Diode calibration	IV.5
F	-	Free disk blocks	IV.6
Gn	-	Go to coordinate set n	IV.7
H	-	go to Home	IV.8
I	-	Identification label	IV.9
K	-	Kill Autolok	IV.10
Ln	-	Load coordinate set n	IV.11
M	-	Message	IV.12
P	-	Playback	IV.13
Q	-	Quick load coordinate set n	IV.14
R	-	Rewind	IV.15
S	-	Scan	IV.16
T	-	Test scan	IV.17
U	-	User-defined scan parameters	V
X,Y	-	go to	IV.18
Z	-	Zero current coordinates	IV.19

IV.1: CTRL/C - Exit to monitor

The effect of hitting C while the CTRL key (at the left-hand side of the keyboard) is depressed depends on the status of the program.

If TRACE MONITOR has printed an asterisk on the terminal and is waiting for a keyboard command, CTRL/C causes program TRACE to terminate. Control passes to DOS, which prints a $ on the terminal. To reload TRACE, see step 4 of Section III.

If program TRACE is waiting for any other keyboard input, CTRL/C aborts that input and causes an immediate jump to TRACE MONITOR. A new command can then be given.

IV.2: A - A/D converter test

The operator can get a direct reading of the 10-bit A/D converter by giving the A command. The decimal result is printed on the terminal.

IV.3: B - Beginning of tape

A clean magtape must have a logical end-of-tape (double endfile) at its beginning before DOS can write on it. The B command will provide this formatting mark. (Magtape must previously have been specified as the storage device - see Section V.)

It is not necessary to write an endfile or end-of-tape after outputting data. DOS automatically provides these for you.

IV.4: C - Current Coordinates

After typing a C, you will be requested to supply the coordinates of the present position of the densitometer carriage. TRACE will print

```
CURRENT X:
```

on the terminal. The user then types in a decimal number of up to six digits, which may be preceeded by a minus sign. The number is terminated by hitting the RETURN key. TRACE then prints

```
CURRENT Y:
```

on the terminal. The user then supplies the value of the Y-coordinate in the same manner. Each time the RETURN is hit, the number just typed in should be displayed in the appropriate set of lights on the densitometer interface panel below the computer disk.

Note: If you wish to current position to be the origin, i.e., coordinates (0,0), you can easily accomplish that by giving TRACE MONITOR the Z command (see Section IV -19).

IV.5: D - Diode calibration

The current feeding the light-emitting diode should be calibrated before PLAYBACK mode is entered. Since the digital-to-analogue converter has an 8-bit register, it can accept digital inputs between 0 and 255. Thus, when

```
DIODE CURRENT:
```

is printed on the terminal, the user supplies a number in this range (terminated with a RETURN). The current which this number produces is indicated on the 3-digit lighted display on the densitometer interface panel. These values run between 0 and about 6.83. The user may use the GAIN and OFFSET controls near the display lights to produce required dynamic range. Since this process generally takes several tries, TRACE will continue to request diode current levels until the user hits the LINE FEED key, at which time control returns to MONITOR.

IV.6: F - Free disk blocks

When the F command is used, TRACE prints on the terminal the number of blocks on the disk, out of a total of 4800, which are available for data storage.

IV.7: Gn - Go to coordinate set n

When Gn (where n is an integer between 1 and 8) is typed, the carriage will move to the coordinates previously specified by the corresponding Ln command (see Section IV.11).

IV.8: H - move to Home

Typing an H causes the densitometer carriage to move to its (0,0) position. The bell or beeper on the terminal is rung when origin has been reached.

IV.9: I - Identification label

The first record of each scan is a string of characters which serves to identify the scan. If the user has not specified a label, the string UNIDENTIFIED SCAN is used. Once the user specifies an ident string, the most recent string specified is used.

After the operator gives the "I" command, the message

```
        IDENT:
```

is printed on the terminal. The user may then type up to 40 characters, terminating with a RETURN. Today's date, as given to DOS with the DA command (see Section III, step 2), is automatically appended to the end of the label.

Editing: The last character typed can be deleted by hitting the RUBOUT key. The deleted character will be echoed between slashes. Successive RUBOUT's are permitted. To restart the entire text, type CTRL/U (i.e., hit U while holding the CTRL key down.

IV.10: K - Kill Autolok

If the operator wishes to manually move the densitometer carriage, he should first use the K command to disable Autolok, which keeps the carriage locked to a specified position.

IV.11: Ln - Load coordinate set n

When Ln (where n is an integer between 1 and 8) is typed, TRACE will then request a pair of coordinates. Each of the coordinates may be up to six digits long, may be preceeded by a minus sign, and is terminated with a RETURN. The carriage will then move to those coordinates any time the corresponding Gn command is given.

IV.12: M - Message

A message, or comment, may optionally be written out as the second record of a data file. A comment differs from the ident label in two important respects:

a) The comment may be of virtually any length, and it may contain imbedded CR's. The message is terminated with the LINE FEED key.

b) The comment is written out only in the <u>next</u> data file. An ident string is written out in every data file.

The message COMMENT: is printed on the terminal, after which the user may enter his comment. Editing is the same as described in Section IV.9.

Note: The message is stored in the program's data buffer. When a scan is requested, the comment is written out onto the output file before the data-taking begins. However, <u>any</u> intervening operation which uses the data buffer will destroy the message. This includes the P(playback), T (test scan) and F (free disk) commands. The message should be entered just before the S (scan) command is given.

IV.13: P - Playback

The P command is used to create a photograph from digital data. The light-emitting diode should have been previously calibrated (see Section IV-5). The TEST/OPERATE switch on the densitometer interface panel must be in the OPERATE position. The SCAN/PLAYBACK switch on the densitometer control panel must be in the PLAYBACK position. The LAMP switch on the control panel must be ON.

The computer prints: SCALE FACTOR

The user responds with a decimal number between 1 and 100, followed by a RETURN. The scale factor allows the user to magnify a frame up to 100 times without changing the film density. If a value N has been specified as the scale factor, each data value is used for N consecutive points, and each line is printed N times. Note, however, that the PTS/LINE and the number of LINES specified in the scan parameters refer to the direct data, not the scaled frame. TRACE handles the scaling.

The computer prints: PARAMETER SOURCE? (K OR R)

The user responds with one of those two letters. K means that the user-specified scan parameters have been entered through the keyboard. R means that they are to be read from the input file. If K is typed and a complete set of scan parameters has not previously been typed in, TRACE goes immediately to that routine (see Section V).

The computer then prints: FILE NAME:

The user must now type the complete name of the input data file, terminating it with a RETURN.

The computer then prints: TYPE ANY KEY TO CONTINUE

Now darken the room and place the unexposed film on the densitometer platten. Then hit any key on the keyboard to begin the playback. The bell on the terminal is rung at the completion of the playback.

IV.14; Qn - Quick load coordinate set n

When Qn (where n is an integer between 1 and 8) is typed, the current coordinates are stored as destination set n. The carriage will return to its current position whenever the corresponding Gn command is given.

IV.15: R - Rewind magtape

Magtape must previously have been specified as the storage device (see Section V) before this command can be executed.

IV.16: S - Scan

The S command causes the densitometer to scan and digitize according to the previously entered user scan parameters. If a set of parameters has not previously been entered, TRACE goes immediately to that routine (see Section V).

TRACE first creates and opens a uniquely-named data file. It then prints the file name on the terminal. If a new identification has been entered since the last time the S command was used, the ident label is also printed on the terminal.

The densitometer carriage returns to the origin and the terminal bell rings at the completion of the scan.

IV.17: T - Test scan

A test scan is identical to a data scan (section IV.16) except that an output file is not created.

IV.18: X,Y - go to

These commands can be used to move the carriage to any location. MONITOR prints a message requesting the destination. The user then types a decimal number which may be preceeded by a minus sign and is terminated with a RETURN. The carriage will immediately move to the requested position. Note that the use of one of these commands does not obligate you to use the other also.

IV.19: Z - Zero

The Z command causes the current position of the carriage to become the origin. It does not move the carriage.

V. User-Defined Scan Parameters

The parameters which control the densitometer scan are entered by means of a dialogue with the computer. In the dialogue which follows, the symbol (CR) means that the user terminates his input by hitting the RETURN key. The steps marked with an asterisk are omitted if a line scan (pattern L) has been requested. (See section IV.9 for editing rules.)

COMPUTER	USER
X-DIR	R for right, or L for left. This refers to the scan direction on the film, <u>not</u> the direction in which the carriage moves.
Y-DIR	F for front, or B for back. This refers to the scan direction on the film and is the same as the direction in which the yoke moves.
PATTERN	E (edge scan): All lines traced in the same direction. B (boustrophedonic): Alternate lines traced in opposite directions. R (raster): Same as B, but data order is reversed for even-numbered lines, so data look as if edge scan was performed. L (line scan): Scan will be series of arbitrarily positioned lines.
DELTA X	The distance between digitized points, in microns (CR).
PTS/LINE	The number of points digitized on each line (CR).
Y STEP*	The distance between lines, in microns (CR).
LINES*	The number of lines in each frame (CR).
FRAME n X = Y =	The user specifies a pair of numbers which will be the starting coordinates of a frame. Each number may be up to six digits long and may be preceeded by a minus sign (CR). Up to 32 coordinate pairs may be specified. Each pair will be the starting coordinates of a different, identically shaped frame. All such frames will be scanned when the scan command (S) is given to TRACE MONITOR, and all of the data will be put in one data file. The computer continues to request coordinate pairs (up to 32) until the user types the LINE FEED key.

PRECEDING PAGE BLANK NOT FILMED

The first time the U command is given to TRACE MONITOR, the entire dialogue described above is carried out. Any additional U commands cause the computer to print OPTION:, and the user responds with one of the following:

U	- Complete dialogue
X	- X-direction
Y	- Y-direction
P	- Pattern
DX	- Delta X
DY	- Y Step
NP	- Points per line
NL	- Lines per frame
C<CR>	- Change all starting coordinates
CN	- Change a particular pair of coordinates
CX	- Set all X-start coordinates to the same value
CY	- Set all Y-start coordinates to the same value
B	- Backup
F	- Fortran compatibility
L	- Letter for output filename
M	- Storage medium
S	- Scale increment switch
V	- Speed
E	- Exit to TRACE MONITOR
#	- Number for output filename

If the requested option is part of the complete dialogue shown above, the appropriate piece of the dialogue will be carried out. (If DELTA X is specified, PTS/LINE will also be requested by the computer.) The other options work as follows:

OPTION	COMPUTER	USER
CN	FRAME	The user types a one or two digit number <CR> which must not exceed the number of frames previously specified in the complete dialogue. The computer will then request a single pair of coordinates, which the user provides <CR>.
CX CY	X= Y=	The user types in a coordinate <CR> which will be used as the X-start or Y-start coordinate for all of the requested frames.
B	BACKUP?	Y or N. Requesting BACKUP means that the specified X-start coordinate will be interpreted as the middle of the scan line rather than the beginning of it. Default: NO.
F	FTN I/O?	Y or N. Should the data be written out in PDP-11 Fortran-compatible form, or in a somewhat simpler format? Default: YES. (See Section IX for details.)
L	SERIES LETTER:	The user types a single letter. This will be used as the first character of the output filename. Default: A. (See Section VIII for details.)
M	STORAGE	D (disk) or M or T (tape or magtape). This is the device on which the data file will be created or found. Default: DISK.

OPTION	COMPUTER	USER
S		This sets the scale increment step to unity. This provides slightly better positional accuracy but may slow down the scan speed. (See Section VII for details.)
V	SPEED	The user types in a number between 1 and 255 (CR). The computer will reject the number if it is above the maximum speed allowed for the specified DELTA X and line length. (See Section VII for details.)
#	NUMBER:	The user types in a number between 0 and 99 (CR). The number will be used as the number part of the data filename the next time a data scan is made. Note that changing the series letter automatically resets the number to unity.

Note that all parameters remain set until they are changed. They do NOT revert to their default values.

VI. Sense Switch Options

A few functions can be invoked by setting (lifting) switches on the computer front panel. The currently available options are:

Switch ϕ - Runaway control

Occasionally, the densitometer looses track of where it is and begins to run away. Setting switch ϕ will stop the motors. When the switch is reset (down), the carriage will head fro its original destination, unless switch 4 has also been set. In that case, control transfers to TRACE MONITOR.

Switch 3 - Display data

At the end of any line, the data in the data buffer (up to the first 300 points) will be printed on the terminal if switch 3 is set. At the first such request in a scan, DISPLAY REQUESTED is printed on the terminal, and the computer waits until any key on the keyboard is hist. This is to give the user time to record the file name. After that, the screen is automatically cleared (if the 4010 terminal is being used) and the data is displayed. The data is held on the screen and the scan is stopped until any key on the keyboard is hit.

Switch 4 - Abort scan

At the end of any line, the entire scan will be terminated and the data file closed if switch 4 is found to be set. The message SCAN ABORTED is printed on the terminal.

VII. Scale Increment and Carriage Speed

While the densitometer carriage is capable of moving up to 200,000 microns/sec., the interface electronics are able to count no more than 50,000 position encoder pulses per second. Therefore, a SCALE INCREMENT switch is provided which allows the interface to see each pulse (1), every other pulse (2), or every fourth pulse (4). Since the densitometer knows its position to only within a scale increment, a setting of 1 must be used if extreme positional accuracy (i.e., ± one micron) is required. Note that the distance between points is equally accurate at all settings.

When the INCREMENT CONTROL switch on the densitometer interface panel is in the MANUAL position, the scale increment is set with the SCALE INCREMENT switch to its left. When the INCREMENT CONTROL switch is in the AUTO position, scale increment selection is handled by the computer. The AUTO position must be used when PROGRAM TRACE is running, and the MANUAL position must be used when it is not.

PROGRAM TRACE selects the largest scale increment compatible with the user-supplied DELTA X. It then chooses the optimum carriage speed, which is determined by the scale increment and by the length of the scan line. The use can minimize the time required to scan a frame (particularly a large one) by choosing a DELTA X which is evenly divisible by 4.

PRECEDING PAGE BLANK NOT FILMED

VIII. Input/Output Structure

Program TRACE has been written to run under the control of the PDP-11 disk operating system (DOS). Thus, the file structuring and read/write operations are handled automatically by DOS and need not concern the user. The file structure has been designed to meet the requirements of disk storage, but it is also fully compatible with direct transfers to or from magtape.

When the user requests a data scan (S command), the system first creates a uniquely named file. The filename consists of a letter and a one or two digit number, followed by the extension .DAT. Thus, a typical filename might be B37.DAT. The user can specify the letter by using OPTION L (see Section V). The number is automatically reset to unity when a new series letter is requested.

All filenames in a user's directory must be unique. If the program attempts to create a file which already exists, the attempt will fail. However, the program has a built-in facility for searching for unused names. It first tries changing the number, then the letter. Since there are 26 x 99 possibilities, success is assured. If the data file has a different name than you expected, it is probably because a conflict occurred.

If the disk is being used as the primary storage device, some care must be taken not to fill it up. Especially if large data arrays are being stored, the disk should be purged fairly often. The data files should be transferred from disk to magtape, after which they should be deleted. Both operations are done by program PIP. PIP will also tell you how much disk space (in blocks) is free if you give it the /FR switch. The disk holds a total of 4800 blocks.

IX. Data Format

The data are stored on the disk in formatted binary linked files. At the user's option, the records can be written in Fortran-compatable (hereafter abbreviated as FC) or non-Fortran-compatable (NFC) form. NFC records are somewhat simpler and are recommended if the user plans to process his data with an assembly language program or at a facility where the Fortran file structure differs from that of the PDP-11. If processing is to be done with PDP-11 Fortran, FC form is strongly recommended. (Note: only one word is allotted per variable, so compile your Fortran programs with the /ON switch.)

In FC form, the first word in each record (as read by Fortran) is a code word indicating the contents of the record. In NFC form, the first word is meaningless and the second word is the code word. In both cases, the next word is a "word count" (WC) word which contains the number of data words to follow (not including itself).

The first logical record contains the identification label. Its code word contains a 1 if a comment block does <u>not</u> follow, or a 4 if a comment block <u>does</u> follow. The WC word is followed by 31 words, each containing two ASCII characters. The actual ident string terminates with <carriage return> <line feed> characters, and the rest of the words are filled with zeros.

If a comment block is present, it is the second logical record, and its code word contains a 2. Again each word contains two ASCII characters.

The next record, having a code of 3, contains the scan parameters. See next page for details.

Each of the remaining logical records contains the data from one scan line. Its code word contains the negative of its line number within its frame. Thus, the code word for the first line in each frame contains a -1, the second line a -2, etc. Note that a data file may contain more than one frame.

PRECEDING PAGE BLANK NOT FILMED

The arrangement of data in the scan parameter record is as follows:

Word No.	Function
1	Points per line
2	X - direction (0 for left, -1 for right)
3	Y - direction (0 for front, -1 for back)
4	Delta - X
5	Y - step
6	X - distance*
7	X - distance**
8	No. of lines per frame
9	Scan pattern (0 for edge, -1 for raster, +1 for line)
10	Scanning speed
11	Backup? (0 for NO, -1 for YES)
12	No. of frames (2n-2, where n is the actual no. of frames)
13	Fortran - compatable? (0 for YES, -1 for NO)
14-45	X - start coordinates*
46-77	X - start coordinates**
78-109	Y - start coordinates*
110-141	Y - start coordinates**

* Low half of double precision number

** High half of double precision number

```
 CCCCCCCCCC    0000000000   NN        NN  TTTTTTTTTTTT  RRRRRRRRRRR   LL
CCCCCCCCCCCC  000000000000  NNN       NN  TTTTTTTTTTTT  RRRRRRRRRRRR  LL
CC        CC  00        00  NNNN      NN       TT       RR        RR  LL
CC            00        00  NN NN     NN       TT       RR        RR  LL
CC            00        00  NN  NN    NN       TT       RR        RR  LL
CC            00        00  NN   NN   NN       TT       RRRRRRRRRRRR  LL
CC            00        00  NN    NN  NN       TT       RRRRRRRRRRR   LL
CC            00        00  NN     NN NN       TT       RR    RR      LL
CC            00        00  NN      NNNN       TT       RR     RR     LL
CC        CC  00        00  NN       NNN       TT       RR      RR    LL
CCCCCCCCCCCC  000000000000  NN        NN       TT       RR       RR   LLLLLLLLLLLL
 CCCCCCCCCC    0000000000   NN        NN       TT       RR        RR  LLLLLLLLLLLL
```

```
    11        ØØØØØØØØØØ                ∧AAAAAAAAA   PPPPPPPPPPP   RRRRRRRRRRR
   111       ØØØØØØØØØØØØ               AAAAAAAAAAAA  PPPPPPPPPPPP  RRRRRRRRRRRR
  1111       ØØ       ØØØ               AA        AA  PP        PP  RR        RR
    11       ØØ      Ø ØØ               AA        AA  PP        PP  RR        RR
    11       ØØ     Ø  ØØ               AA        AA  PP        PP  RR        RR
    11       ØØ    Ø   ØØ    ********   AA        AA  PPPPPPPPPPPP  RRRRRRRRRRRR
    11       ØØ   Ø    ØØ    ********   AA        AA  PPPPPPPPPPP   RRRRRRRRRRR
    11       ØØ  Ø     ØØ               AAAAAAAAAAAA  PP            RR    RR
    11       ØØ Ø      ØØ               AAAAAAAAAAAA  PP            RR     RR
    11       ØØØ       ØØ               AA        AA  PP            RR      RR
 11111111    ØØØØØØØØØØØØ               AA        AA  PP            RR       RR
 11111111     ØØØØØØØØØØ                AA        AA  PP            RR        RR
```

TABLE OF CONTENTS

```
        .TITLE CONTRL
        :6 FEBRUARY 1974
        :LOCKHEED SOLAR OBSERVATORY, RYE CANYON, CALIFORNIA
        :OPERATING SYSTEM FOR MODEL 1010G(MODIFIED) DENSITOMETER
        :VERSION 3

        .GLOBL SETUP,IDENT,COMENT,MFLAG,DMOVE,SETDUN,IDCODE
        .GLOBL IDBUF,RDCHAR,LSTCHR,HOME,ERA,ABC,NWORDS,DATA
        .GLOBL NEGOK,CRLF,MESAGE,RDASC,RDASC2,S.EXEC,P.EXEC
        .GLOBL IDHEAD,CBLOCK,TOOLRG,MON,LDVAL,TSTSCN,DEVICE
        .GLOBL DELAY,GO.X,GO.Y,SWITCH,XGOL,YGOL,XNOWL,YNOWL
        .GLOBL IFLAG,RDVAL
        .MCALL .EXIT,.DTCVT,.BIN2D,.REGS,.RSTRT,.INIT,.RLSE
        .MCALL .TRAN,.WAIT
        .REGS

        LF=12
        CR=15
        SPACE=40
        RUBOUT=177
        TPS=177564
        TPB=177566
        ADS=167050
        ADC=167052
        MOTORS=167000
        DAC=167072
        DR11A=177522

        .MACRO  ASK TEXT
        MOV #.,ERA                      :ERROR RETURN ADDRESS
        JSR R5,MESAGE                   :TYPE OUT MESSAGE
        .BYTE CR,LF
        .ASCIZ &TEXT&
        .EVEN
        .ENDM

        .MACRO  MSG TEXT
        MOV #.,ERA
        JSR R5,MESAGE
        .ASCIZ *TEXT*
        .EVEN
        .ENDM

        .MACRO  ECHO CHAR
        TSTB TPS
        BPL .-4
        MOVB CHAR,TPB
        .ENDM
```

```
                .SBTTL  MONITOR

        MONITR: MOV SP,STACK
                MOV SWITCH,DR11A          ;INITIALIZE INCREMENT SWITCH TO 1
                ASK <PROGRAM TRACE>
                ASK <VERSION 3B>
                ASK <SET 'INCREMENT CONTROL' SWITCH TO AUTO>
                .RSTRT MON                ;SET RESTART ADDRESS

        MON:    JSR PC,CRLF
                ASK <MONITOR>

        STAR:   MOV STACK,SP              ;RESET STACK POINTER
                ASK <* >
                MOV #-1,MFLAG             ;CHAR REQUEST COMES FROM MONITOR
                JSR PC,RDCHAR             ;GET A COMMAND LETTER
                CLR MFLAG                 ;MONITOR REQUEST SATISFIED
                MOV #23.,RØ               ;ONLY 23 RECOGNIZED CHARS

        2$:     CMPB R1,SYMBOL(RØ)        ;IS CHAR IN TABLE?
                BEQ SHIFT                 ;IF SO, BRANCH
                SOB RØ,2$                 ;CHECK ANOTHER CHAR

        WHAT:   ASK <WHAT?>
                BR STAR

        SHIFT:  DEC RØ
                ASL RØ                    ;MAKE RØ A WORD OFFSET
                JSR PC,@COMAND(RØ)        ;GO TO PROPER ROUTINE
                BR STAR                   ;GET NEXT INSTRUCTION



        SYMBOL: .ASCII /ØABCDEFGHIKLMPQRSTUXYZ/
                .BYTE 3,15                ;CTRL/C,CR
                .EVEN
```

```
COMAND: ADCCHK
        EOT
        ORIGIN
        DIODE
        DO.E
        FREE
        GOTOXY
        HOME
        IDENT
        CLEAR
        LOADXY
        COMENT
        P.EXEC
        QLOAD
        DO.R
        S.EXEC
        TSTSCN
        SETUP
        GOTO.X
        GOTO.Y
        ZERO
        EXIT
        NOVA
```

```
            .SBTTL  EXECUTE MONITOR COMMANDS

  EXIT:     MOV #6000,MOTORS          :DISABLE AUTOLOK
            CLR DR11A
            .EXIT

  DO.E:     JSR PC,RDCHAR             :GET 2ND LETTER
            CMP R1,#'F                :ENDFILE?
            BNE 1$                    :IF NOT, BRANCH
            MOV #6,CODE               :IF SO, SET FUNCTION CODE
            BR COMPLY
  1$:       CMP R1,#'T                :END OF TAPE?
            BNE WHAT                  :IF NOT, ERROR
  EOT:      MOV #10,CODE              :IF SO, SET FUNCTION CODE
            BR COMPLY

  DO.R:     MOV #12,CODE              :REWIND FUNCTION CODE
  COMPLY:   JSR PC,CRLF
            JSR R5,@DEVICE            :EXECUTE FUNCTION
  CODE:     0
  NOVA:     RTS PC

  GOTO.X:   ASK <X DESTINATION:  >
            MOV #-1,NEGOK             :NEG COORD IS LEGAL
            JSR PC,RDASC2             :BIG NUMBER IS OK
            MOV R3,XGOL               :STORE NUMBER AS DESTINATION
            MOV R2,XGOL+2
            JSR PC,GO.X               :MOVE TO DESIRED X-COORDINATE
            BR BEEP

  GOTO.Y:   ASK <Y DESTINATION:  >
            MOV #-1,NEGOK             :SAME PROCEDURE FOR Y
            JSR PC,RDASC2
            MOV R3,YGOL
            MOV R2,YGOL+2
            JSR PC,GO.Y               :MOVE TO DESIRED Y-COORDINATE
  BEEP:     ECHO #7
            RTS PC

  ORIGIN:   ASK <CURRENT X:  >
            MOV #-1,NEGOK             :NEG COORD IS LEGAL
            JSR PC,RDASC2             :BIG NUMBER IS OK
            MOV #20,R1                :INDICATE X-DISPLAY REGISTER
            JSR PC,LDVAL              :LOAD NUMBER AS CURRENT COORD
            CLR R0                    :INDICATE X-UPPER LIMIT REGISTER
            JSR PC,LDVAL              :AUTOLOK MAY NEED IT
            MSG <CURRENT Y:  >
            MOV #-1,NEGOK
            JSR PC,RDASC2
            MOV #60,R1                :INDICATE Y-DISPLAY REGISTER
            JSR PC,LDVAL
            MOV #40,R1                :INDICATE Y-UPPER LIMIT REGISTER
            JSR PC,LDVAL              :LOAD IT FOR AUTOLOK
            RTS PC
```

```
ZERO:   CLR R2                          ;SPECIFY Ø,Ø AS THE
        CLR R3                          ;   CURRENT COORDS
        MOV #4,R4
        CLR R1                          ;SET INDICATOR FOR XUL (Ø),
1$:     JSR PC,LDVAL                    ;   XDISP (2Ø), YUL (4Ø),
        ADD #2Ø,R1                      ;   AND YDISP (6), AND
        SOB R4,1$                       ;   LOAD THE REGISTERS
        MOV #4,RØ
        MOV #XNOWL,R1
        MOV #YNOWL,R2
2$:     CLR (R1)+                       ;MAKE THE CURRENT AND DESTINATION
        CLR (R2)+                       ;   COORDINATES ZERO
        SOB RØ,2$
        MOV #6ØØØ,MOTORS                ;TURN AUTOLOK OFF
        RTS PC

LOADXY: JSR PC,RDNUM                    ;GET NUMBER OF COORD PAIR
        ASK <X: >
        JSR PC,1$                       ;READ AND STORE X-COORD
        MSG <Y: >                       ;DO SAME FOR Y-COORD
1$:     MOV R1,-(SP)                    ;SAVE BUF POINTER ON THE STACK
        MOV #-1,NEGOK                   ;NEGATIVE COORD IS LEGAL
        JSR PC,RDASC2                   ;SO IS BIG NUMBER
        MOV (SP)+,R1                    ;RESTORE BUF POINTER
        MOV R3,(R1)+                    ;STORE LOW HALF OF COORD
        MOV R2,(R1)+                    ;   AND HIGH HALF ALSO
        RTS PC

GOTOXY: JSR PC,RDNUM                    ;GET NUMBER OF COORD PAIR
        MOV (R1)+,XGOL                  ;R1 IS BUFFER POINTER
        MOV (R1)+,XGOL+2
        MOV (R1)+,YGOL
        MOV (R1),YGOL+2
        JSR PC,DMOVE                    ;GO TO DESIRED LOCATION
        BR BEEP                         ;RING THE BELL

QLOAD:  JSR PC,RDNUM                    ;GET NUMBER OF COORD PAIR
        MOV R1,RØ                       ;MOVE BUFFER POINTER TO RØ
        MOV #2Ø,R1                      ;INDICATE X-DISPLAY REGISTER
        JSR PC,1$                       ;READ AND STORE X-COORD
        MOV #6Ø,R1                      ;INDICATE Y-DISPLAY REGISTER
1$:     MOV RØ,-(SP)                    ;SAVE BUFFER POINTER ON THE STACK
        JSR PC,RDVAL                    ;READ THE INDICATED DISPLAY REG
        MOV (SP)+,RØ                    ;RESTORE THE BUFFER POINTER
        MOV R3,(RØ)+                    ;STORE LOW HALF OF COORD
        MOV R2,(RØ)+                    ;   AND HIGH HALF ALSO
        RTS PC

RDNUM:  JSR PC,RDCHAR                   ;INPUT A CHAR
        SUB #61,R1                      ;CONVERT TO BINARY (N-1)
        BGE 2$                          ;(N-1) MUST NOT BE NEGATIVE
1$:     JMP WHAT                        ;ERROR MESSAGE
2$:     CMP R1,#7                       ;(N-1) MUST NOT EXCEED 7
        BGT 1$                          ;BR IF ERROR
        ASH #3,R1                       ;CONVERT TO 4-WORD OFFSET
        ADD #GOBUF,R1                   ;POINT TO BUFFER
        RTS PC
        .SBTTL  COMPUTE FREE DISK SPACE

FREE:   .INIT LINK
        .TRAN LINK,MFD          :READ MASTER FILE DIRECTORY BLOCK
        .WAIT LINK

        CLR COUNTR              :COUNTR WILL CONTAIN FREE BLOCK COUNT
        MOV #5,RØ               :THERE ARE FIVE BIT MAPS
        MOV #MAPBLK,R1          :GET DISK ADDR OF FIRST BIT MAP

1$:     MOV (R1)+,RDMAP         :SET DISK ADDR IN TRAN BLOCK
        JSR R5,@44              :SAVE REGS ON STACK
        .TRAN LINK,RDMAP        :READ BITMAP INTO DATA BUFFER
        .WAIT LINK
        JSR R5,@46              :RESTORE REGS
        MOV #DATA+8.,R2         :IGNORE FOUR CONTROL WORDS
        MOV #6Ø.,R3             :BIT MAP CONTAINS 6Ø WORDS

2$:     MOV (R2)+,R4            :PICK UP BIT MAP WORD
        MOV #16.,R5             :IT CONTAINS 16 BITS

3$:     ASR R4                  :SHIFT BOTTOM BIT INTO CARRY BIT
        BCS 4$                  :IF CARRY IS SET, BLOCK IS NOT FREE
        INC COUNTR              :COUNT FREE BLOCK
4$:     SOB R5,3$               :CHECK ANOTHER BIT

        SOB R3,2$               :DO ANOTHER WORD

        SOB RØ,1$               :DO ANOTHER MAP
        .RLSE LINK              :RELEASE DATASET WHEN DONE

        .BIN2D 7$,COUNTR        :CONVERT NUMBER TO DECIMAL ASCII
        MOV #7$,RØ              :POINT TO START OF ASCII STRING
        MOV #4,R1               :WE WILL CHECK FIRST FOUR DIGITS
5$:     CMPB (RØ),#6Ø           :IS IT A ZERO?
        BNE 6$                  :IF NOT, BRANCH
        MOVB #4Ø,(RØ)+          :CHANGE ZERO TO SPACE
        SOB R1,5$               :   AND CHECK THE NEXT DIGIT

6$:     JSR R5,MESAGE
        .BYTE CR,LF
7$:     .BLKB 5
        .ASCIZ / FREE BLOCKS/
        RTS PC


ERRADR: ASK <.INIT FAILED>      :ERROR MESSAGE
        JMP MON
```

```
                .SBTTL  INPUT IDENT STRING

 IDENT:         JSR PC,CRLF
                ASK <IDENT:  >
                MOV #IDBUF,R2               ;STORAGE POINTER

                MOV #IDENT,RSTART           ;RESTART ADDR
                CLR COUNTR                  ;INITIALIZE CHAR COUNTER

 1$:            JSR PC,RDCHAR               ;GET CHAR
                CMP R1,#CR                  ;CARRIAGE RETURN?
                BEQ 2$                      ;IF SO, BRANCH
                JSR PC,STORE                ;IF NOT, STORE THE CHAR

                CMP COUNTR,#41.             ;TOO MANY CHARS?
                BLT 1$                      ;IF NOT, GET ANOTHER
                ASK <TOO LONG>
                BR IDENT

 2$:            MOV R2,-(SP)                ;SAVE R2 ON THE STACK
                .DTCVT DATE                 ;ENCODE DATE AND STORE
                MOV (SP)+,R2                ;RESTORE R2 FROM STACK
                MOV #19.,RØ                 ;19 CHARS ARE ADDED TO IDENT STRING
                ADD RØ,COUNTR               ;THEY MUST BE COUNTED
                MOV #TRACE,R1               ;POINTER TO "TRACED DATE"
 3$:            MOVB (R1)+,(R2)+            ;APPEND DATE TO IDENT STRING
                SOB RØ,3$

 4$:            JSR PC,ICRLF                ;END WITH CR,LF

                MOV #62.,RØ                 ;SIZE OF IDENT BUFFER
                SUB COUNTR,RØ               ;HOW MANY UNFILLED BYTES?
 5$:            CLRB (R2)+                  ;NULL THEM OUT
                SOB RØ,5$
                INC COUNTR                  ;COUNT MUST BE ROUNDED UP
                ASR COUNTR                  ;    AND CHANGED TO WORD COUNT,
                MOV COUNTR,ICOUNT           ;    THEN STORED
                CLR IFLAG                   ;INDICATE NEW IDENT STRING
                RTS PC
```

```
        .SBTTL  INPUT COMMENT BLOCK

COMENT: JSR PC,CRLF
        ASK <COMMENT:>
        JSR PC,CRLF
        MOV #DATA,R2             ;STORAGE POINTER
        MOV #COMENT,RSTART       ;RESTART ADDR
        CLR COUNTR               ;INITIALIZE CHAR COUNTER

1$:     JSR PC,RDCHAR            ;GET CHAR
        CMP R1,#LF               ;LINE FEED?
        BEQ 2$                   ;IF SO, DONE
        JSR PC,STORE             ;OTHERWISE, STORE CHAR
        BR 1$                    ;   AND GET ANOTHER

2$:     CMPB -1(R2),#LF          ;WAS PREVIOUS CHAR A CRLF?
        BEQ 3$                   ;IF SO, BRANCH
        JSR PC,ICRLF             ;IF NOT, INSERT ONE
3$:     CLRB (R2)+               ;NULL THE NEXT BYTE
        INC COUNTR               ;   AND COUNT IT
        MOV COUNTR,ABC           ;ACTUAL BYTE COUNT
        ADD #4,ABC               ;INCLUDE CODE WORD & WORD COUNT
        ASR COUNTR               ;CHANGE BYTE CNT TO WORD CNT
        MOV COUNTR,NWORDS        ;STORE AS WORD BEFORE BUFFER
        COM CBLOCK               ;COMMENT INDICATOR
        RTS PC

STORE:  CMP R1,#CR               ;CARRIAGE RETURN?
        BEQ ICRLF                ;IF SO, MAKE IT A CRLF
        CMP R1,#RUBOUT           ;RUBOUT?
        BNE 1$
        DEC COUNTR               ;UNCOUNT THE BAD CHAR
        BMI 2$                   ;IF COUNTER IS NEG, RESTART
        ECHO #'/                 ;PRINT THE DELETED CHAR,
        ECHO -(R2)               ;   SURROUNDED BY A PAIR
        ECHO #'/                 ;   OF SLASHES
        RTS PC
1$:     CMP R1,#25               ;CNTRL/U TO RESTART
        BNE 3$
2$:     TST (SP)+                ;POP THE RTS ADDR
        JMP @RSTART              ;GO TO THE RESTART ADDR
3$:     MOVB R1,(R2)+            ;STORE THE CHAR
        INC COUNTR               ;   AND COUNT IT
        RTS PC

ICRLF:  MOVB #CR,(R2)+
        MOVB #LF,(R2)+
        ADD #2,COUNTR            ;UPDATE BYTE COUNT
        RTS PC
```

```
CONTRL  MACRO VRØ5A 1Ø-APR-74 Ø1:29 PAGE 1Ø
PARAMETER STORAGE

                .SBTTL  PARAMETER STORAGE

        IDHEAD: 68.,1,68.,3
        IDCODE: 1
        ICOUNT: 1Ø.
        IDBUF:  .ASCIZ /UNIDENTIFIED SCAN/<CR><LF>
                .BLKB 42.

        TRACE:  .ASCII / = TRACED /
        DATE:   .BLKB 1Ø.
        GOBUF:  .BLKW 32.
        MFDBUF: .BLKW 3
        MAPBLK: .BLKW 5

        COUNTR: Ø
        MFLAG:  Ø
        IFLAG:  Ø
        CBLOCK: Ø
        RSTART: Ø
        STACK:  Ø


                ERRADR                        ;ERROR TRANSFER ADDR
        LINK:   Ø,Ø,1                         ;LINK BLOCK
                .RAD5Ø /DK/

        MFD:    1,MFDBUF,8.                   ;TRAN 8 WDS FROM BLOCK 1 INTO MFDBUF
                5,Ø

        RDMAP:  Ø,DATA,64.                    ;TRAN 64 WORDS INTO DATA BUFFER
                5,Ø

                .END    MONITR

ERRORS DETECTED:  Ø
FREE CORE:  12146. WORDS
*CONTRL.L2/NL:TTM:SYM:BIN:LOC<CONTRL
```

```
 SSSSSSSSSS   EEEEEEEEEEEE  TTTTTTTTTTTT  UU        UU  PPPPPPPPPPP
SSSSSSSSSSSS  EEEEEEEEEEEE  TTTTTTTTTTTT  UU        UU  PPPPPPPPPPPP
SS        SS  EE                 TT       UU        UU  PP        PP
SS            EE                 TT       UU        UU  PP        PP
SS            EE                 TT       UU        UU  PP        PP
SSSSSSSSSSS   EEEEEEEE           TT       UU        UU  PPPPPPPPPPPP
 SSSSSSSSSSS  EEEEEEEE           TT       UU        UU  PPPPPPPPPPP
          SS  EE                 TT       UU        UU  PP
          SS  EE                 TT       UU        UU  PP
SS        SS  EE                 TT       UU        UU  PP
SSSSSSSSSSSS  EEEEEEEEEEEE       TT       UUUUUUUUUUUU  PP
 SSSSSSSSSS   EEEEEEEEEEEE       TT        UUUUUUUUUU   PP
```

```
    11      ØØØØØØØØØØ                 AAAAAAAAAA   PPPPPPPPPPP   RRRRRRRRRRR
   111     ØØØØØØØØØØØØ               AAAAAAAAAAAA  PPPPPPPPPPPP  RRRRRRRRRRRR
  1111     ØØ       ØØØ               AA        AA  PP        PP  RR        RR
    11     ØØ      Ø ØØ               AA        AA  PP        PP  RR        RR
    11     ØØ     Ø  ØØ               AA        AA  PP        PP  RR        RR
    11     ØØ    Ø   ØØ    ********   AA        AA  PPPPPPPPPPPP  RRRRRRRRRRRR
    11     ØØ   Ø    ØØ    ********   AA        AA  PPPPPPPPPPP   RRRRRRRRRRR
    11     ØØ  Ø     ØØ               AAAAAAAAAAAA  PP            RR    RR
    11     ØØ Ø      ØØ               AAAAAAAAAAAA  PP            RR     RR
    11     ØØØ       ØØ               AA        AA  PP            RR      RR
 11111111  ØØØØØØØØØØØØ               AA        AA  PP            RR       RR
 11111111   ØØØØØØØØØØ                AA        AA  PP            RR        RR
```

TABLE OF CONTENTS

```
        .TITLE SETUP
        ;16 JANUARY 1974

        .GLOBL SETUP,DELTAX,DELTAY,XMOVE,EXEC,XDIR,YDIR,MON
        .GLOBL ERA,XTRAVL,XTRAVH,NSCANS,PATERN,SPEED,BACKUP
        .GLOBL SETDUN,NLINES,XLBUF,XHBUF,YLBUF,YHBUF,DEVICE
        .GLOBL RTNSPD,RDCHAR,DISK,TAPE,LSTCHR,XRAMPS,TOOLRG
        .GLOBL NPOINT,NEGOK,CRLF,MESAGE,RDASC,RDASC2,IOMODE
        .GLOBL DX,SWCSET,SWCSPD,VELSEL,SWITCH,SERIES,NUMBER
        .GLOBL RAMP,SPDSET,TOCS,TOCKS
        .MCALL .BIN2D,.REGS
        .REGS

        LF=12
        CR=15
        PLUS=53
        MINUS=55
        DR11A=177522

        .MACRO  ASK TEXT
        MOV #.,ERA                  ;ERROR RETURN ADDRESS
        JSR R5,MESAGE               ;TYPE OUT MESSAGE
        .BYTE CR,LF
        .ASCIZ &TEXT&
        .EVEN
        .ENDM

        .MACRO  MSG TEXT
        MOV #.,ERA
        JSR R5,MESAGE
        .ASCIZ *TEXT*
        .EVEN
        .ENDM
        .SBTTL  SETUP EXECUTIVE

SETUP:  TST SETDUN                      ;HAS FULL SETUP BEEN DONE?
        BPL EXEC                        ;IF NOT, DO IT
        ASK <OPTION:  >
        JSR PC,RDCHAR                   ;GET CONTROL CHAR
        MOV #15.,RØ                     ;ONLY 15 LEGAL CHARS
1$:     CMPB R1,TABLE(RØ)               ;IS CHAR IN TABLE?
        BEQ 2$                          ;IF SO, BRANCH
        SOB RØ,1$                       ;TRY NEXT ENTRY IN TABLE
        JMP ERROR

2$:     CMP RØ,#13.                     ;IS 2ND LETTER REQUIRED?
        BGE 3$                          ;IF SO, BRANCH
        JSR PC,CRLF                     ;OTHERWISE, GO TO NEXT LINE
3$:     DEC RØ
        ASL RØ                          ;MAKE RØ A WORD OFFSET
        JSR PC,@OPTION(RØ)              ;GO TO PROPER ROUTINE
        RTS PC                          ;RETURN TO MONITOR

TABLE:  .ASCII /Ø#BEFLMPSUVXYCDN/
        .EVEN

OPTION: NUMSET
        GETBAK
        MON
        FORTIO
        LETTER
        MEDIUM
        GETPAT
        SETSWC
        EXEC
        GETVEL
        GETX
        GETY
        C2
        D2
        N2

SETSWC: MOV #2,SWCSET                   ;INDICATE 1 MICRON
        MOVB SWCNUM+2,SWITCH+1          ;SET SWITCH BITS
        MOV SWITCH,DR11A                ;SET SWITCH
        MOV DX,DELTAX                   ;MAKE DELTAX CORRECT FOR 1-MICRON SWITCH
        MOV XTRAVL,R3
        MOV XTRAVH,R2                   ;RECOVER SCAN LINE LENGTH

SPDSET: MOV #XMOVE,RØ                   ;INDICATE X-MOVE
        JSR PC,VELSEL                   ;GET PROPER SPEED FOR THIS SPEED SETTING
        MOV (R5),RAMP                   ;STORE RAMP FOR THIS SPEED
        MOVB SWCSPD(R4),SPEED           ;STORE SPEED
        MOV SPEED,RTNSPD
        RTS PC
```

```
EXEC:    CLR RØ                        ;RØ IS LIST POINTER
1$:      MOV RØ,-(SP)                  ;SAVE RØ ON THE STACK
         JSR PC,@SEQ(RØ)               ;GO TO PROPER ROUTINE
         MOV (SP)+,RØ                  ;RESTORE RØ FROM STACK
         TST (RØ)+                     ;POINT TO NEXT ROUTINE
         BR 1$                         ;   AND EXECUTE IT

EDUN:    MOV #-1,SETDUN                ;INDICATE COMPLETED SETUP
         JMP MON                       ;EXIT DIRECTLY TO MONITOR

SEQ:     GETX
         GETY
         GETPAT
         GETDX
         GETDY
         GETNL
         GETXY
         EDUN

N2:      JSR PC,RDCHAR                 ;GET 2ND COMMAND LETTER
         JSR PC,CRLF                   ;ADVANCE TO NEXT LINE
         CMP R1,#'P
         BNE 1$
         JMP GETNP
1$:      CMP R1,#'L
         BNE JE
         JMP GETNL
JE:      JMP ERROR

D2:      JSR PC,RDCHAR                 ;GET 2ND COMMAND LETTER
         JSR PC,CRLF                   ;ADVANCE TO NEXT LINE
         CMP R1,#'X
         BEQ GETDX
         CMP R1,#'Y
         BNE JE
         JMP GETDY

LETTER:  ASK <SERIES LETTER: >         ;THIS ROUTINE PROVIDES A NEW
         JSR PC,RDCHAR                 ;   FIRST LETTER FOR THE
         JSR PC,CRLF                   ;   OUTPUT FILENAME
         CMPB R1,#'A                   ;AT LEAST ASCII 'A'?
         BLT JE                        ;IF NOT, ERROR
         CMPB R1,#'Z                   ;ABOVE ASCII 'Z'?
         BGT JE                        ;IF SO, ERROR
         MOV R1,SERIES                 ;STORE SERIES DESIGNATOR
         MOV #1,NUMBER                 ;INITIALIZE COUNT
         RTS PC

NUMSET:  ASK <NUMBER: >
         JSR PC,RDASC                  ;GET SMALL POSITIVE NUMBER
         CMP R3,#99.                   ;MUSTN'T BE OVER 99
         BLE 1$
         JMP TOOLRG
1$:      MOV R3,NUMBER
         RTS PC
```

```
                .SBTTL  INPUT SCANNING PARAMETERS

GETX:           JSR PC,CRLF
                ASK <X-DIR    >
                JSR PC,RDCHAR
                CLR XDIR
                CMP R1,#'L
                BEQ 1$                         ;XDIR=Ø MEANS "LEFT"
                CMP R1,#PLUS
                BEQ 1$
                COM XDIR                       ;XDIR=-1 MEANS "RIGHT"
                CMP R1,#'R
                BEQ 1$
                CMP R1,#MINUS
                BNE JE
1$:             RTS PC

GETY:           ASK <Y-DIR    >
                JSR PC,RDCHAR
                CLR YDIR
                CMP R1,#'B
                BEQ 1$                         ;YDIR=Ø MEANS "BACK"
                CMP R1,#PLUS
                BEQ 1$
                COM YDIR                       ;YDIR=-1 MEANS "FRONT"
                CMP R1,#'F
                BEQ 1$
                CMP R1,#MINUS
                BNE JE
1$:             RTS PC

GETDX:          ASK <DELTA X  >
                JSR PC,RDASC
                MOV R3,DX
                BLE BADNUM                     ;MUST BE >Ø
                CMP #2ØØØ.,R3
                BMI BADNUM                     ;MUST NOT EXCEED 2 MM

                MOV #2,R4                      ;INITIALIZE POINTER
1$:             MOV R3,DELTAX                  ;ASSUME R3 CORRECT DELTA-X
                ASR R3                         ;SHIFT LOW BIT INTO THE C-BIT
                BCS 2$                         ;BRANCH IF IT'S SET
                SOB R4,1$                      ;OTHERWISE, CHECK THE NEXT BIT
                MOV R3,DELTAX                  ;IT HAS NOW BEEN DIVIDED BY 4
2$:             MOV R4,SWCSET                  ;STORE INDEX OF SWITCH SETTING
                MOVB SWCNUM(R4),SWITCH+1;STORE SCALE INCREMENT SWITCH BITS
                MOV SWITCH,DR11A               ;ACTUATE THE SWITCH
```

```
GETNP:  MSG <PTS/LINE >
        JSR PC,RDASC
        CMP R3,BUFSIZ               ;CANNOT EXCEED BUFFER SIZE
        BLOS 1$
        JMP TOOLRG
1$:     CMP R3,#1                   ;MUST BE >1
        BLE BADNUM
        MOV R3,NPOINT               ;STORE NUMBER OF POINTS
        DEC R3
        MOV R3,R2                   ;MUST BE IN EVEN REGISTER
        MUL DX,R2                   ;   FOR DOUBLE WORD RESULT
        JSR PC,SPDSET               ;SELECT OPTIMUM SPEED
        MOV TOCS,TOCKS              ;STORE THE DELAY TIMER
        MOV R3,XTRAVL               ;STORE LO ORDER WORD
        MOV R2,XTRAVH               ;STORE HI ORDER WORD
        RTS PC

GETDY:  TST PATERN
        BGT 1$
        MSG <Y STEP    >
        JSR PC,RDASC
        MOV R3,DELTAY
1$:     RTS PC

BADNUM: MOV ERA,ERRET
        ASK <ILLEGAL NUMBER>
        JSR PC,CRLF
        JMP @ERRET                  ;GO TO ERROR RETURN ADDR

GETNL:  TST PATERN                  ;SINGLE LINE SCAN?
        BLE 1$                      ;IF NOT, BRANCH
        MOV #1,NLINES
        RTS PC
1$:     MSG <LINES     >
        JSR PC,RDASC                ;SMALL NUMBER ONLY
        MOV R3,NLINES
        RTS PC

GETPAT: ASK <PATTERN  >
        JSR PC,RDCHAR
        MOV #4,R2                   ;POINTER TO CHAR LIST
1$:     CMPB 3$(R2),R1              ;LOOK FOR A MATCH
        BEQ 2$                      ;IF FOUND, BRANCH
        SOB R2,1$                   ;OTHERWISE, DECREMENT POINTER
        BR ERROR                    ;ERROR IF NO MATCH
2$:     SUB #3,R2                   ;MAKE RANGE -2 TO +1
        MOV R2,PATERN               ;STORE RESULT
        RTS PC
3$:     .ASCIZ /ØBREL/
```

```
GETVEL: MSG <SPEED    >
        JSR PC,RDASC
        MOV SWCSET,RØ
        MOVB SWCSPD(RØ),R2          :GET MAX SPEED FOR THIS SWITCH SETTING
        BIC #1774ØØ,R2              :ELIMINATE EXTENDED SIGN
        TST R3                      :COMPARE R3 TO ZERO
        BLE BADNUM                  :SPEED MUST BE >Ø
        CMP R3,R2                   :MUST NOT BE GREATER THAN MAX SPEED
        BGT BADNUM                  :BRANCH IF TOO FAST
        MOV R3,SPEED                :STORE NEW SPEED
        MOV #5,RØ                   :POINTER FOR SPEEDS LIST
1$:     CMPB R3,SWCSPD(RØ)          :COMPARE NEW SPEED TO LIST ITEM
        BLE 2$                      :IF NOT FASTER, BRANCH
        SOB RØ,1$                   :OTHERWISE, COMPARE TO HIGHER SPEED
2$:     ASL RØ                      :MAKE RØ A WORD OFFSET
        MOV XRAMPS(RØ),RAMP         :GET APPROPRIATE RAMP FOR NEW SPEED
        RTS PC

FORTIO: CLR IOMODE                  :Ø MEANS FORTRAN-COMPATIBLE I/O
        ASK <FTN I/O? >
        JSR PC,RDCHAR
        CMP R1,#'Y                  :YES?
        BEQ 1$
        COM IOMODE                  :-1 MEANS NON-FORTRAN
        CMP R1,#'N                  :NO?
        BNE ERROR
1$:     RTS PC

GETBAK: MSG <BACKUP?  >
        JSR PC,RDCHAR
        CLR BACKUP                  :ZERO MEANS NO
        CMP R1,#'N
        BEQ 1$
        COM BACKUP                  :-1 MEANS YES
        CMP R1,#'Y
        BNE ERROR
1$:     RTS PC

MEDIUM: ASK <STORAGE  >
        JSR PC,RDCHAR
        MOV #DISK,DEVICE            :ASSUME DISK STORAGE
        CMP R1,#'D                  :"D" FOR DISK?
        BEQ 1$                      :IF SO, EXIT
        MOV #TAPE,DEVICE            :CHANGE DEVICE POINTER
        CMP R1,#'T                  :"T" FOR TAPE?
        BEQ 1$
        CMP R1,#'M                  :"M" FOR MAGTAPE IS ACCEPTABLE
        BNE ERROR
1$:     RTS PC

ERROR:  MOV ERA,ERRET               :SAVE ERA BECAUSE.....
        ASK <WHAT?>                 :   "ASK" RESETS IT
        JMP @ERRET                  :GO TO ERROR RETURN ADDR
```

```
SETUP   MACRO VRØ5A 1Ø-APR-74 Ø1:38 PAGE 7
INPUT SCANNING PARAMETERS

C2:     JSR PC,RDCHAR               ;GET 2ND LETTER OF COMMAND
        CMPB R1,#CR                 ;IS IT A CARRIAGE RETURN?
        BEQ GETXY                   ;IF SO, INPUT WHOLE ARRAY

        CMPB R1,#'N                 ;CHANGE A PARTICULAR FRAME?
        BNE 2$                      ;IF NOT, BRANCH
        ASK <FRAME >
        JSR PC,RDASC                ;GET THE FRAME NUMBER
        DEC R3
        BMI ERROR                   ;NUMBER HAD TO BE POSITIVE
        ASL R3
        CMP R3,NSCANS               ;NSCANS CONTAINS (2N-2)
        BLE 1$                      ;BRANCH IF NUMBER IS OK
        JMP TOOLRG                  ;GO TO ERROR ROUTINE IF NEEDED
1$:     MOV R3,COUNTR               ;'COUNTR' CONTAINS FRAME POINTER
        BR XYINPT                   ;XYINPT WILL PROVIDE RTS

2$:     CMPB R1,#'X                 ;SET ALL X COORDS?
        BNE 3$                      ;IF NOT, BRANCH
        ASK <X = >
        MOV #XLBUF,COUNTR           ;COUNTR WILL PROVIDE LIST POINTER
        BR 4$

3$:     CMPB R1,#'Y                 ;CHANGE ALL Y COORDS?
        BNE ERROR                   ;THAT WAS LAST LEGAL OPTION
        ASK <Y = >
        MOV #YLBUF,COUNTR           ;POINTER TO Y-STORAGE ARRAY

4$:     MOV #-1,NEGOK               ;A NEG COORD IS LEGAL
        JSR PC,RDASC2               ;NUMBER MAY T/BE LARGE
        MOV COUNTR,RØ               ;POINTER TO LOW ORDER LIST
        MOV RØ,R1
        ADD #64.,R1                 ;POINTER TO HIGH ORDER LIST
        MOV #32.,R4                 ;CHANGE ALL VALUES FOR THE COORD
5$:     MOV R3,(RØ)+                ;STORE LOW ORDER WORD
        MOV R2,(R1)+                ;STORE HIGH ORDER WORD
        SOB R4,5$                   ;GO THRU ENTIRE ARRAY
        RTS PC
```

```
GETXY:  CLR COUNTR                  ;POINTER TO FRAME NUMBER
        MOV #1,FCOUNT               ;FRAME NUMBER
1$:     .BIN2D FBUF+1,FCOUNT        ;ENCODE FRAME NUMBER
        CMPB FBUF+4,#60             ;IS THERE A LEADING ZERO?
        BNE 2$                      ;IF NOT, BRANCH
        MOVB #40,FBUF+4             ;IF SO, CHANGE IT TO A SPACE
2$:     MOV FBUF+4,3$               ;PUT NUMBER IN TEXT STRING
        JSR R5,MESAGE
        .ASCII <CR><LF>/FRAME /
3$:     0,0

        JSR PC,XYINPT               ;GET A COORD PAIR
        TST LSTCHR                  ;WAS <LF> TYPED?
        BEQ 5$                      ;IF SO, INPUT IS DONE
        CMP R0,#58.                 ;HAS LIMIT BEEN REACHED?
        BGE 4$                      ;IF SO, PRINT MESSAGE
        ADD #2,COUNTR               ;POINT TO NEXT STORAGE LOCATIONS
        INC FCOUNT                  ;INCREMENT FRAME NUMBER
        BR 1$                       ;GET NEXT COORD SET

4$:     ASK <LIMIT REACHED>
5$:     MOV R0,NSCANS               ;STORE SCAN COUNT (2N - 2)
        RTS PC


XYINPT: ASK <X = >
        MOV #-1,NEGOK               ;NEG COORD IS LEGAL
        JSR PC,RDASC2               ;SO IS LARGE NUMBER
        TST LSTCHR                  ;WAS <LF> TYPED?
        BEQ 1$                      ;IF SO, IGNORE X AND EXIT
        MOV COUNTR,R0
        MOV R3,XLBUF(R0)            ;STORE LOW ORDER WORD
        MOV R2,XHBUF(R0)            ;STORE HIGH ORDER WORD

        MSG <Y = >                  ;DO SAME FOR Y
        MOV #-1,NEGOK
        JSR PC,RDASC2
        MOV COUNTR,R0
        MOV R3,YLBUF(R0)
        MOV R2,YHBUF(R0)
1$:     RTS PC
```

```
        .SBTTL  PARAMETER STORAGE

;************************************************************************
NPOINT: Ø
XDIR:   Ø
YDIR:   Ø
DX:     Ø
DELTAY: 1ØØ.
XTRAVL: Ø
XTRAVH: Ø                               ;THESE ADDR'S MUST REMAIN
NLINES: Ø                               ;INTACT AND IN THIS ORDER
PATERN: Ø
SPEED:  1ØØ
BACKUP: Ø
NSCANS: Ø
IOMODE: Ø
XLBUF:  .BLKW 32.
XHBUF:  .BLKW 32.
YLBUF:  .BLKW 32.
YHBUF:  .BLKW 32.
;************************************************************************

DELTAX: Ø
DEVICE: DISK
SETDUN: Ø
ERA:    Ø
ERRET:  Ø
COUNTR: Ø
FBUF:   Ø,Ø,Ø
FCOUNT: Ø
SWCSET: 2
SWCNUM: .BYTE 3Ø,5Ø,1ØØ,Ø
BUFSIZ: 12ØØØ.
        .END

ERRORS DETECTED:  Ø
FREE CORE:  12518. WORDS
,SETUP,L2/NL:TTM:SYM:BIN:LOC<SETUP
```

```
 SSSSSSSSSS    CCCCCCCCCC    AAAAAAAAAA   NN        NN
SSSSSSSSSSSS  CCCCCCCCCCCC  AAAAAAAAAAAA  NNN       NN
SS        SS  CC        CC  AA        AA  NNNN      NN
SS            CC            AA        AA  NN NN     NN
SS            CC            AA        AA  NN  NN    NN
SSSSSSSSSSS   CC            AA        AA  NN   NN   NN
 SSSSSSSSSSS  CC            AA        AA  NN    NN  NN
          SS  CC            AAAAAAAAAAAA  NN     NN NN
          SS  CC            AAAAAAAAAAAA  NN      NNNN
SS        SS  CC        CC  AA        AA  NN       NNN
SSSSSSSSSSSS  CCCCCCCCCCCC  AA        AA  NN        NN
 SSSSSSSSSS    CCCCCCCCCC   AA        AA  NN        NN
```

```
   11        ØØØØØØØØØØ                  AAAAAAAAAA   PPPPPPPPPPP   RRRRRRRRRRR
  111       ØØØØØØØØØØØØ                 AAAAAAAAAAAA  PPPPPPPPPPPP  RRRRRRRRRRRR
 1111       ØØ       ØØØ                 AA        AA  PP        PP  RR        RR
   11       ØØ      Ø ØØ                 AA        AA  PP        PP  RR        RR
   11       ØØ     Ø  ØØ                 AA        AA  PP        PP  RR        RR
   11       ØØ    Ø   ØØ    ********     AA        AA  PPPPPPPPPPPP  RRRRRRRRRRRR
   11       ØØ   Ø    ØØ    ********     AA        AA  PPPPPPPPPPP   RRRRRRRRRRR
   11       ØØ  Ø     ØØ                 AAAAAAAAAAAA  PP            RR    RR
   11       ØØ Ø      ØØ                 AAAAAAAAAAAA  PP            RR     RR
   11       ØØØ       ØØ                 AA        AA  PP            RR      RR
11111111    ØØØØØØØØØØØØ                 AA        AA  PP            RR       RR
11111111     ØØØØØØØØØØ                  AA        AA  PP            RR        RR
```

PRECEDING PAGE BLANK NOT FILMED

```
SCAN     MACRO VR05A 10-APR-74 01:39
TABLE OF CONTENTS

    2-   1          SCAN EXECUTIVE
    3-   1          PLAYBACK EXECUTIVE
    4-  48          ABORT SCAN
    5-   1          INITIALIZE SCAN
    6-   1          SCAN ONE LINE
    7-   1          MOVE TO NEXT SCAN LINE
    7-  24          COMPLETION OF FRAME SCAN
    7-  45          MOVE CARRIAGE TO THE ORIGIN
    8-   1          MOVE ROUTINES
   10-   1          SETUP FOR MOVE
   11-   1          PARAMETER STORAGE
```

```
        .TITLE  SCAN
        ;1 FEBRUARY 1974

        .GLOBL XDIR,YDIR,DELTAX,XTRAVL,XTRAVH,DELTAY,HOME,DSP
        .GLOBL NSCANS,PATERN,SPEED,KEY,XLBUF,XHBUF,CRLF,RDASC
        .GLOBL YLBUF,YHBUF,SETDUN,NLINES,NPOINT,DEVICE,TSTSCN
        .GLOBL RDXY,LDVAL,NEGATE,RDCHAR,MESAGE,DELAY,DX,XMOVE
        .GLOBL XNOWL,YNOWL,S.EXEC,P.EXEC,EXEC,MON,RAMP,CBLOCK
        .GLOBL DMOVE,GO.X,GO.Y,XGOH,YGOH,XGOL,YGOL,BACKUP,CRT
        .GLOBL SWCSPD,SWITCH,XRAMPS,SPDSEL,TEMP,SPDSET,RTNSPD
        .GLOBL RDVAL,TOCS,TOCKS,DATA
        .MCALL .PARAM
        .PARAM

        MOTORS = 167000         ;DENSITOMETER MOTOR CONTROL
        DBNB   = 167032         ;DIVIDE-BY-N BUFFER
        DBNS   = 167034         ;DIVIDE-BY-N STARUS REGISTER
        LIMITS = 167040         ;LIMIT FLAGS REGISTER
        ADC    = 167052         ;A/D CONVERTER BUFFER
        DAC    = 167072         ;D/A CONVERTER BUFFER
        DR11A  = 177522         ;SCALE INCREMENT SWITCH BITS


        .MACRO  ASK TEXT
        JSR R5,MESAGE
        .ASCIZ <CR><LF>@TEXT@
        .EVEN
        .ENDM

        .MACRO  MSG TEXT
        JSR R5,MESAGE
        .ASCIZ @TEXT@
        .EVEN
        .ENDM

        .ASECT
        .=170
        DBNINT                  ;DIVIDE-BY-N INTERRUPT VECTOR
        340                     ;PRIORITY 7


        .CSECT
```

```
             .SBTTL  SCAN EXECUTIVE

   S.EXEC:   MOV #-1,TFLAG               ;-1 MEANS REAL RUN
   TSTSCN:   TST SETDUN                  ;-1 IF SETUP COMPLETED
             BMI 1$
             JMP EXEC                    ;GO TO SETUP ROUTINE

   1$:       MOV #1Ø122,PASS.A           ;"MOV R1,(R2)+" INSTRUCTION CODE
             MOV #1Ø142,PASS.B           ;"MOV R1,-(R2)" INSTRUCTION CODE
             CLR DSP                     ;CLEAR DISPLAY FLAG
             TST TFLAG                   ;IS THIS A REAL RUN?
             BPL 2$                      ;IF NOT, DON'T OPEN A FILE
             JSR R5,@DEVICE              ;WRITE INITIALIZING DATA ON
             2                           ;    PRIMARY STORAGE DEVICE
             MOV #14,SOURCE              ;14 MEANS 1ST DATA RECORD

   2$:       MOV XTRAVL,XTL              ;MOVE THE X-TRAVEL DISTANCE
             MOV XTRAVH,XTH              ;    TO A WORK AREA
             MOV PATERN,PAT              ;DITTO FOR PATTERN
             JSR PC,SCAN                 ;INITIALIZE SCAN

   3$:       JSR PC,XSCAN                ;START SCAN OF ONE LINE

             MOV MODE,5$                 ;INSTALL DATA TRANSFER INSTRUCTION
             MOV #141,DBNS               ;ENABLE DIVIDE-BY-N INTERRUPT
   4$:       MOV ADC,R1                  ;PICK UP THE DIGITIZED DENSITY
             ASH #-2,R1                  ;REDUCE IT TO THE REAL 1Ø BITS
   5$:       MOV R1,(R2)+                ;STORE IT
             WAIT                        ;WAIT FOR DIVIDE-BY-N INTERRUPT
             SOB R3,4$                   ;COUNT THE POINT, DO IT AGAIN
             MOV TEMP,PSW                ;RESTORE THE PROCESSOR STATUS

             JSR PC,SIGNAL               ;STOP MOTOR, CHECK ABORT SWITCH
             TST TFLAG                   ;IS THIS A REAL RUN?
             BPL 6$                      ;IF NOT, DON'T RECORD THE DATA
             MOV SOURCE,.+12             ;SPECIFIES WHETHER 1ST DATA RECORD
             JSR R5,@DEVICE              ;RECORD DATA ON STORAGE DEVICE
             Ø
             CLR SOURCE                  ;ZERO MEANS "NOT 1ST RECORD"

   6$:       BIT #1Ø,SWR                 ;IS SWITCH 3 SET?
             BEQ 7$                      ;IF NOT, BRANCH
             JSR PC,CRT                  ;IF SO, DISPLAY DATA ON TERMINAL
             BIT #2Ø,SWR                 ;HAS ABORT BEEN REQUESTED?
             BEQ 7$                      ;IF NOT, BRANCH
             JMP ZAP                     ;IF SO, JUMP TO ABORT ROUTINE

   7$:       DEC LCOUNT                  ;COUNT THE LINE
             BEQ 8$                      ;IF LAST LINE, BRANCH
             JSR PC,NEXTLN               ;GO TO NEXT LINE
             BR 3$                       ;    AND SCAN IT
   8$:       MOV #14,SOURCE              ;INDICATE 1ST RECORD OF NEW FRAME
             JSR PC,ENDSCN               ;RETURN TO ORIGIN, CHECK FRAME COUNT
             BR 3$                       ;SCAN NEXT FRAME
```

```
        .SBTTL  PLAYBACK EXECUTIVE

P.EXEC: MOV #-1,TFLAG
        MOV #-3,SOURCE
        ASK <SCALE FACTOR  >
        JSR PC,RDASC            ;INPUT A SMALL NUMBER
        TST R3                  ;IS IT <Ø?
        BGT 2$                  ;IF SO, BRANCH
1$:     MSG <WHAT?>             ;IF NOT, ERROR
        BR P.EXEC
2$:     CMP R3,#1ØØ.            ;1ØØ IS UPPER LIMIT
        BGT 1$                  ;ERROR IF EXCEEDED
        MOV R3,NPULSE           ;STORE THE PULSE COUNT

3$:     MSG <PARAMETER SOURCE? (K OR R)  >
        JSR PC,RDCHAR
        JSR PC,CRLF
        CMP R1,#'K              ;"K" FOR KEYBOARD?
        BEQ 4$
        MOV #-5,SOURCE
        CMP R1,#'R              ;"R" FOR RECORDED?
        BEQ 5$
        MSG <WHAT?>
        JSR PC,CRLF
        BR 3$

4$:     TST SETDUN              ;-1 IF SETUP COMPLETED
        BMI 5$
        JMP EXEC                ;GO TO SETUP ROUTINE

5$:     CLR DSP                 ;CLEAR DISPLAY FLAG
        MOV SOURCE,.+12         ;SPECIFY REQUESTED PARAMETER SOURCE
        JSR R5,@DEVICE          ;OPEN INPUT FILE
        -3
        MOV #-11,SOURCE         ;LOOK FOR 1ST DATA RECORD

        CLR R3
        CLR R2
        MOV NPULSE,RØ           ;COMPUTE THE ACTUAL LINE LENGTH
6$:     ADD XTRAVL,R3           ;   BY ADDING THE SINGLE-STEP
        ADC R2                  ;   LENGTH NPULSE TIMES
        ADD XTRAVH,R2
        SOB RØ,6$
        MOV NPULSE,RØ           ;A CORRECTION MUST BE INCLUDED
        DEC RØ                  ;   TO ACCOUNT FOR THE ENDPOINT
        MUL DX,RØ

        ADD R1,R3
        ADC R2
        ADD RØ,R2
        MOV R3,XTL              ;STORE THE RESULT IN A WORKING AREA
        MOV R2,XTH
        JSR PC,SPDSET           ;SELECT OPTIMUM SPEED FOR THIS LINE
        CLR R5                  ;INDICATE 1ST FRAME
        MOV PATERN,PAT          ;PUT SCAN PATTERN IN SCRATCH WORD
        JSR PC,KEY              ;HIT KEY TO CONTINUE
        JSR PC,SCAN             ;INITIALIZE SCAN
```

```
PHOTO:  MOV SOURCE,,+12          :INDICATES DATA RECORD NUMBER
        JSR R5,@DEVICE           :READ A LINE OF DATA
        -1
        MOV #-1,SOURCE           :NEXT RECORD WON'T BE THE 1ST

        BIT #1Ø,SWR              :IS SWITCH 3 SET?
        BEQ 1$                   :IF NOT, BRANCH
        JSR PC,CRT               :IF SO, DISPLAY DATA ON TERMINAL
        BIT #2Ø,SWR              :HAS ABORT BEEN REQUESTED?
        BNE ZAP                  :IF SO, GO TO ABORT ROUTINE
1$:     NEG PAT                  :SET UP REPEAT PATTERN
        MOV NPULSE,LREP          :NPULSE ALSO CAUSES LINE REPEAT
        MOV #Ø122Ø1,PASS.A       :"MOV (R2)+,R1" INSTRUCTION CODE
        MOV #Ø142Ø1,PASS.B       :"MOV -(R2),R1" INSTRUCTION CODE

2$:     JSR PC,XSCAN             :START SCAN OF ONE LINE

        MOV MODE,4$              :INSTALL DATA TRANSFER INSTRUCTION
        MOV #1Ø1,DBNS            :ENABLE DIVIDE-BY-N INTERRUPT
3$:     MOV NPULSE,RØ            :NO. OF PULSES FOR EACH POINT
4$:     MOV (R2)+,R1             :DIGITAL INPUT VALUE
        CMP R1,#255.             :8-BIT LIMIT
        BLOS 5$                  :CONTINUE IF IN RANGE
        MOV #255.,R1             :OTHERWISE, SET TO MAX
5$:     MOV R1,DAC               :SEND SIGNAL TO DIODE
        WAIT                     :WAIT FOR INTERRUPT
        SOB RØ,5$                :COUNT THE PULSE NUMBER
        SOB R3,3$                :COUNT THE POINT, DO IT AGAIN
        MOV TEMP,PSW             :RESTORE THE PROCESSOR STATUS

        JSR PC,SIGNAL            :STOP MOTOR, CHECK ABORT SWITCH
        DEC LREP                 :LINE REPEAT COUNTER
        BEQ 6$                   :Ø MEANS DONE WITH THE LINE
        JSR PC,NEXTLN            :DO EDGE MOVE TO NEXT LINE
        BR 2$                    :DO THE LINE REPEAT

6$:     NEG PAT                  :RESTORE THE SPECIFIED PATTERN
        DEC LCOUNT               :COUNT THE LINE
        BEQ 7$                   :IF LAST LINE, BRANCH
        JSR PC,NEXTLN            :GO TO NEXT LINE
        BR PHOTO                 :   AND CONTINUE

7$:     MOV #-11,SOURCE          :INDICATE 1ST RECORD OF NEW FRAME
        JSR PC,ENDSCN            :RETURN TO ORIGIN, CHECK FRAME COUNT
        BR PHOTO                 :PLAY BACK NEXT FRAME


        .SBTTL  ABORT SCAN
ZAP:    JSR PC,CRLF
        ASK <SCAN ABORTED>
THUD:   ASK <RESET SWITCH 4>
        TST TFLAG                :WAS THIS A REAL RUN?
        BPL 1$                   :IF NOT, NO FILE WAS OPENED
        JSR R5,@DEVICE           :CLOSE DATA FILE
        4
1$:     CLR TFLAG                :RESET THE TEST FLAG
        JMP MON                  :EXIT DIRECTLY TO MONITOR
        .SBTTL  INITIALIZE SCAN

SCAN:   CLR R5                  ;R5 IS NSCANS COUNTER
        CMP PATERN,#-1          ;OFFSET WILL BE ZERO UNLESS....
        BNE 1$                  ;   RASTER SCAN IS BEING DONE
        MOV NPOINT,CNTDWN       ;THEN THE DATA POINTER WILL BE
        ASL CNTDWN              ;   DISPLACED BY NPOINT WORDS

1$:     MOV XTL,R3              ;GET X-TRAVEL....
        MOV XTH,R2
        TST XDIR                ;POS OR NEG MOVE?
        BPL 2$                  ;BR IF POSITIVE
        JSR PC,NEGATE           ;MAKE TRAVEL DISTANCE NEGATIVE
        MOV R3,XTL              ;   AND STORE IT
        MOV R2,XTH

2$:     TST BACKUP              ;IS BACKUP DESIRED?
        BPL START               ;IF NOT, BRANCH
        ASHC #-1,R2             ;DIVIDE X-TRAVEL BY 2
        BIC #3,R3               ;BE SURE IT'S DIVISIBLE BY 4
        MOV R3,SHIFTL           ;STORE THE RESULT
        MOV R2,SHIFTH

START:  MOV XLBUF(R5),R3        ;GET X-START COORD
        MOV XHBUF(R5),R2

        TST BACKUP              ;IS BACKUP DESIRED?
        BPL 1$                  ;IF NOT, BRANCH
        SUB SHIFTL,R3           ;SUBTRACT THE SHIFT TO GET
        SBC R2                  ;   THE DESIRED STARTING COORD
        SUB SHIFTH,R2

1$:     MOV R3,STARTL           ;STORE X-START
        MOV R2,STARTH
        MOV R3,STOPL
        MOV R2,STOPH

        ADD XTL,STOPL           ;GET X-STOP BY ADDING X-TRAVEL
        ADC STOPH               ;   TO X-START
        ADD XTH,STOPH

        TST XDIR                ;IS IT A POSITIVE MOVE?
        BMI 2$                  ;IF NOT, BRANCH
        SUB RAMP,R3             ;SUBTRACT STANDOFF DISTANCE
        SBC R2                  ;   FROM X-START
        BR 3$
2$:     ADD RAMP,R3             ;ADD STANDOFF FOR NEG MOVE
        ADC R2
3$:     MOV R3,XGOL             ;STORE DESTINATION COORD
        MOV R2,XGOH

        MOV YLBUF(R5),YGOL      ;GET Y-START COORD
        MOV YHBUF(R5),YGOH
        JSR PC,DMOVE            ;MOVE TO STARTING POSITION
        MOV NLINES,LCOUNT       ;PUT NLINES IN COUNTER
        MOV XDIR,VECTOR
        RTS PC
        .SBTTL  SCAN ONE LINE

XSCAN:  MOV PASS-A,MODE             ;TRANSFER INSTRUCTION FOR FORWARD SCAN
        CLR OFFSET
        TST PAT                     ;EDGE OR RASTER?
        BEQ 1$                      ;BR IF EDGE
        CMP VECTOR,XDIR             ;FORWARD OR REVERSE SCAN?
        BEQ 1$                      ;BR IF FORWARD SCAN
        MOV STOPL,R3                ;"STOP" IS START OF REVERSE
        MOV STOPH,R2                ;   SCAN
        CMP #-1,PATERN              ;RASTER SCAN?
        BNE 2$                      ;BR IF BOUSTROPHEDONIC
        MOV PASS-B,MODE             ;REVERSE DATA TRANSFER DIRECTION
        MOV CNTDWN,OFFSET           ;   AND LOAD OFFSET
        BR 2$
1$:     MOV STARTL,R3               ;NORMAL START ON FORWARD
        MOV STARTH,R2               ;   SCAN

2$:     MOV #1Ø,R1                  ;INDICATE X-LOWER LIMIT
        JSR PC,LDVAL                ;LOAD DCRS LIMIT REGISTER

        MOV SPEED,STAT              ;PUT MOTOR SPEED IN CONTROL WORD
        TST VECTOR                  ;POS OR NEG SCAN?
        BMI 3$                      ;BR IF NEG
        BIS POSX,STAT               ;SET THE POS-X DRIVE BIT
        MOV #1Ø24Ø1,MOTEMP          ;SET Y AUTLOK
        BR 4$
3$:     BIS NEGX,STAT               ;SET THE NEG-X DRIVE BIT
        MOV #424Ø1,MOTEMP           ;SET Y AUTOLOK ON

4$:     MOV #6Ø,R1                  ;INDICATE Y-DISPLAY REGISTER
        JSR PC,RDVAL                ;GET CURRENT Y-COORD
        SUB YGOL,R3                 ;COMPUTE DISTANCE FROM Y-TARGET
        BPL .+4                     ;BR IF NON-NEG
        NEG R3                      ;MAKE IT POSITIVE IF NECESSARY
        CMP R3,#1                   ;IS IT WITHIN ONE MICRON?
        BGT 4$                      ;LOOP IF IT ISN'T

        MOV #DATA,R2                ;DATA BUFFER POINTER
        ADD OFFSET,R2               ;POINT TO TOP OF BUFFER IF RASTER SCAN
        MOV NPOINT,R3               ;NO. OF POINTS
        MOV DELTAX,DRNB             ;SET N FOR DIVIDE-BY-N
        INC LIMITS                  ;CLEAR X-LIMIT FLAGS
        MOV SWITCH,DR11A            ;BE SURE SWITCH IS CORRECT
        MOV PSW,TEMP                ;SAVE OLD STATUS WORD
        SPL 6                       ;MASK CLOCK PULSES
        MOV STAT,MOTORS             ;TURN MOTOR ON
        24Ø,24Ø                     ;GIVE MOTORS TIME TO READ SPEED
        MOV MOTEMP,MOTORS           ;TURN ON Y AUTOLOK, LEAVING X MOTORS ON

        TSTB LIMITS                 ;IS LOWER LIMIT FLAG SET?
        BPL .-4                     ;WAIT FOR IT
        RTS PC                      ;RETURN TO EXEC
        .SBTTL  MOVE TO NEXT SCAN LINE

NEXTLN: TST YDIR                        ;WHICH DIR IS Y-STEP?
        BMI 1$                          ;BR IF NEG
        ADD DELTAY,YGOL                 ;NEXT LINE IS AT Y+DY
        ADC YGOH
        BR 2$
1$:     SUB DELTAY,YGOL                 ;SUBTRACT DY FOR NEG STEP
        SBC YGOH

2$:     TST PAT                         ;RASTER OR EDGE SCAN?
        BPL 3$                          ;BR IF EDGE
        COM VECTOR                      ;CHANGE TRACING DIRECTION
        JSR PC,GO.Y                     ;ONLY Y-MOVE REQUIRED FOR RASTER
        RTS PC
3$:     JSR PC,FLYBAK                   ;FLYBACK ALSO NEEDED IF EDGE
        RTS PC                          ;RETURN TO EXEC






        .SBTTL  COMPLETION OF FRAME SCAN

ENDSCN: CMP R5,NSCANS                   ;HAVE WE SCANNED ALL FRAMES?
        BEQ 1$                          ;IF SO, EXIT
        TST (R5)+                       ;INC R5 BY 2
        JMP START                       ;SCAN ANOTHER FRAME

1$:     TST TFLAG                       ;WAS THIS A REAL RUN?
        BPL 2$                          ;IF NOT, FILE WAS NEVER OPENED
        JSR R5,@DEVICE                  ;CLOSE DATA FILE
        4
2$:     JSR PC,HOME                     ;MOVE TO THE ORIGIN
        CLR CBLOCK                      ;BE SURE COMMENT FLAG IS DOWN

        TST (SP)+                       ;POP EXEC RETURN ADDRESS
        RTS PC                          ;    AND RETURN TO MONITOR






        .SBTTL  MOVE CARRIAGE TO THE ORIGIN

HOME:   CLR TFLAG                       ;THERE IS NOT AN OPEN FILE
        CLR XGOL                        ;SET ORIGIN AS DESTINATION
        CLR XGOH
        CLR YGOL
        CLR YGOH
        JSR PC,DMOVE                    ;MOVE TO ORIGIN

        TSTB TPS
        BPL .-4
        MOV #7,TPB                      ;RING THE BELL!
        RTS PC
        .SBTTL  MOVE ROUTINES

GO.X:   MOV #6ØØØ,MOTORS           ;TURN OFF AUTOLOK
        JSR PC,RDXY                ;GET CURRENT COORDINATES
        MOV #XMOVE,RØ              ;INDICATE X SETUP
        JSR PC,MOVSET              ;SET UP X MOVE
        MOV #74Ø1,AUTO             ;SET AUTOLOK FOR X AND Y
        BR MOVCHK

GO.Y:   MOV #6ØØØ,MOTORS           ;TURN OFF AUTOLOK
        JSR PC,RDXY
        MOV #24Ø1,AUTO             ;SET AUTOLOK FOR Y
        COM AUTOWT                 ;WAIT FOR Y AUTOLOK ONLY

MOVE.Y: MOV #YMOVE,RØ              ;INDICATE Y SETUP
        JSR PC,MOVSET

MOVCHK: TST RØ                     ;CHECK FOR AUTOLOK MOVE
        BEQ ALOK                   ;BR IF AUTOLOK ONLY

MOVE:   MOV SWITCH,DR11A           ;BE SURE SCALE INCR SWITCH IS CORRECT
        MOV STAT,MOTORS            ;TURN MOTOR ON OR CHANGE SPEED
        MOV #-1,TEMP               ;RUNAWAY TIMER
        MOV #4Ø1,LIMITS            ;CLEAR LIMIT FLAGS

MOV.W:  BIT #1,SWR                 ;HAS SWITCH Ø BEEN SET?
        BNE PANIC                  ;IF SO, ABORT
        DEC TEMP                   ;RUNAWAY TIMER
        BEQ PANIC                  ;IF IT REACHES Ø, IT'S A RUNAWAY
        BIT R1,LIMITS              ;TEST FOR LOWER LIMIT FLAG
        BEQ MOV.W                  ;LOOP TIL IT'S FOUND

ALOK:   MOV #1,TEMP                ;USE TEMP AS PASS COUNTER
1$:     MOV #4Ø1,LIMITS            ;CLEAR LIMIT FLAGS
        MOV AUTO,MOTORS            ;TURN ON AUTOLOK
        TST AUTOWT                 ;Y ONLY OR BOTH AUTOLOKS?
        BMI 2$                     ;IF ONLY Y AUTOLOK ON
        BIT #1ØØ,LIMITS            ;TEST FOR XUL FLAG
        BEQ .-6                    ;LOOP UNTIL IT'S SET
2$:     BIT #4ØØØØ,LIMITS          ;DO THE SAME FOR THE YUL FLAG
        BEQ .-6                    ;WHEN BOTH ARE SET, YOU'RE THERE
        DEC TEMP                   ;DEC PASS COUNTER
        BPL 1$                     ;MAKE 2ND PASS
        CLR AUTOWT                 ;CLEAR AUTOLOK DIRECTION FLAG
        RTS PC

PANIC:  MOV #6ØØØ,MOTORS           ;TURN THE MOTORS OFF
        MOV #4Ø1,LIMITS            ;CLEAR THE LIMIT FLAGS
1$:     BIT #1,SWR                 ;WAIT UNTIL SWITCH Ø IS RESET
        BNE 1$
        BIT #2Ø,SWR                ;HAS SWITCH 4 BEEN SET?
        BEQ DMOVE                  ;IF NOT, RESUME SCAN
        JMP THUD                   ;SWITCH 4 MEANS ABORT MOVE
```

```
DMOVE:  MOV #6ØØØ,MOTORS            ;TURN OFF AUTOLOK
        JSR PC,RDXY
        MOV #XMOVE,RØ               ;INDICATE X SETUP
        MOV #74Ø1,AUTO              ;SET AUTOLOK FOR X AND Y
        JSR PC,MOVSET
        TST RØ
        BEQ MOVE.Y                  ;IF XMOVE LENGTH ZERO,MOVE ONLY Y
        MOV STAT,MOTEMP             ;SAVE X MOTOR COMMAND
        MOV #YMOVE,RØ               ;INDICATE Y SETUP
        JSR PC,MOVSET
        TST RØ
        BNE MOVE2                   ;BR IF YMOVE NOT ZERO LENGTH
        MOV LOLIMX,R1               ;SET MASK FOR XLL FLAG
        BR MOVE                     ;MOV ONLY X COORDINATE

MOVE2:  MOV SWITCH,DR11A            ;BE SURE SCALE INCR SWITCH IS CORRECT
        MOV #4Ø1,LIMITS             ;CLEAR LIMITS
        BISB STAT+1,MOTEMP+1        ;SET Y TRAVEL BITS IN XWORD
        BIC #2ØØØ,MOTEMP            ;CLR Y-SPEED BIT (Y-SPEED SET BY STAT)
        MOV STAT,MOTORS             ;TURN ON Y MOTOR
        MOV MOTEMP,MOTORS           ;TURN ON X MOTOR, LEAVING ON Y

        MOV #-1,TEMP                ;INITIALIZE RUNAWAY TIMER
1$:     BIT #1,SWR                  ;CHECK RUNAWAY SWITCH
        BNE PANIC
        DEC TEMP                    ;RUNAWAY TIMER
        BEQ PANIC
        TSTB LIMITS                 ;TEST FOR X LOWER LIMIT
        BPL 2$                      ;BRANCH IF NOT FOUND
        BIC #142377,MOTEMP          ;CLEAR X-DIR AND SPEED BITS
        BIS #5ØØ1,MOTEMP            ;X-AUTOLOK BITS
        MOV MOTEMP,MOTORS           ;TURN ON X-AUTOLOK
        BR MOV.W
2$:     TST LIMITS                  ;TEST FOR Y LOWER LIMIT
        BPL 1$                      ;BRANCH IF NOT FOUND
        BIC #34377,MOTEMP           ;CLEAR Y-DIR AND SPEED BITS
        BIS #24Ø1,MOTEMP            ;Y-AUTOLOK BITS
        MOV MOTEMP,MOTORS           ;TURN ON X-AUTOLOK
        MOV LOLIMX,R1               ;MOV.W LOOKS FOR BIT TEST IN R1
        JMP MOV.W

FLYBAK: JSR PC,RDXY                 ;PICK UP CURRENT COORDINATES
        MOV #YMOVE,RØ
        JSR PC,MOVSET               ;SET UP THE Y-MOVE
        TST XDIR
        BMI 1$                      ;NEGATIVE SCAN MEANS POSITIVE RETURN
        MOV NEGX,MOTEMP             ;SET NEGATIVE MOVE BITS
        BR 2$
1$:     MOV POSX,MOTEMP             ;SET POSITIVE MOVE BITS
2$:     MOVB RTNSPD,MOTEMP          ;SET UP X-MOTOR SPEED
        MOV #24Ø1,AUTO              ;SET AUTOLOK FOR Y
        COM AUTOWT                  ;Y AUTOLOK ONLY
        TST RØ                      ;RØ=Ø IF Y-MOVE IS AUTOLOK ONLY
        BNE MOVE2                   ;IF LONG Y MOVE, DO DOUBLE MOVE
        MOV MOTEMP,STAT             ;SINGLE MOVE MOTOR CONTROL WORD
        MOV LOLIMX,R1
        JMP MOVE                    ;MOVE X ONLY
```

```
          .SBTTL  SETUP FOR MOVE

          ;RØ MUST POINT TO XMOVE OR YMOVE

MOVSET:   CLR MOVDIR                  ;Ø MEANS POS MOVE
          MOV 4(RØ),R3                ;LO HALF OF DESTINATION COORD
          MOV 6(RØ),R2                ;HI HALF OF DESTINATION COORD
          MOVB 15(RØ),R1              ;SELECT PROPER DCRS UPPER LIMIT
          JSR PC,LDVAL                ;SET DESTINATION AS UPPER LIMIT
          SUB (RØ),R3                 ;SUBTRACT CURRENT COORD TO GET
          SBC R2                      ;   LENGTH OF MOVE
          SUB 2(RØ),R2
          BMI NEGMOV                  ;BR IF RESULT IS NEGATIVE
          BNE POSMOV                  ;BR IF LARGE POSITIVE NUM
          TST R3                      ;IS IT A ZERO-LENGTH MOVE?
          BNE POSMOV                  ;IF NOT, BRANCH
ZERET:    CLR RØ                      ;INDICATE ZERO MOVE
          RTS PC                      ;IF SO, RETURN

NEGMOV:   MOV 1Ø(RØ),STAT             ;SET "NEG MOV" BIT
          JSR PC,NEGATE               ;MAKE MOVE LENGTH POSITIVE
          COM MOVDIR                  ;-1 FOR NEG MOVE
          BR SPDTST

POSMOV:   MOV 12(RØ),STAT             ;SET "POS MOV" BIT

SPDTST:   MOV R5,-(SP)                ;SAVE R5 ON THE STACK
          JSR PC,SPDSEL               ;SELECT OPTIMUM SPEED
          MOV (R5),R1                 ;PUT THE RAMP IN R1
          MOV (SP)+,R5                ;RESTORE R5
          CMP R4,#5                   ;CHECK FOR AUTOLOK MOVE
          BEQ ZERET                   ;IF AUTOLOK MOVE, R4=5
          SUB R1,R3                   ;SUBTRACT RAMP FROM LENGTH OF MOVE
          SBC R2
          BIC #3,R3                   ;MAKE SURE MOVLEN DIVISIBLE BY 4
          TST MOVDIR                  ;WHICH DIRECTION DOES MOVE GO?
          BPL 1$                      ;BR IF POSITIVE MOVE
          JSR PC,NEGATE               ;NEGATE MOVE LENGTH IF MOVE NEG
1$:       ADD (RØ),R3                 ;ADD MOVLEN TO CURRENT COORDINATE
          ADC R2                      ;   TO GET MOTOR CUTOFF POINT
          ADD 2(RØ),R2

          MOVB SWCSPD(R4),STAT        ;MOVE SPEED TO MOTOR STATUS
          MOVB 14(RØ),R1              ;SELECT LOWER LIMIT REG
          JSR PC,LDVAL                ;SET LOWER LIMIT REG
          MOV 2Ø(RØ),R1               ;IDENTIFY LOWER LIMIT FLAG BIT
          RTS PC


SIGNAL:   CLR DBNS                    ;TURN OFF DIV-BY-N INTERRUPT
          MOV #6ØØØ,MOTORS            ;TURN MOTOR OFF
          MOV TOCKS,TOCS              ;SET UP PROPER DELAY COUNTER
          JSR PC,DELAY                ;GIVE MOTOR TIME TO STOP
          BIT #2Ø,SWR                 ;IS SWITCH 4 SET?
          BNE .+4                     ;IF SO, ABORT SCAN
          RTS PC                      ;OTHERWISE, RETURN TO EXEC
          JMP ZAP
```

```
        .SBTTL  PARAMETER STORAGE

;*********************************************************************
XNOWL:  XMOVE:  Ø                       ; Ø
        Ø                   ;XNOWH      ; 2
XGOL:   Ø                               ; 4
XGOH:   Ø                               ; 6
NEGX:   44ØØØ                           ;1Ø    THESE ADDR'S MUST REMAIN
POSX:   1Ø4ØØØ                          ;12    INTACT AND IN THIS ORDER
        .BYTE 1Ø,Ø          ;XLIMS      ;14
        1ØØ                 ;UPLIMX     ;16
LOLIMX: 2ØØ                             ;2Ø
        2Ø                  ;XDISP      ;22
XRAMPS: 7932.                           ;24
        2Ø76.
         592.
         264.
         1Ø8.
          12.
;*********************************************************************
YNOWL:  YMOVE:  Ø                       ; Ø
        Ø                   ;YNOWH      ; 2
YGOL:   Ø                               ; 4
YGOH:   Ø                               ; 6
        12ØØØ               ;NEGY       ;1Ø    THESE ADDR'S MUST REMAIN
        22ØØØ               ;POSY       ;12    INTACT AND IN THIS ORDER
        .BYTE 5Ø,4Ø         ;YLIMS      ;14
        4ØØØØ               ;UPLIMY     ;16
LOLIMY: 1ØØØØØ                          ;2Ø
        6Ø                  ;YDISP      ;22
YRAMPS: 63Ø.                            ;24
        232.
         74.
         35.
         12.
          6.
;*********************************************************************



SWCSPD: .BYTE 255.,128.,64.,32.,16.,1
```

```
SCAN     MACRO VR05A 10-APR-74 01:39 PAGE 12
PARAMETER STORAGE

      1 RAMP:    100.
      2 AUTO:    0
      3 AUTOWT:  0
      4 PASS.A:  0
      5 PASS.B:  0
      6 MODE:    0
      7 CNTDWN:  0
      8 OFFSET:  0
      9 MOTEMP:  0
     10 VECTOR:  0
     11 SHIFTL:  0
     12 SHIFTH:  0
     13 STARTL:  0
     14 STARTH:  0
     15 STOPL:   0
     16 STOPH:   0
     17 XTL:     0
     18 XTH:     0
     19 LCOUNT:  0
     20 STAT:    0
     21 RTNSPD:  0
     22 NPULSE:  1
     23 SWITCH:  .BYTE 0,100
     24 SOURCE:  0
     25 DSP:     0
     26 PAT:     0
     27 LREP:    0
     28 TEMP:    0
     29 TOCKS:   0
     30 TFLAG:   0
     31 MOVDIR:  0
     32 DBNINT:  RTI
     33
     34          .END

ERRORS DETECTED:  0
FREE CORE:  12288. WORDS
,SCAN,L2/NL:TTM:SYM:BIN:LOC<SCAN
```

```
          UU   TTTTTTTTTTTT    IIIIIIII     LL            TTTTTTTTTTTT   YY        YY
UU        UU   TTTTTTTTTTTT    IIIIIIII     LL            TTTTTTTTTTTT    YY      YY
UU        UU        TT             II       LL                 TT          YY    YY
UU        UU        TT             II       LL                 TT           YY  YY
UU        UU        TT             II       LL                 TT            YYYY
UU        UU        TT             II       LL                 TT             YY
UU        UU        TT             II       LL                 TT             YY
UU        UU        TT             II       LL                 TT             YY
UU        UU        TT             II       LL                 TT             YY
UU        UU        TT             II       LL                 TT             YY
UU        UU        TT             II       LL                 TT             YY
UUUUUUUUUUUU        TT         IIIIIIII     LLLLLLLLLLLL       TT             YY
 UUUUUUUUUU         TT         IIIIIIII     LLLLLLLLLLLL       TT             YY
```

```
   11         ØØØØØØØØØØ                   AAAAAAAAAA   PPPPPPPPPPP    RRRRRRRRRRR
  111        ØØØØØØØØØØØØ                 AAAAAAAAAAAA  PPPPPPPPPPPP   RRRRRRRRRRRR
 1111        ØØ       ØØØ                 AA        AA  PP        PP   RR        RR
   11        ØØ      Ø ØØ                 AA        AA  PP        PP   RR        RR
   11        ØØ     Ø  ØØ                 AA        AA  PP        PP   RR        RR
   11        ØØ    Ø   ØØ    ********     AA        AA  PPPPPPPPPPPP   RRRRRRRRRRRR
   11        ØØ   Ø    ØØ    ********     AA        AA  PPPPPPPPPPP    RRRRRRRRRRR
   11        ØØ  Ø     ØØ                 AAAAAAAAAAAA  PP             RR    RR
   11        ØØ Ø      ØØ                 AAAAAAAAAAAA  PP             RR     RR
   11        ØØØ       ØØ                 AA        AA  PP             RR      RR
11111111     ØØØØØØØØØØØØ                 AA        AA  PP             RR       RR
11111111      ØØØØØØØØØØ                  AA        AA  PP             RR        RR
```

```
UTILTY  MACRO VR05A 10-APR-74 01:40
TABLE OF CONTENTS
```

```
        .TITLE UTILTY
        ;22 JANUARY 1974

        .GLOBL RDASC,RDASC2,RDCHAR,CRLF,TOOLRG,ERA,NPOINT,KEY
        .GLOBL LSTCHR,SIGN,NEGOK,NEGATE,MESAGE,CRT,MFLAG,TEMP
        .GLOBL RDVAL,LDVAL,RDXY,XNOWL,YNOWL,SWCSET,SPDSEL,MON
        .GLOBL DELAY,DSP,XMOVE,VELSEL,TOCS,DATA
        .MCALL .BIN2D,.PARAM
        .PARAM

        DCBUF =167030             ;DCRS BCD READ/WRITE REGISTER
        DCSTAT=167036             ;DCRS CONTROL REGISTER
        SPACE=40
        MINUS=55
        RUBOUT=177

        .MACRO  MSG TEXT
        JSR R5,MESAGE
        .ASCIZ <CR><LF>/TEXT/<CR><LF>
        .EVEN
        .ENDM

        .MACRO  PRINT STRING
        MOV R0,-(SP)
        MOV #STRING,R0
        TSTB (R0)
        BEQ .+16
        TSTB TPS
        BPL .-4
        MOVB (R0)+,TPB
        BR .-16
        JSR PC,CRLF
        MOV (SP)+,R0
        .ENDM

        .MACRO  ECHO CHAR
        TSTB TPS
        BPL .-4
        MOVB CHAR,TPB
        .ENDM

        .MACRO  ECHO3 CHAR
        ECHO #'/
        ECHO CHAR
        ECHO #'/
        .ENDM
        .SBTTL  ASCII-TO-BINARY DECODING

RDASC2: MOV #6,CHRLIM                   ;LONG NUMBER ENTRY POINT
        MOV #-1,DP
        BR S1

RDASC:  MOV #5,CHRLIM                   ;SHORT NUMBER ENTRY POINT
        CLR DP
S1:     MOV #1,LSTCHR

NEWNUM: CLR R4                          ;CHAR COUNTER
        CLR BUF
        CLR BUF+2
        CLR BUF+4
        CLR SIGN

READ:   JSR PC,RDCHAR                   ;PUT CHAR IN R1
        CMP R1,#CR                      ;IS IT A CARRIAGE RETURN?
        BEQ XCR
        CMP R1,#LF                      ;IS IT A LINE FEED?
        BEQ XLF
        CMP R1,#SPACE                   ;IS IT A SPACE?
        BEQ XSP
        CMP R1,#RUBOUT                  ;IS IT A RUBOUT?
        BEQ 2$
        CMP R1,#25                      ;IS IT CTRL/U?
        BNE 1$
        ECHO #136
        ECHO #125                       ;ECHO ↑U
        JSR PC,CRLF                     ;IF SO, GO TO A NEW LINE
        BR NEWNUM                       ;   AND START OVER

1$:     CMP R1,#MINUS                   ;IS IT A MINUS SIGN?
        BNE TSTCHR                      ;IF NOT, BRANCH
        TST NEGOK                       ;IS NEG NUM ALLOWED?
        BPL BADCHR                      ;IF NOT, PRINT ERROR MSG
        TST R4                          ;IS IT FIRST CHAR?
        BNE BADCHR                      ;IF NOT, ERROR
        TST SIGN                        ;HAS MINUS ALREADY BEEN TYPED?
        BMI BADCHR                      ;IF SO, ERROR
        COM SIGN                        ;SIGN=-1 MEANS NEG NUM
        BR READ                         ;GET FIRST DIGIT

2$:     DEC R4                          ;UNCOUNT THE PRECEEDING CHAR
        BPL 3$                          ;BR UNLESS THAT WAS FIRST DIGIT
        TST SIGN                        ;HAD MINUS PREVIOUSLY BEEN TYPED?
        BPL NEWNUM                      ;IF NOT, START OVER
        ECHO3 #MINUS                    ;IF SO, ECHO /-/
        BR NEWNUM                       ;   AND THEN START OVER
3$:     ECHO3 BUF(R4)                   ;ECHO DELETED CHAR BETWEEN SLASHES
        CLRB BUF(R4)                    ;CLEAR DELETED CHAR
        BR READ                         ;GET ANOTHER CHAR
```

```
UTILTY  MACRO VR05A 10-APR-74 01:40 PAGE 3
ASCII-TO-BINARY DECODING

TSTCHR: CMP R1,#60                  ;AT LEAST ASCII ZERO?
        BLT BADCHR
        CMP R1,#71                  ;NOT ABOVE ASCII 9?
        BGT BADCHR
        CMP R4,CHRLIM               ;TOO MANY CHARS?
        BEQ TOOBIG

        MOVB R1,BUF(R4)             ;STORE CHAR IN BUFFER
        INC R4                      ;POINT TO NEXT BUFFER LOC
        BR READ                     ;GET NEXT CHAR

XSP:    DEC LSTCHR                  ;-1 MEANS SPACE
XLF:    DEC LSTCHR                  ; 0 MEANS LF
XCR:    NOP                         ;+1 MEANS CR
        JSR PC,DECODE               ;CONVERT ASCII TO BINARY
        CLR NEGOK
        RTS PC                      ;RETURN TO CALLING ROUTINE


TOOBIG: TST (SP)+                   ;POP THE RTS ADDRESS
TOOLRG: MSG <TOO BIG>
        JMP @ERA                    ;GO TO ERROR RETURN ADDR

BADCHR: MSG WHAT?
        TST (SP)+                   ;POP THE RTS ADDR
        JMP @ERA                    ;GO TO ERROR RETURN ADDR
        .SBTTL  CHOOSE OPTIMUM MOTOR SPEED

SPDSEL: CLR R4                          ;ASSUME ALL SPEEDS VALID
        MOV #4,R1                       ;INITIALIZE DELAY COUNTER FOR Y
        CMP R0,#XMOVE                   ;IS IT AN X-MOVE?
        BNE V1                          ;BRANCH IF YMOVE

VELSEL: MOV SWCSET,R4                   ;PICK UP SWITCH SETTING
        MOV #16.,R1                     ;INITIALIZE DELAY COUNTER FOR X
V1:     MOV R4,R5                       ;COPY R4
        ASL R5                          ;CHANGE R5 TO WORD OFFSET
        ADD R0,R5                       ;R0 CONTAINS XMOVE: OR YMOVE:
        ADD #24,R5                      ;R5 NOW POINTS TO RAMPS LIST
        TST R2                          ;IS IT A VERY LONG MOVE?
        BGT 3$                          ;IF SO, USE MAX ALLOWED SPEED
        MOV R3,-(SP)                    ;SAVE R3 ON THE STACK
        ASHC #-1,R2                     ;DIVIDE MOVE LENGTH BY 2

1$:     CMP R3,(R5)+                    ;IS MOVE LONGER THAN CRITICAL LENGTH?
        BHIS 2$                         ;IF SO, THE POINTER IS SET
        ASR R1                          ;SHORTER DELAY FOR LOWER SPEED
        INC R4                          ;IF NOT, INCREMENT R4
        CMP R4,#5                       ;HAVE WE GONE THRU THE LIST?
        BLT 1$                          ;IF NOT, CHECK THE NEXT VALUE
2$:     MOV (SP)+,R3                    ;RESTORE R3 FROM THE STACK
        TST -(R5)                       ;CANCEL LAST AUTO-INCREMENT

3$:     RTS PC


        .SBTTL  NEGATE DOUBLE PRECISION NUMBER

NEGATE: NEG R2                          ;THIS ROUTINE NEGATES THE
        NEG R3                          ;   DOUBLE WORD INTEGER
        SBC R2                          ;   CONTAINED IN R2 (HIGH)
        RTS PC                          ;   AND R3 (LOW)


        .SBTTL  PAUSE TO LET MOTORS STOP

DELAY:  MOV TICS,CLOCK
1$:     MOV TOCS,R0
        DEC R0
        BGT .-2
        SUB #1,CLOCK
        BGT 1$
        RTS PC
```

```
         .SBTTL  READ CHAR FROM KEYBOARD

RDCHAR:  CLR TKS                      :DISABLE INTERRUPT
         TSTB TKS                     :HAS KEY BEEN HIT?
         BPL .-4
         MOVB TKB,R1                  :MOVE CHAR TO R1
         BIC #1776ØØ,R1               :REDUCE IT TO 7-BIT ASCII
         CMP R1,#CR                   :IS IT A CARRIAGE RETURN?
         BEQ CRLF
         CMP R1,#LF                   :IS IT A LINE FEED?
         BEQ CRLF
         CMP R1,#3                    :IS IT CTRL/C?
         BEQ 1$
         ECHO R1                      :ECHO THE CHAR
         BR ENABLE

1$:      TST MFLAG                    :IS REQUEST FROM MONITOR?
         BMI ENABLE                   :IF SO, USE NORMAL RETURN
         ECHO #136
         ECHO #1Ø3                    :OTHERWISE, ECHO ↑C
         JMP MON                      :   AND GO DIRECTLY TO MONITOR

KEY:     MSG <TYPE ANY KEY TO CONTINUE >
         JSR PC,RDCHAR

CRLF:    ECHO #CR
         ECHO #LF

ENABLE:  MOV #1ØØ,TKS                 :ENABLE KB-INTERRUPT
         RTS PC



         .SBTTL  SEND MESSAGE TO TELEPRINTER

MESAGE:  TSTB (R5)                    :ALL CHARS PRINTED?
         BEQ 1$                       :IF SO, BRANCH
         ECHO (R5)+                   :PRINT THE CHAR
         BR MESAGE                    :   AND GET THE NEXT ONE
1$:      INC R5                       :POINT TO BYTE AFTER NULL
         BIT #1,R5                    :IS R5 AN EVEN NUMBER?
         BEQ 2$                       :IF SO, BRANCH
         INC R5                       :IF NOT, MAKE IT POINT TO NEXT
2$:      RTS R5                       :   WORD BOUNDARY
```

```
        .SBTTL  READ COORDINATES FROM DCRS

RDVAL:  CLR DCSTAT
        BIS #1ØØ,R1                 ;SET "READ" BIT
        MOV R1,DCSTAT               ;LOAD DCRS COMMAND REGISTER
        MOVB #1,DCSTAT+1            ;SET "EXTERNAL" BIT
        TST DCSTAT                  ;LOOP UNTIL "EXT MODE" FLAG
        BPL .-4                     ;   IS SET

        CLR R4                      ;R4 IS PLACE INDICATOR
1$:     MOVB DCBUF,BUF(R4)          ;STORE BCD DIGIT
        BISB #6Ø,BUF(R4)            ;CHANGE BCD TO ASCII
        INC R4
        CMP R4,#6                   ;HAVE 6 DIGITS BEEN READ?
        BNE 1$                      ;IF NOT, READ ANOTHER DIGIT

        CLR SIGN
        MOVB DCBUF,SIGN             ;READ 1 FOR POS, Ø FOR NEG
        DEC SIGN                    ;MAKE IT Ø FOR POS, -1 FOR NEG

        CLR BUF+6
        MOV #-1,DP                  ;INDICATE BIG NUM ALLOWED
        MOV #6,R4                   ;INDICATE 6 DIGIT NUMBER
        JSR PC,DECODE               ;DECODE ASCII INTO BINARY
        CLR DCSTAT                  ;LEAVE EXTERNAL MODE
        RTS PC                      ;RETURN TO CALLING ROUTINE





        .SBTTL  LOAD COORDINATE INTO DCRS

LDVAL:  JSR R5,@44                  ;SAVE THE REGISTERS ON THE STACK
        JSR PC,ENCODE               ;CONVERT BINARY TO BCD

        CLR RØ                      ;RØ IS DIGIT POINTER
        CLR DCSTAT
        MOV R1,DCSTAT               ;LOAD INTERNAL REGISTER ADDR
        MOVB #1,DCSTAT+1            ;SET "EXTERNAL" BIT
        TST DCSTAT                  ;LOOP UNTIL "EXT MODE" BIT
        BPL .-4                     ;   IS SET

1$:     MOVB BUF(RØ),DCBUF          ;LOAD A DIGIT
        INC RØ                      ;COUNT THE DIGIT
        CMP RØ,#6                   ;WAS THAT THE 6TH DIGIT?
        BLT 1$                      ;IF NOT, LOAD ANOTHER
        INC SIGN                    ;Ø FOR NEG, 1 FOR POS
        MOVB SIGN,DCBUF             ;LOAD THE SIGN

        CLR DCSTAT                  ;LEAVE EXTERNAL MODE
        JSR R5,@46                  ;RESTORE THE REGISTERS
        RTS PC                      ;RETURN TO CALLING ROUTINE
```

```
          .SBTTL  DECODE ASCII INTO BINARY

DECODE:   MOV #BUF,ADDR                 ;STORE THE BUFFER ADDRESS
          CLR R2                        ;R2 IS HIGH ORDER WORD
          CLR R3                        ;R3 IS LOW ORDER WORD
          CMP R4,#4                     ;MORE THAN 4 CHARS?
          BLE 1$                        ;IF NOT, BRANCH
          TST DP                        ;BIG NUM ALLOWED?
          BMI 4$                        ;IF SO, BRANCH
1$:       JSR PC,LOWNUM                 ;DECODE LOW ORDER DIGITS
2$:       TST DP                        ;BIG NUM ALLOWED
          BPL 3$                        ;IF NOT, EXIT
          TST SIGN                      ;WAS NUM NEG?
          BPL 3$                        ;IF NOT, EXIT
          JSR PC,NEGATE                 ;MAKE NUMBER NEGATIVE
3$:       RTS PC

4$:       INC ADDR                      ;START OF LOW 4 DIGITS
          SUB #5,R4                     ;ARE THERE 5 OR 6 DIGITS?
          BEQ 5$                        ;BR IF 5
          INC ADDR                      ;NOW IT'S START OF LO 4 DIGITS

5$:       JSR PC,LOWNUM                 ;DECODE LOW 4 DIGITS
          MOV #BUF,R0                   ;ADDR OF HIGHEST DIGIT
          TST R4                        ;5 OR 6 DIGITS?
          BEQ 6$                        ;BR IF 5
          JSR PC,DIGIT6                 ;JSR IF 6

6$:       MOVB (R0),R4                  ;GET THE 10,000'S DIGIT
          BIC #177760,R4                ;CONVERT ASCII TO BCD
          BEQ 2$                        ;IF ZERO, DONE
7$:       ADD #10000.,R3                ;OCTAL 23420
          ADC R2
          SOB R4,7$
          BR 2$                         ;NOW WE'RE DONE

DIGIT6:   MOVB (R0)+,R4                 ;GET THE 100,000'S DIGIT
          BIC #177760,R4                ;CONVERT ASCII TO BCD
          BEQ 2$                        ;IF ZERO, DONE
1$:       ADD #103240,R3                ;LOW 16 BITS OF 303240 (100,000.)
          ADC R2
          INC R2                        ;17TH BIT OF 303240
          SOB R4,1$                     ;DO AGAIN IF NECESSARY
2$:       RTS PC                        ;GO TO 10,000'S PLACE DECODER

LOWNUM:   MOV ADDR,-(SP)                ;MACRO EXPANSION OF THE
          MOV #2,-(SP)                  ;    .D2BIN CONVERSION FROM
          EMT 42                        ;    ASCII TO BINARY
          BVC 1$                        ;V SET IF OUT OF RANGE
          ADD #6,SP                     ;CLEAR 3 WORDS FROM STACK
          JMP TOOBIG                    ;PRINT ERROR MESSAGE
1$:       MOV (SP)+,R3                  ;NUM WAS RETURNED ON STACK
          TST (SP)+                     ;SO WAS STATUS WORD
          RTS PC

```

```
               .SBTTL  ENCODE BINARY INTO BCD

     ENCODE:   CLR BUF
               CLR TEMP
               CLR SIGN
               TST R2                         ;TEST HIGH ORDER WORD
               BPL 1$                         ;BR IF NUM IS POSITIVE
               COM SIGN                       ;SIGN=-1 FOR NEG
               JSR PC,NEGATE                  ;MAKE NUM POSITIVE
               TST R2                         ;RETEST HIGH ORDER WORD
     1$:       BEQ 6$                         ;BR IF HIGH ORDER WORD IS ZERO

               CLR R0                         ;R0 IS COUNTER
     2$:       SUB #103240,R3                 ;LOW 16 BITS OF 303240 (100,000.)
               SBC R2
               DEC R2                         ;17TH BIT OF 303240
               BMI 3$                         ;IF NEG, YOU'VE GONE TOO FAR
               INC R0                         ;COUNT
               BR 2$                          ;   AND DO IT AGAIN

     3$:       MOVB R0,TEMP                   ;STORE THE DIGIT
               ADD #103240,R3                 ;RESTORE THE LAST 100,000.
               ADC R2
               INC R2

               CLR R0                         ;THEN GO TO THE 2ND DIGIT
     4$:       SUB #23420,R3                  ;SUBTRACT 10,000.
               SBC R2
               BMI 5$                         ;IF NEG, YOU'VE GONE TOO FAR
               INC R0                         ;COUNT
               BR 4$                          ;   AND DO IT AGAIN

     5$:       MOVB R0,TEMP+1                 ;STORE THE DIGIT
               ADD #23420,R3                  ;RESTORE THE LAST 10,000.

     6$:       .BIN2D BUF+1,R3                ;CONVERT BINARY TO ASCII

               BIS TEMP,BUF                   ;INSERT HIGH ORDER DIGIT(S)
               BIC #170360,BUF                ;CONVERT ASCII TO BCD
               BIC #170360,BUF+2
               BIC #170360,BUF+4
               RTS PC


               .SBTTL  GET CURRENT X,Y COORDINATES

     RDXY:     MOV #20,R1                     ;INDICATE X-DISPLAY REGISTER
               JSR PC,RDVAL                   ;GET CURRENT X-COORD
               MOV R3,XNOWL                   ;   AND STORE IT
               MOV R2,XNOWL+2
               MOV #60,R1                     ;INDICATE Y-DISPLAY REGISTER
               JSR PC,RDVAL                   ;GET CURRENT Y-COORD
               MOV R3,YNOWL                   ;   AND STORE IT
               MOV R2,YNOWL+2
               RTS PC
```

```
            .SBTTL  DISPLAY DATA ON TERMINAL

CRT:        TST DSP                     ;1ST REQUEST FOR DISPLAY?
            BMI 5$                      ;IF NOT, BRANCH
            COM DSP                     ;HOIST THE FLAG
            MSG <DISPLAY REQUESTED>
            JSR PC,KEY                  ;WAIT FOR KEY TO BE HIT
5$:         JSR R5,MESAGE               ;CLEAR THE CRT SCREEN
            .BYTE 33,14,0
            .EVEN

            MOV #1,LN                   ;NO. OF 1ST DATA POINT ON LINE
            MOV NPOINT,TEMP
            MOV #DATA,R1                ;R1 POINTS TO DATA WORD

            MOV #10.,R0                 ;DISPLAY 10 DATA PTS PER LINE
1$:         MOV #VBUF,R4                ;CHAR BUFFER FOR DISPLAY LINE
            SUB R0,TEMP                 ;POINTS LEFT AFTER THIS LINE
            BPL 2$                      ;IF NON-NEG, BRANCH
            ADD TEMP,R0                 ;IF NEG, TRUNCATE THIS LINE

2$:         JSR R5,@44                  ;SAVE THE REGISTERS
            .BIN2D BUF,LN               ;ENCODE NO. OF 1ST PT. IN LINE
            JSR R5,@46                  ;RESTORE THE REGISTERS
            JSR PC,FILL                 ;PUT RESULT IN LINE BUFFER
            MOV SPACES,(R4)+            ;PUT EXTRA SPACES AFTER PT. NO.

3$:         MOV (R1)+,VALUE             ;GET DATA POINT
            JSR R5,@44                  ;SAVE THE REGISTERS
            .BIN2D BUF,VALUE            ;ENCODE THE DATA VALUE
            JSR R5,@46                  ;RESTORE THE REGISTERS
            JSR PC,FILL                 ;PUT DATA PT. IN LINE BUFFER
            SOB R0,3$                   ;COUNT THE POINT, GET ANOTHER
            CLR (R4)                    ;TERMINATE THE LINE
            PRINT VBUF                  ;WRITE LINE ON TERMINAL SCREEN

            CMP LN,#290.                ;HAVE MORE THAN 29 LINES
            BGT 4$                      ;   BEEN WRITTEN ON TERMINAL?
            TST TEMP                    ;ARE WE OUT OF DATA?
            BLE 4$
            MOV #10.,R0                 ;IF NOT, SET UP ANOTHER LINE
            ADD R0,LN
            BR 1$                       ;   AND PRINT IT
4$:         JMP KEY                     ;RETURN AFTER KEYBOARD CHAR

FILL:       CLR R2
            MOV #4,R3                   ;IGNORE UP TO 4 LEADING ZEROS
1$:         CMPB BUF(R2),#60            ;IS IT ASCII ZERO?
            BNE 2$                      ;IF NOT, THE REST IS SIGNIFICANT
            MOVB #40,BUF(R2)            ;IF SO, SUBSTITUTE A SPACE
            INC R2                      ;POINT TO NEXT CHAR
            SOB R3,1$                   ;   AND CHECK IT
2$:         MOVB #40,BUF+5              ;ADD A PADDING SPACE
            MOV BUF,(R4)+               ;STORE THE ENCODED DATA VALUE
            MOV BUF+2,(R4)+             ;   IN THE LINE BUFFER
            MOV BUF+4,(R4)+
            RTS PC
```

```
UTILTY  MACRO VRØ5A 1Ø-APR-74 Ø1:4Ø PAGE 1Ø
PARAMETER STORAGE

                .SBTTL  PARAMETER STORAGE

        ADDR:   Ø
        SIGN:   Ø
        CHRLIM: Ø
        LSTCHR: Ø
        DP:     Ø
        NEGOK:  Ø
        CLOCK:  Ø
        TICS:   4ØØØØ
        TOCS:   1
        LN:     Ø
        VALUE:  Ø
        SPACES: .BYTE 4Ø,4Ø
        VBUF:   .BLKB 76.
        BUF:    .BLKB 8.

                .END

ERRORS DETECTED:  Ø
FREE CORE:  12366. WORDS
,UTILTY.L2/NL:TTM:SYM:BIN:LOC<UTILTY
```

```
RRRRRRRRRRR   EEEEEEEEEEEE   CCCCCCCCCC    0000000000   RRRRRRRRRRR   DDDDDDDDDD
RRRRRRRRRRRR  EEEEEEEEEEEE  CCCCCCCCCCCC  000000000000  RRRRRRRRRRRR  DDDDDDDDDDD
RR        RR  EE            CC        CC  00        00  RR        RR  DD        DD
RR        RR  EE            CC            00        00  RR        RR  DD        DD
RR        RR  EE            CC            00        00  RR        RR  DD        DD
RRRRRRRRRRRR  EEEEEEEE      CC            00        00  RRRRRRRRRRRR  DD        DD
RRRRRRRRRRR   EEEEEEEE      CC            00        00  RRRRRRRRRRR   DD        DD
RR    RR      EE            CC            00        00  RR    RR      DD        DD
RR     RR     EE            CC            00        00  RR     RR     DD        DD
RR      RR    EE            CC        CC  00        00  RR      RR    DD        DD
RR       RR   EEEEEEEEEEEE  CCCCCCCCCCCC  000000000000  RR       RR   DDDDDDDDDDD
RR        RR  EEEEEEEEEEEE   CCCCCCCCCC    0000000000   RR        RR  DDDDDDDDDD
```

```
     11        ØØØØØØØØØØ                  AAAAAAAAAA   PPPPPPPPPPP   RRRRRRRRRRR
    111       ØØØØØØØØØØØØ                AAAAAAAAAAAA  PPPPPPPPPPPP  RRRRRRRRRRRR
   1111       ØØ       ØØØ                AA        AA  PP        PP  RR        RR
     11       ØØ      Ø ØØ                AA        AA  PP        PP  RR        RR
     11       ØØ     Ø  ØØ                AA        AA  PP        PP  RR        RR
     11       ØØ    Ø   ØØ    ********    AA        AA  PPPPPPPPPPPP  RRRRRRRRRRRR
     11       ØØ   Ø    ØØ    ********    AA        AA  PPPPPPPPPPP   RRRRRRRRRRR
     11       ØØ  Ø     ØØ                AAAAAAAAAAAA  PP            RR    RR
     11       ØØ Ø      ØØ                AAAAAAAAAAAA  PP            RR     RR
     11       ØØØ       ØØ                AA        AA  PP            RR      RR
  11111111    ØØØØØØØØØØØØ                AA        AA  PP            RR       RR
  11111111     ØØØØØØØØØØ                 AA        AA  PP            RR        RR
```

```
RECORD  MACRO VRØ5A 1Ø-APR-74 Ø1:4Ø
TABLE OF CONTENTS
```

```
        .TITLE  RECORD
        ;6 FEBRUARY 1974

        .GLOBL MESAGE,IDHEAD,CBLOCK,RDCHAR,SETDUN,IOMODE,IDBUF
        .GLOBL DISK,TAPE,NPOINT,CRLF,IDCODE,MON,SERIES,NUMBER
        .GLOBL ABC,NWORDS,DATA,IFLAG
        .MCALL .INIT,.OPENI,.OPENO,.READ,.WRITE,.WAIT,.CLOSE
        .MCALL .RLSE,.RADPK,.BIN2O,.BIN2D

        R0=%0
        R1=%1
        R2=%2
        R3=%3
        R4=%4
        R5=%5
        SP=%6
        PC=%7
        TPS=177564
        TPB=177566



        .MACRO PRINT    ADDR
        MOV #ADDR,R1
        JSR PC,TYPE
        .ENDM

        .MACRO MSG      TEXT
        JSR R5,MESAGE
        .ASCIZ /TEXT/
        .EVEN
        .ENDM

        .CSECT
```

```
            .SBTTL  DEVICE AND FUNCTION

DISK:       CLR DEVICE                  ;ZERO MEANS DISK
            MOV DK,DEV                  ;SPECIFY DEVICE FOR LINK BLOCK
            BR FUNC


TAPE:       MOV #-1,DEVICE              ;-1 MEANS TAPE
            MOV MT,DEV                  ;USE MT DRIVER


FUNC:       MOV (R5)+,R4                ;GET FUNCTION CODE
            TST R4                      ;OUTPUT OR INPUT
            BMI 1$                      ;BR IF INPUT
            JSR PC,@OUTPUT(R4)          ;GO TO PROPER OUTPUT ROUTINE
            RTS R5                      ;RETURN TO CALLER

1$:         COM R4                      ;MAKE OFFSET POSITIVE
            CLR INDEX
            JSR PC,@INPUT(R4)           ;GO TO PROPER INPUT ROUTINE
            RTS R5                      ;RETURN TO CALLER


OUTPUT:     O.DATA                      ;Ø MEANS DATA
            O.OPEN                      ;2 MEANS OPEN
            CLOSE                       ;4 MEANS CLOSE
            EOF                         ;6 MEANS ENDFILE
            EOT                         ;1Ø MEANS LOGICAL END OF TAPE
            REWIND                      ;12
            O.DAT1                      ;14 MEANS 1ST DATA BLOCK


INPUT:      I.DATA                      ;Ø MEANS DATA
            I.OPEN                      ;2 MEANS OPEN (PARAMS FROM KB)
            I.OPNP                      ;4 MEANS OPEN, READ PARAMS
            CLOSE                       ;6
            I.DAT1                      ;1Ø MEANS 1ST DATA BLOCK
```

```
        .SBTTL  OUTPUT ROUTINES

O.OPEN: .INIT LINK
TRY2:   MOVB SERIES,FNAM            :PICK UP THE SERIES LETTER
        .BIN2D BUF,NUMBER           :ENCODE THE NUMBER
        MOV #FNAM+1,RØ              :POINT TO NEXT CHAR LOC
        CMPB BUF+3,#6Ø              :IS THERE A LEADING ZERO?
        BEQ 1$                      :IF SO, SKIP IT
        MOVB BUF+3,(RØ)+            :IF NOT, USE DIGIT FOR FILENAME
1$:     MOVB BUF+4,(RØ)+            :LAST DIGIT ALWAYS APPEARS
        CLRB (RØ)                   :INSERT A TRAILING NULL

        .RADPK FNAM                 :PACK FILENAME IN RADIX-5Ø
        MOV (SP)+,FILE              :RESULT IS RETURNED ON THE STACK
        TST (SP)+                   :SO IS A STATUS WORD
        CLR FILE+2                  :FILENAME HAS ONLY 3 CHARS

        INC NUMBER                  :SET UP NUMBER FOR NEXT TIME
        CMP NUMBER,#1ØØ.            :ONLY 2 DIGITS ALLOWED
        BLT 2$                      :IF OK, BRANCH
        MOV #1,NUMBER               :IF TOO BIG, RESET IT TO UNITY
        INC SERIES                  :   AND CHANGE SERIES LETTER
        CMPB SERIES,#'Z             :IT MUST BE AN ALPHABETIC CHAR
        BLE 2$                      :BR IF LEGAL
        MOVB #'A,SERIES             :OTHERWISE, RESET IT TO 'A!

2$:     .OPENO LINK,FILE            :OPEN DATA FILE FOR OUTPUT
        MOV #1,STATUS               :1 MEANS OPEN FOR OUTPUT

        MOV #1,IDCODE               :1 IS THE CODE NUMBER FOR IDENT
        TST CBLOCK                  :   UNLESS A COMMENT FOLLOWS
        BPL 3$
        MOV #4,IDCODE               :THEN 4 IS THE IDENT CODE
3$:     .WRITE LINK,IDHEAD          :WRITE OUT IDENT STRING
        .WAIT LINK

        JSR PC,CRLF
        JSR PC,FMSG                 :PRINT <FILE NAME:  >
        CLR FNAM+6
        PRINT FNAM                  :FILENAME
        MSG <.DAT>
        JSR PC,CRLF

        TST IFLAG                   :HAS IDENT PREVIOUSLY BEEN PRINTED?
        BMI 4$                      :IF SO, BRANCH
        MSG <IDENT:  >
        PRINT IDBUF                 :PRINT IDENT STRING
        JSR PC,CRLF
        COM IFLAG                   :MAKE FLAG NEGATIVE

4$:     TST CBLOCK                  :IS THERE A COMMENT?
        BPL 5$                      :IF NOT, BRANCH
        MOV #2,CODE                 :"COMMENT" CODE
        JSR PC,WRITE                :WRITE OUT COMMENT
        CLR CBLOCK
```

```
5$:      MOV #NPOINT,R1            ;1ST WORD IN PARAM BLOCK
         MOV #DATA,R2
         MOV #141.,RØ              ;TOTAL WORDS IN PARAM BLOCK
         MOV RØ,NWORDS             ;STORE IT IN FRONT OF PARAMS
         JSR PC,SHUNT              ;MOVE PARAM BLOCK TO DATA BUFFER
         MOV #286.,ABC             ;BYTE COUNT
         MOV #3,CODE               ;PARAM BLOCK CODE WORD
         JSR PC,WRITE              ;WRITE OUT PARAM BLOCK
         RTS PC




O.DAT1:  MOV #-1,CODE              ;-1 MEANS 1ST DATA BLOCK
         MOV NPOINT,NWORDS         ;NO. OF DATA WORDS IN RECORD
         BR .+6
O.DATA:  DEC CODE                  ;NEGATIVE OF LINE NUMBER
         MOV NPOINT,ABC            ;GET WORD COUNT
         ADD #2,ABC                ;INCLUDE CODE WORD AND POINT COUNT
         ASL ABC                   ;DOUBLE IT FOR BYTE COUNT
         JSR PC,WRITE              ;WRITE OUT DATA RECORD
         RTS PC




CLOSE:   .CLOSE LINK               ;CLOSE FILE
         .RLSE LINK                ;RELEASE DRIVER
         NEG STATUS                ;INDICATE "CLOSED"
         RTS PC
```

```
WRITE:  ADD #2,ABC                  ;COUNT FORTRAN CODE WORD
        TST IOMODE                  ;IS FTN-TYPE OUTPUT REQUIRED?
        BPL 2$                      ;IF IT IS, BRANCH
1$:     .WRITE LINK,HEADER
        .WAIT LINK
        RTS PC

2$:     CMP ABC,#124.               ;WILL DATA FIT IN ONE RECORD?
        BGT 3$                      ;IF NOT, BRANCH
        MOV #3,FTNCOD               ;IF SO, INDICATE 1ST-AND-LAST RECORD
        BR 1$                       ;    AND WRITE IT OUT

3$:     MOV #1,FTNCOD               ;INDICATE 1ST RECORD
        MOV ABC,BCNT                ;STORE TOTAL BYTE COUNT
        MOV #124.,ABC               ;124 BYTES PER RECORD
        .WRITE LINK,HEADER          ;WRITE OUT 1ST RECORD
        .WAIT LINK

        CLR FCODE                   ;Ø INDICATES INTERMEDIATE RECORD
        MOV #124.,BPERR             ;124 BYTES PER RECORD
        MOV #HEADER+122.,POINT      ;HEADER FOR NEXT RECORD
        BR 5$

4$:     JSR PC,WRT                  ;WRITE OUT THE NEXT RECORD
        ADD #122.,POINT             ;HEADER FOR NEXT RECORD
5$:     SUB #122.,BCNT              ;GET REMAINING BYTE COUNT
        CMP BCNT,#124.              ;WILL THIS BE LAST RECORD?
        BGT 4$                      ;IF NOT, BRANCH

        MOV #2,FCODE                ;INDICATE LAST FORTRAN RECORD
        MOV BCNT,BPERR              ;SIZE OF FINAL RECORD
        JSR PC,WRT                  ;WRITE OUT FINAL RECORD
        RTS PC


WRT:    JSR PC,SAVE                 ;SAVE DATA POINTS, SET UP HEADER
        MOV BPERR,4(R1)             ;ACTUAL BYTE COUNT OF RECORD
        MOV FCODE,6(R1)             ;FORTRAN RECORD CODE
        EMT 2                       ;.WRITE
        .WAIT LINK
        JSR PC,UNSAVE               ;RESTORE DATA POINTS
        RTS PC
        .SBTTL  INPUT ROUTINES

I.OPNP: COM INDEX                       :-1 ASKS FOR PARAMS
I.OPEN: .INIT LINK                      :INIT THE DATASET
        JSR PC,FILNAM                   :GET FILENAME
I.TRY2: .OPENI LINK,FNAME               :OPEN THE INPUT FILE
        TST INDEX                       :MUST PARAMS BE READ?
        BPL 1$                          :IF NOT, BRANCH
        JSR PC,I.ID                     :VERIFY IDENT STRING
        JSR PC,I.PRM                    :READ IN THE PARAMS
1$:     CLR STATUS                      :Ø MEANS OPENED FOR INPUT
        RTS PC


I.DAT1: COM INDEX                       :-1 FOR 1ST DATA RECORD
I.DATA: JSR PC,READ                     :READ A RECORD IN
        TST CODE                        :IS IT A DATA RECORD
        BPL 2$                          :IF SO, BRANCH
        RTS PC                          :IF NOT, RETURN

2$:     TST ABC                         :CHECK BYTE COUNT
        BEQ 3$                          :BR IF ZERO
        TST INDEX                       :HAS DATA ALREADY BEEN READ?
        BMI I.DATA                      :IF NOT, READ ANOTHER RECORD
        BR BADREC                       :IF SO, BAD RECORD

3$:     BIT #4ØØØØ,MODE                 :TEST EOF BIT
        BNE 4$
        MSG <ENDFILE FOUND>
        JMP MON

4$:     CLR ECODE
        MOVB MODE+1,ECODE               :GET STATUS BYTE
        .BIN2O STAT,ECODE               :CONVERT TO OCTAL ASCII
        MOVB #4Ø,STAT+2                 :INSERT A SPACE
        MSG <UNSUCCESSFUL READ - STATUS BYTE>
        PRINT STAT+2                    :PRINT STATUS BYTE
        JMP MON

BADREC: MSG <CAN'T READ RECORD - WRONG FORMAT>
        JSR PC,CRLF
        JSR PC,CLOSE                    :CLOSE THE FILE
        JMP MON                         :EXIT DIRECTLY TO MONITOR
```

```
READ:     .READ LINK,HEADER          ;READ A RECORD IN
          .WAIT LINK
          TST IOMODE                 ;IS FTN-TYPE INPUT REQUIRED?
          BMI 5$                     ;IF NOT, EXIT

          CMP FTNCOD,#3              ;WAS IT THE ONLY RECORD?
          BEQ 5$                     ;IF SO, DONE
          CMP FTNCOD,#1              ;WAS IT 1ST FORTRAN RECORD?
          BEQ 2$                     ;IF SO, CONTINUE
1$:       JMP BADREC                 ;IF NOT, ERROR

2$:       MOV #HEADER+122.,POINT     ;HEADER FOR NEXT READ
3$:       JSR PC,SAVE                ;SAVE DATA, SET UP HEADER
          EMT 4                      ;.READ
          .WAIT LINK
          MOV POINT,R1               ;"POINT" CONTAINS HEADER ADDR
          TST 6(R1)                  ;DO MORE RECORDS FOLLOW?
          BNE 4$                     ;IF NOT, BRANCH
          JSR PC,UNSAVE              ;RESTORE THE DATA POINTS
          ADD #122.,POINT            ;ADDR OF NEXT HEADER
          BR 3$

4$:       CMP 6(R1),#2               ;WAS THAT THE LAST RECORD?
          BNE 1$                     ;IF NOT, ERROR
          JSR PC,UNSAVE              ;RESTORE THE DATA POINTS
5$:       RTS PC


SAVE:     MOV (SP)+,R0               ;PUT RTS ADDR IN R0
          MOV POINT,R1               ;POINTER TO LINE BUFFER HEADER
          MOV (R1),-(SP)             ;SAVE 4 DATA WORDS ON THE STACK
          MOV 2(R1),-(SP)
          MOV 4(R1),-(SP)
          MOV 6(R1),-(SP)
          MOV #124.,(R1)             ;MAX SIZE OF THE BUFFER
          MOV #1,2(R1)               ;FORMATTED BINARY RECORD
          MOV R1,-(SP)               ;START EXPANSION OF .READ
          MOV #LINK,-(SP)            ;   OR .WRITE MACRO
          MOV R0,PC                  ;RTS

UNSAVE:   MOV (SP)+,R0               ;PUT RTS ADDR IN R0
          MOV POINT,R1               ;START OF CORRUPTED DATA
          MOV (SP)+,6(R1)            ;RESTORE 4 DATA WORDS
          MOV (SP)+,4(R1)            ;   FROM THE STACK
          MOV (SP)+,2(R1)
          MOV (SP)+,(R1)
          MOV R0,PC                  ;RTS
```

```
FILNAM: JSR PC,FMSG
        MOV #12.,RØ
1$:     CLRB FNAM-1(RØ)           ;CLEAR FNAM,ENAM
        SOB RØ,1$

2$:     JSR PC,CHECK              ;GET AND CHECK CHAR
        CMP RØ,#6                 ;TOO MANY CHARS?
        BNE 3$                    ;IF NOT, BRANCH
5$:     JMP QM
3$:     TST RØ                    ;IS THIS FIRST CHAR?
        BGT 4$                    ;IF NOT, BRANCH
        TST NUM                   ;IS IT A LETTER?
        BPL 4$                    ;IF SO, BRANCH
        BR 5$
4$:     MOVB R1,FNAM(RØ)          ;STORE THE CHAR
        INC RØ                    ;   AND COUNT IT
        BR 2$                     ;THEN GET ANOTHER

DOT:    CLR RØ
        CLR ENAM
        CLR ENAM+2
1$:     JSR PC, CHECK             ;GET CHAR FOR EXTENSION
        TST NUM                   ;IS IT A DIGIT?
        BPL 3$
2$:     JMP QM                    ;IF SO, THAT'S A NO-NO
3$:     CMP RØ, #3                ;TOO MANY CHARS?
        BEQ 2$
        MOVB R1,ENAM(RØ)          ;STORE THE CHAR
        INC RØ                    ;   AND COUNT IT
        BR 1$                     ;THEN GET ANOTHER

PACK:   .RADPK FNAM               ;PACK 3 CHARS IN RADIX-5Ø
        MOV (SP)+,FNAME           ;RESULT RETURNED ON STACK
        .RADPK FNAM+3
        MOV (SP)+,FNAME+2
        .RADPK ENAM
        MOV (SP)+,FNAME+4
        ADD #6,SP                 ;3 EXTRA WORDS WERE LEFT ON STACK
        RTS PC
```

```
CHECK:  CLR NUM                     ;NUM IS DIGIT FLAG
        JSR PC,RDCHAR
        CMP R1,#3                   ;CTRL/C?
        BNE 1$
        JMP MON                     ;EXIT TO MONITOR
1$:     CMP R1,#177                 ;RUBOUT?
        BNE 2$
        TST (SP)+                   ;POP THE RTS ADDR
        JMP FILNAM                  ;   AND START OVER
2$:     CMP R1,#15                  ;CARRIAGE RETURN?
        BNE 3$
        TST (SP)+
        BR PACK                     ;FILENAME COMPLETE
3$:     CMP R1,#56                  ;DOT?
        BNE 4$
        TST (SP)+
        BR DOT
4$:     CMP R1,#'Z                  ;GREATER THAN ASCII "Z"?
        BGT QMARK
        CMP R1,#'A                  ;AT LEAST ASCII "A"?
        BLT 5$
        RTS PC
5$:     CMP R1,#'Ø
        BLT QMARK
        CMP R1,#'9
        BGT QMARK
        COM NUM                     ;-1 INDICATES DIGIT
        RTS PC

QMARK:  TST (SP)+                   ;POP RTS ADDR
QM:     TSTB TPS
        BPL .-4
        MOVB #'?,TPB                ;PRINT A "?"
        JSR PC,CRLF
        JMP FILNAM                  ;   AND START OVER

```

```
RECORD  MACRO VRØ5A 1Ø-APR-74 Ø1:4Ø PAGE 1Ø
INPUT ROUTINES

I.ID:     JSR PC,READ               :READ A RECORD
          CLR FLAG
          CMP CODE,#4               :4 IS IDENT CODE IF COMMENT FOLLOWS
          BEQ 1$                    :BR IF IT IS IDENT STRING
          CMP CODE,#1               :1 IS IDENT CODE (NO COMMENT)
          BNE 4$                    :IF NOT IDENT, BRANCH
1$:       MOV #DATA,R1              :CHAR POINTER
2$:       TSTB TPS
          BPL .-4
          MOVB (R1)+,TPB            :WRITE IDENT STRING ON TERMINAL
          TSTB (R1)                 :CHECK FOR NULL BYTE
          BNE 2$                    :IF NOT NULL, SEND THE CHAR
3$:       RTS PC

4$:       MSG <NO IDENT STRING>
          JSR PC,CRLF
          COM FLAG                  :-1 MEANS IMPROPER RECORD
          MSG <ACKNOWLEDGE>
          JSR PC,RDCHAR             :INPUT A CHAR
          CMP R1,#'K                :K FOR KILL?
          BNE 3$                    :IF NOT, BRANCH
          JSR PC, CLOSE             :IF SO,CLOSE THE FILE
          JMP MON                   :   AND EXIT TO MONITOR


I.PRM:    TST FLAG                  :HAS BAD RECORD BEEN READ?
          BMI 2$                    :IF SO, "READ" IS UNNECESSARY
1$:       JSR PC,READ               :READ A LOGICAL RECORD
2$:       CLR FLAG
          CMP CODE,#3               :IS IT PARAM BLOCK?
          BEQ 3$                    :IF SO, BRANCH
          CMP CODE,#2               :IS IT A COMMENT?
          BEQ 1$                    :IF SO, READ ANOTHER RECORD
          MSG <CAN'T FIND SCAN PARAMETERS>
          JSR PC,CRLF
          JSR PC,CLOSE              :IF SO, CLOSE THE FILE
          JMP MON                   :   AND EXIT TO MONITOR
3$:       MOV #-1,SETDUN            :THIS CONSTITUTES A COMPLETE SETUP
          MOV #DATA,R1
          MOV #141.,RØ
          MOV #NPOINT,R2

SHUNT:    MOV (R1)+,(R2)+           :TRANSFER SCAN PARAMETERS TO
          SOB RØ,SHUNT              :   PROPER LOCATIONS
          RTS PC
```

```
        .SBTTL  DEVICE CONTROL ROUTINES

EOT:    MOV #2,R0
        BR EOF+4

EOF:    MOV #1,R0
        TST STATUS                      ;WAS LAST OPERATION "OUTPUT"
        BLE 1$                          ;WAS FILE CLOSED?
        JSR PC,CLOSE                    ;IF NOT, CLOSE IT
1$:     MOV #EF,SF                      ;POINTER TO EOF BLECK
        JSR PC,ENTRY                    ;EXECUTE SPECIAL FUNCTION

REWIND: MOV #1,R0
        MOV #RW,SF                      ;POINTER TO REWIND BLOCK

ENTRY:  TST DEVICE                      ;IS IT MAGTAPE?
        BPL ILLCMD                      ;IF NOT, BRANCH
        .INIT LINK
1$:     MOV SF,-(SP)                    ;EXPANSION OF .SPEC
        MOV #LINK,-(SP)
        EMT 12
        SOB R0,1$
        .RLSE LINK
        RTS PC

ILLCMD: JSR PC,LEGAL
        MSG <ON MAGTAPE>
RTN:    JSR PC,CRLF
        JMP MON

ARG:    JSR PC,LEGAL
        MSG <AFTER OUTPUT>
        BR RTN

LEGAL:  MSG <ONLY LEGAL >
        RTS PC

TYPE:   TSTB (R1)
        BNE 1$
        RTS PC
1$:     TSTB TPS
        BPL .-4
        MOVB (R1)+,TPB
        BR TYPE

FMSG:   MSG <FILE NAME:  >
        RTS PC

        .SBTTL  ERROR PROCESSING ROUTINES

ERRORL: MSG <BUFFER SPACE NOT AVAILABLE>
        JSR PC,CRLF
        JMP MON                 ;EXIT TO MONITOR

ERRORD: CMPB FILE-1,#2          ;FILE ALREADY EXISTS?
        BNE 1$                  ;OTHER PROBLEMS ARE FATAL
        JMP TRY2                ;   BUT A NEW NAME IS AVAILABLE
1$:     MOVB FILE-1,ECODE

EMSG:   MSG <DATA FILE CANNOT BE OPENED - ERROR CODE >
        MOV #"ØØ,ERRC           ;GENERATE ASCII ERROR CODE
        BIT #1Ø,ECODE           ;2-DIGIT NUMBER?
        BEQ 1$                  ;IF NOT, BRANCH
        INC ERRC                ;IF SO, MAKE HIGH DIGIT "1"
1$:     BIC #17777Ø,ECODE       ;LEAVE ONLY LOW DIGIT
        BISB ECODE,ERRC+1       ;CREATE ASCII
        JSR R5,MESAGE
ERRC:   Ø
        .BYTE 15,12,Ø,Ø
        JMP MON                 ;EXIT TO MONITOR

ERRORF: CMPB FNAME-1,#2         ;CAN'T FIND FILE?
        BNE 1$
        MSG <CAN'T FIND FILE>
        JSR PC,CRLF
        MOV #I.TRY2,-(SP)       ;CREATE AN RTS ADDR
        JMP FILNAM              ;ASK FOR NEW FILENAME
1$:     MOVB FNAME-1,ECODE
        BR EMSG




        .SBTTL  LINK AND FILENAME BLOCKS

        ERRORL
LINK:   Ø,Ø,1                   ;LINK BLOCK
DEV:    Ø

        ERRORD,Ø
FILE:   Ø,Ø                     ;DATA FILENAME BLOCK
        .RAD5Ø /DAT/
        Ø,Ø

        ERRORF,Ø
FNAME:  Ø,Ø,Ø                   ;INPUT FILENAME BLOCK
        Ø,Ø

RW:     .BYTE 3,3               ;REWIND BLOCK
        Ø,Ø,Ø

EF:     .BYTE 2,3               ;ENDFILE BLOCK
        Ø,Ø,Ø
```

```
RECORD  MACRO VRØ5A 1Ø-APR-74 Ø1:4Ø PAGE 15
PARAMETER STORAGE

                .SBTTL  PARAMETER STORAGE

        BUF:    .BLKB 6
        FNAM:   .BLKB 8.
        ENAM:   Ø,Ø
        NUM:    Ø
        INDEX:  Ø
        ECODE:  Ø
        DEVICE: Ø
        FLAG:   Ø
        STATUS: Ø
        SF:     Ø
        BCNT:   Ø
        BPERR:  Ø
        FCODE:  Ø
        POINT:  Ø
        SERIES: .ASCIZ /A/
        NUMBER: 1
        SPACES: .BYTE 4Ø,4Ø
        DK:     .RAD5Ø /DK/
        MT:     .RAD5Ø /MT/
        STAT:   Ø,Ø,Ø,Ø

        HEADER: 24ØØ6.
        MODE:   1
        ABC:    Ø
        FTNCOD: Ø
        CODE:   Ø
        NWORDS: Ø
        DATA:   .BLKW 12ØØØ.

                .END

ERRORS DETECTED:  Ø
FREE CORE:  11966. WORDS
,RECORD.L2/NL:TTM:SYM:BIN:LOC<RECORD
```