B01

EOFIMIRI: SEQ	00010000	770715	PDP10 411	-AHDRIDZRKLESEQ	00013000C

.RE" *.

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZRKL-E-D
PRODUCT NAME:	RK11/RK05 DYNAMIC TEST
DATE CREATED:	APRIL, 1977
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	JIM KAPADIA
REVISED BY:	PERVEZ ZAKI TOM SAWYER CHUCK HESS

TABLE OF CONTENTS

1.0 ABSTRACT

THE RK11/RK05 DYNAMIC TEST AIMS AT
1. DEMONSTRATING THE ELECTROMECHANICAL INTREGRITY OF THE DRIVE.
2. CHECKING THE LINEAR POSITIONER CONTROL AND SPEED CONTROL
3. VERIFYING THE INTREGITY OF THE READ/WRITE LOGIC
4. PROVIDING A TIMER FOR THE SEEK FUNCTION.

THIS IS A TEST ONE LEVEL HIGHER THAN THE BASIC RK11 LOGIC TESTS.

2.0 REQUIREMENTS

2.1 EQUIPMENT

A. PDP11 WITH CONSOLE TELETYPE.
B. 8K OF MEMORY
C. RK11 OR RKV11 CONTROLLER
D. 1-8 RK05 OR RK05F DRIVES (DRIVE TYPES MAY BE MIXED)

2.2 PRELIMINARY PROGRAMS

RK11 LOGIC TEST I (MAINDEC-11-DZRKJ)
RK11 LOGIC TEST II (MAINDEC-11-DZRKK)

2.3 EXECUTION TIME

ERROR FREE FIRST PASS ON PDP11/20 WITH CORE MEMORY TAKES APPROXIMATELY 5 MINUTES (WITHOUT THE SEEK TIMER AND GRAPH, ADDITIONAL 3.5 MINUTES FOR THESE). LESS FOR FASTER MACHINES OR MEMORIES.

3.0 STARTING ADDRESS

200 FOR ANY NORMAL MODE OF OPERATION. ALL SWITCHESS DOWN

210 FOR FUNCTION SELECTING PROGRAM (CONVERSATIONAL MODE).

4.0 PROGRAM CONTROL MODES & OPERATOR ACTION

PAPER TAPE LOADING
RKDP DUMP MODE
RKDP CHAIN MODE
ACT11

4.1 PAPER TAPE LOADING

4.1.1 LOAD PROGRAM INTO MEMORY USING STANDARD PROCEDURE FOR ABSOLUTE TAPES.

4.1.2 MAKE SURE THAT THE DRIVES TO BE CHECKED ARE LOADED WITH DISKS AND ARE IN 'RUN'. 'WRT ENABLE' THEM. CHECK THAT 'WRT PROT' LIGHT ON THESE DRIVES IS OFF. PUT DRIVES THAT ARE NOT TO BE TESTED ON 'LOAD'.

4.1.3 LOAD ADDRESS 200

4.1.4 SET SWITCHES IF DESIRED (SEE SEC 6.0)

PRESS START.

4.1.5 THE PROGRAM IDENTIFIES ITSELF

RK11 DYNAMIC TEST
MAINDEC-11-DZRKL-E

THEN IT PROCEEDS TO FIND WHICH DRIVES ARE PRESENT AND PRINTS OUT THE DRIVES FOUND. IF AN RK-05F IS DETECTED, AN F IS APPENDED TO THE DRIVE NUMBER:

DRIVES PRESENT
0
1

AFTER TYPING OUT THE DRIVE NUMBER THAT IS GOING TO BE TESTED, EXECUTION OF THE TESTS START.

AFTER ALL THE TESTS HAVE BEEN EXECUTED ON ONE DRIVE THEY ARE EXECUTED ON THE NEXT DRIVE, IF PRESENT. THIS IS REPEATED TILL ALL DRIVES ARE TESTED.

AT THE END OF A PASS THE FOLLOWING IS TYPED OUT:

END PASS X X=0,1,2.....

CONTROL IS TRANSFERRED BACK TO THE BEGINNING OF THE PROGRAM AND RE-EXECUTION BEGINS.

4.2 RKDP DUMP MODE

4.2.1 THE PROGRAM IS LOADED BY THE RKDP MONITOR.

4.2.2 SET SA=200. SELECT ANY SWITCHES YOU WANT AND PRESS START.

4.2.3 THE PROGRAM IDENTIFIES ITSELF AND PRINTS OUT:

'TO TEST' DRIVE 'N' HALT PROGRAM, REMOVE RKDP PACK AND REPLACE IT
WITH A WORK PACK, CLEAR LOCATION 40, AND RESTART PROGRAM'

4.3 RKDP CHAIN MODE

THE PROGRAM IS CHAIN LOADED FROM RKDP PACK ON DRIVE 'N'. AFTER IDENTIFYING ITSELF, THE FOLLOWING MESSAGE APPEARS:

'DRIVE 'N' NOT TESTED'

DRIVE 'N' WILL NOT BE TESTED SINCE THE RKDP PACK IS ON THAT DRIVE.

4.4 ACT11 MODE

THE PROGRAM IS LOADED BY THE ACT11 MONITOR. AFTER IDENTIFYING ITSELF, ASCERTAINS THE NUMBER OF DRIVES PRESENT AND PROCEEDS TO TEST EACH OF THEM AS BEFORE.

5.0 DRIVE SELECTION

IF ANY PARTICULAR DRIVE IS TO BE SELECTED FOR TESTING, PUT THAT DRIVE ON 'RUN', 'WRITE ENABLE'. PUT THE REST OF THE DRIVES ON 'LOAD', 'WRITE LOCK'. THEN START AS USUAL.

6.0 SWITCH OPTIONS

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34, THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' WHENEVER THE PROGRAM ENTERS THE SCOPE ROUTINE OR BEGINS A NEW TEST. THE 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

EACH TIME SWITCH SETTING ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED., 'RUBOUT' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE

'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

SW<15>=1	HALT ON ERROR
SW<14>=1	LOOP ON TEST
SW<13>=1	INHIBIT ERROR PRINTOUTS
SW<12>=1	CYCLE ON ERROR TO THE PREVIOUS 'SCOPE' STATEMENT
SW<11>=1	DUMP ALL RK11 REGISTERS ON ERROR
SW<10>=1	RING BELL ON ERROR
SW<09>=1	LOOP ON SPECIFIC ERROR
SW<08>=1	LOOP ON TEST INDICATED BY USER (SEE SEC. 6.8)
SW<06>=1	TYPE SEEK TIMER
SW<05>=1	TYPE THE GRAPHS
SW<04>=1	PRINT THE COMPLETE GRAPH
SW<03>=1	TERMINATE FUNCTION SELECTED BY USER
SW<02>=1	DROP THE DRIVE AFTER MAXIMUM ALLOWABLE NUMBER OF ERRORS OCCUR
SW<00>=1	ASK FOR PATTERN TO BE WRITTEN OR WRITE CHECKED (FUNCTION SELECTION PROGRAM)

6.1 SW<15>

THE PROGRAM HALTS ON ENCOUNTERING AN ERROR, AFTER TYPING OUT THE ERROR MESSAGE AND PERTINENT INFORMATION. PRESSING "CONTINUE" RESTORES NORMAL OPERATION OF THE PROGRAM.

6.2 SW<14>

THE PROGRAM LOOPS ON THE SUBTEST THAT IS BEING EXECUTED WHEN THE SWITCH IS PUT ON. THIS SWITCH IS USED NORMALLY ALONG WITH SW 15.

6.3 SW<13>

THIS SWITCH INHIBITS ALL ERROR MESSAGES. NORMALLY USED WHEN LOOPING ON TEST (SW 14) OR LOOOPING ON ERROR (SW 9).

6.4 SW<12>

THIS SWITCH ALLOWS THE PROGRAM TO CYCLE FROM THE POINT OF ERROR TO THE PREVIOUS SCOPE STATEMENT. NOTE THAT IN DOING SO ANY INITIALIZATION BEING DONE

AT THE BEGINNING OF THE SUBTEST WILL BE DONE AGAIN AND AGAIN. SEE SEC. 6.7 FOR A DIFFERENT KIND OF SCOPE LOOP.

6.5 SW(11)

THIS SWITCH ALLOWS DUMPING OF ALL RK11 REGISTERS ON ENCOUNTERING AN ERROR.

6.6 SW(10)

RINGS A BELL ON ERROR, USEFUL WHEN ERROR TYPEOUT IS INHIBITED.

6.7 SW(09)

THIS SWITCH PROVIDES THE TIGHTEST POSSIBLE SCOPE LOOP. NOTE THAT UNLIKE SW12 THE INITIALIZATION OF PARAMETERS AT THE BEGINNING OF THE SUBTEST MAY NOT BE DONE IN THIS CASE. THIS SWITCH IS HELPFUL WHEN A PARTICULAR PART OF A SUBTEST IS BEING REPEATED USING DIFFERENT PARAMETERS AND YOU WANT TO SCOPE ON THE PARAMETER IN ERROR. (EXAMPLE: RKDA IS BEING WRITTEN AND READ BACK WITH COUNT PATTERNS FROM 1 TO 177777. PATTERN 561 IS GIVING ERROR, YOU MIGHT NOT WANT TO GO THROUGH THE 560 PATTERNS BEFORE HITTING ERROR ON THE 561TH PATTERN. IN THIS CASE SW 9 WILL GIVE YOU A SCOPE LOOP ON THE 561TH PATTERN ONLY.)

6.8 SW(08)

THIS SWITCH IS USED TO SELECT A PARTICULAR TEST FOR EXECUTION. WHEN THE PROGRAM IS STARTED (200) WITH THIS SWITCH SET, THE FOLLOWING MESSAGE APEARS:

OCTAL TEST#?

THE USER SHOULD REPLY WITH THE TEST NUMBER (OCTAL) HE WANTS TO SELECT, FOLLOWED BY CARRIAGE RETURN.

THE SELECTED TEST IS EXECUTED AGAIN AND AGAIN. TO GET OUT OF THIS LOOP, PUT SW 8 BACK TO 0. THIS WILL RESUME NORMAL OPERATION OF THE PROGRAM. NOTE THAT BEFORE TEST 4 CAN BE EXECUTED TEST 2 SHOULD HAVE BEEN DONE AND TEST 6 SHOULD HAVE BEEN DONE BEFORE TEST 7.

6.9 SW(06)

THIS SWITCH WHEN SET MAKES THE PROGRAM TYPE THE SEEK TIMER. THIS SWITCH CAN BE SET OR RESET BEFORE OR DURING THE SEEK TIMER EXECUTION, AND EVEN WHILE THE TYPEOUT IS OCCURING.

6.10 SW<05>

THIS SWITCH MAKES THE PROGRAM TYPE THE GRAPHS. IF RESET BEFORE THE GRAPH-PLOTTING ROUTINE IS ENTERED, THE GRAPHS WILL BE SKIPPED ENTIRELY. IT CAN BE RESET EVEN AFTER SOME OF THE POINTS HAVE BEEN PLOTTED, TO SKIP PLOTTING REST OF THE POINTS.

6.11 SW<04>

THIS SWITCH IS USED TO SELECT THE COMPLETE GRAPH OUTPUT (SEEK TIMES OF ALL CYLINDERS ARE PLOTTED) NORMALLY WHEN THIS SWITCH IS NOT SET, THE SMALL GRAPH (ONLY SELECTED CYLINDERS PLOTTED) IS PRINTED OUT.

6.12 SW<03>

THIS SWITCH WHEN SET TERMINATES THE EXECUTION OF THE FUNCTION SELECTED BY THE USER (SA=210). A NEW FUNCTION MAY BE INITIATED NOW. IF YOU WANT TO KEEP ON LOOPING ON THE SAME FUNCTION, PUT SW 3 DOWN. SEE SEC. 9.0.

6.13 SW<02>

THIS SWITCH ALLOWS THE PROGRAM TO DROP A DRIVE FROM THE SELECTION LIST AND TESTING, AFTER MAXIMUM ALLOWABLE ERROR COUNT (TOTAL NUMBER OF ERRORS) ON THAT DRIVE IS EXCEEDED. THE MAXIMUM ALLOWABLE ERROR COUNT IS 6, AFTER 6 ERRORS HAVE OCCURED THE DRIVE IS DROPPED AND A MESSAGE (DRIVE # XXXXX DROPPED) IS PRINTED.

6.14 SW<00>

THIS SWITCH IS TO BE USED WITH THE FUNCTION SELECTION PROGRAM (SA=210). IF A WRITE OR A WRITE CHECK FUNCTION IS SELECTED WITH THIS SW SET, THE PROGRAM WILL ASK FOR THE PATTERN TO BE WRITTEN OR WRITE CHECKED (PATRN?). THE USER SHOULD TYPE IN THE (OCTAL) PATTERN. THIS PATTERN WILL BE WRITTEN (OR WRITE CHECKED) ON THE DISK. FOR FURTHER INFORMATION REFER TO SEC. 9.0.

PROGRAM DESCRIPTION

THE FIRST TEST IS AIMED AT DETECTING IMPEPENDING ELECTRO- MECHANICAL FAILURES IN THE DRIVE AND INNER/OUTER LIMIT SWITCHES.

IN THE NEXT TWO TESTS, THE DISK IS FORMATTED AND CHECKED FOR CORRECT FORMATTING. IF THE DISK IS AN RK-05F, THE ENTIRE DISK IS FORMATTED EACH TIME THE EVEN DRIVE IS TESTED. NO FORMATTING IS DONE WHEN THE ODD DRIVE IS TESTED. THE DISK IS CHECKED EACH TIME FOR PROPER FORMAT, HOWEVER.

IN NEXT TWO TESTS THE SEEK LOGIC, POSITIONER, ETC ARE CHECKED OUT BY DOING IMPLIED SEEK, USING TWO DIFFERENT SEEKING PATTERNS. THE FIRST ONE IS A DECREASING SAW-TOOTH PATTERN (0-312-0-311-0-310....), THE SECOND ONE IS A CONVERGING-DIVERGING PATTERN (0-312-1-311-2-310....). ON GETTING AN ERROR, FURTHER ANALYSIS IS DONE TO FIND OUT MORE ABOUT THE NATURE OF ERROR. MANY TIMES ADDITIONAL INFORMATION IS GIVEN FOR THE CONVIENCE OF THE USER. RETRIES ARE DONE WHENEVER AN ERROR OCCURS.

IN THE SUBSEQUENT TESTS EXTENSIVE WRITING IS DONE USING MORE THAN 2000 DIFFERENT PATTERNS. THE DATA IS READ, (SOFTWARE) COMPARED, AND WRITE CHECKED.

EVERYTIME AN ERROR OCCURS RETRIES ARE DONE, TO CHECK IF IT WAS A RECOVERABLE ERROR OR NOT. THE USER CAN CHANGE THE PATTERNS TO BE WRITTEN ON THE DISK. THE DATA TRANSFER BUFFERS CAN BE RE-LOCATED BY THE USER TO DIFFERENT PARTS OF MEMORY. REFER TO LOCATIONS 'PBUF0', 'PBUF1', 'PAT1', 'PTRN01' IN THE LISTINGS FOR MORE DETAILS. SEE SEC 7.1.

THE SHUNT CURRENT CHANGE TEST WRITES, READS AND CHECKS FOR ERRORS ON CYLINDERS 127 AND 128. THIS REGION HAS CRITICAL "PACKING DENSITY" TO "WRITE CURRENT" RATIOS.

THE SEEK TIMER PROVIDES SEEK TIMES AND GRAPHS AS EXPLAINED IN SEC 8.0

A FUNCTION SELECTION SUB-PROGRAM IS PROVIDED FOR USER SELECTION OF FUNCTIONS. SEE SEC 9.0

EVERY TEST IN THE PROGRAM IS PRECEEDED BY AN EXPLANATION OF THAT TEST. THE USER IS ADVISED TO REFER TO THAT, IF MORE INFORMATION IS NEEDED.

7.1 PERMISSIBLE USER PROGRAM MODIFICATIONS

THE USER CAN MAKE MINOR CHANGES IN POINTERS, TABLES, ETC. TO TAKE CARE OF HIS SPECIAL NEEDS. IT IS ADVISABLE TO MAKE CHANGES IF ANY, RIGHT AT THE BEGINING.

7.1.1 SEEK TIMING CAN BE DONE BETWEEN ANY TWO CYLINDERS, BY MAKING CHANGES DESCRIBED IN THE CYLINDER ADDRESS TABLE AT LOCATIONS 'SOAD' AND 'SIAD' IN THE LISTINGS.

7.1.2 IN CASE YOU HAVE A LINE PRINTER AND WANT YOUR OUTPUT ON THE LINE PRINTER, CHANGE LOCATION '$TPS' TO 177514 AND LOCATION '$TPB' TO 177516 (LINE PRINTER VECTORS).

7.1.3 INPUT/OUTPUT DATA BUFFERS (FROM WHERE DATA TRANSFERS WILL BE DONE TO AND FROM THE DISK) CAN BE RELOCATED TO ANYWHERE IN THE 28K OF MEMORY (DO NOT OVERLAY THE PROGRAM). THIS CAN BE DONE BY CHANGING THE CONTENTS OF LOCATIONS 'PBUF0' AND 'PBUF1' TO THE STARTING ADDRESSES OF THE TWO USER SELECTED BUFFERS. IT SHOULD BE NOTED THAT EACH OF THE TWO BUFFERS SHOULD BE 768 (DECIMAL) WORD LONG.

7.1.4 FOUR DIFFERENT PATTERN GENERATOR ROUTINES HAVE BEEN USED IN THIS PROGRAM: A. PTGEN0 B. PTGEN1 C. PTGEN2 D. PTGEN3. THEY HAVE BEEN DESCRIBED IN DETAIL AT CORRESPONDING LOCATIONS IN THE LISTING. THE ORDER IN WHICH THEY ARE CALLED IS DESCRIBED AT THE BEGINING OF TEST 6. THIS CALLING ORDER CAN BE CHANGED BY MAKING CHANGES IN THE FOUR POINTERS A.'PAT0' B. 'PAT1' C. 'PAT2' D. 'PAT3'. THESE 4 POINTERS CONTAIN THE STARTING ADDRESS OF EACH ROUTINE.

7.1.5 AS A SPECIAL CASE OF THE ABOVE, YOU CAN WRITE THE SAME TWO (OR ONE) PATTERN/S ON THE ENTIRE DISK USING 'PTGEN0' ROUTINE. TO WRITE THE SAME ONE PATTERN:
CHANGE LOCATION 'PAT1' TO 'PTGEN0' (STARTING ADDRESS OF PTGEN0)
CHANGE LOCATION 'PAT2' TO 'PTGEN0' (STARTING ADDRESS OF PTGEN0)
CHANGE LOCATION 'PAT3' TO 'PTGEN0' (STARTING ADDRESS OF PTGEN0)
FILL LOCATIONS 'PTRN01' AND 'PTRN02' WITH THE PATTERN YOU WANT.
TO WRITE 2 DIFFERENT PATTERNS (IN ALTERNATING SECTORS):
CHANGE 'PAT1', 'PAT2' AND 'PAT3' AS DESCRIBED ABOVE.
FILL 'PATRN01' AND 'PATRN02' WITH THE TWO PATTERNS YOU WANT.

7.1.6 IN TEST 10, IF YOU WANT TO WRITE AND CHECK CYLINDERS 127 AND 128 WITH PATTERNS OTHER THAN THE 12 USED, CHANGE ANY OR ALL OF THE 12 POINTERS 'SP1' THROUGH

'SP12' TO CONTAIN PATTERNS YOU WANT.

8.0 SEEK TIMER & GRAPHS

THE LAST TEST IN THIS PROGRAM IS THE SEEK TIMER. IN ORDER TO TIME THE SEEKS, THE SECTOR COUNTER HAS BEEN USED AS A TIME BASE. THUS THE ACCURACY OF THE TIMES RECORDED IS AS GOOD AS THE ACCURACY OF THE SECTOR COUNTER (WHICH IN TURN DEPENDS ON THE ROTATION SPEED OF THE DISK).

IN THE FIRST PART OF THIS TIMER, SOME CRITICAL SEEKS HAVE BEEN TIMED (CYLINDERS 0-1, 179-181, 0-3, 0-16, 0-32, 0-202, 0=100) EACH SEEK IS DONE 100 TIMES, TIMES ARE RECORDED, THEN THE TIMES ARE SORTED OUT AND A PRINTOUT IS GIVEN SHOWING HOW MANY TIMES A PARTICULAR SEEK TIME WAS OBTAINED. EXAMPLE: SEEK BETWEEN 0 AND LAST CYLINDER WAS DONE 100 TIMES. 99 TIMES A SEEK TIME OF 95 MS WAS OBTAINED, ONCE IT GAVE 100 MS. THIS GIVES THE USER AN IDEA OF HOW CONSISTENT ARE THE SEEK TIMES.

IF YOU WANT TO TIME SEEK BETWEEN ANY OTHER SET OF CYLINDERS, YOU CAN DO BY FOLLOWING THE INSTRUCTIONS AT LOCATION 'SOAD' IN LISTINGS. SEE SEC 7.1

IN THE SECOND PART, A GRAPH OF THE 'CYLINDER SEEKED FROM 0' IS PLOTTED AGAINST 'SEEK TIME'. TWO GRAPHS ARE AVAILABLE, NORMALLY THE SMALL GRAPH IS PRINTED OUT. THE SMALL GRAPH PLOTS THE SEEK TIMES FOR SELECTED CYLINDERS (ABOUT 49) COVERING THE RANGE FROM CYLINDER 0 TO 202. IT GIVES THE USER A QUICK SEEK CHARACTERISTICS OF A DRIVE.

THE OPTIONAL COMPLETE GRAPH (SW 4) GIVES A GRAPH SIMILAR TO THE ABOVE ONE, BUT PLOTS ALL THE CYLINDERS (203).

THE GRAPH SHOWN ON LAST PAGE IS A SAMPLE OUTPUT. IT SHOULD BE REALIZED THAT DIFFERENT DRIVES MAY HAVE A SLIGHTLY DIFFERENT CHARACTERISTIC.

9.0 FUNCTION SELECTION PROGRAM

THIS PROGRAM GIVES THE USER A CAPABILITY TO SELECT A FUNCTION AND EXECUTE IT, FROM THE CONSOLE TELETYPE.

STARTING ADDRESS=210

ON STARTING THE PROGRAM AT 210, THE FOLLOWING QUESTION APPEARS:

FUNCTION?

THE REPLY SHOULD BE:

WR	FOR WRITE
WC	FOR WRITE CHECK
RD	FOR READ
RC	FOR READ CHECK
CR	FOR CONTROL RESET
DR	FOR DRIVE RESET
SK	FOR SEEK DR

ALL COMMANDS SHOULD BE TERMINATED BY A CARRIAGE RETURN. DEPENDING ON WHICH FUNCTION IS GIVEN THE

FOLLOWING QUESTIONS APPEAR:

RKBA? TYPE IN THE BUS ADDRESS (OCTAL) FOLLOWED BY A C.R.

RKDA? TYPE IN THE DISK ADDRES (OCTAL) FOLLOWED BY C.R.

IF A NON-EXISTENT CYLINDER OR SECTOR IS SELECTED, THE QUESTION IS REPEATED AGAIN.

#WORDS? TYPE IN THE NUMBER OF WORDS YOU WANT TO TRANSFER. IT SHOULD BE IN OCTAL. THUS IF YOU WANT TO READ A SECTOR TYPE IN 400 FOLLOWED BY C.R. ANY NUMBER OF WORDS CAN BE TRANSFERRED DEPENDING ON THE BUFFER SIZE AVAILABLE.

FOR A WRITE FUNCTION: IF SW0 IS SET TO 1 THE PROGRAM WILL ASK FOR THE DATA PATTERN TO BE WRITTEN:
PATRN? THE USER SHOULD TYPE IN THE DATA PATTERN (OCTAL) TO BE WRITTEN, FOLLOWED BY <CR>. THE PATTERN WILL BE WRITTEN ON THE DISK. NOTE THE NUMBER OF WORDS TO BE WRITTEN AND THE DISK ADDRESS SHOULD BE SPECIFIED.

FOR A WRITE CHECK FUNCTION: IF SW 0 IS SET TO 1, THE USER IS ASKED FOR THE PATTERN TO BE WRITE CHECKED: PATRN? THE USER SHOULD TYPE IN THE (OCTAL) PATTERN.

FOR A SEEK FUNCTION: CYL1? CYL2? IN REPLY TO THESE, TYPE IN THE CYLINDER NUMBERS (OCTAL) BETWEEN WHICH THE SEEK IS TO BE DONE. IF A NON EXISTENT CYLINDER IS TYPED IN THE QUESTION IS REPEATED AGAIN.

THE FUNCTION IS EXECUTED AGAIN AND AGAIN. TO GET OUT OF THIS LOOP SW 3 SHOULD BE SET, AT THIS POINT THE QUESTION (FUNCTION?) IS ASKED AGAIN.

IF UPON EXECUTION OF A FUNCTION AN ERROR OCCURS IT IS REPORTED. ALL SWITCH OPTIONS WHICH APPLY TO ANY OTHER ERROR, ALSO APPLY TO THIS ERROR.

IF ON INPUTTING A NUMBER OR COMMAND A MISTAKE IS MADE, THE INPUT STRING CAN BE DELETED BY HITTING 'RUBOUT' KEY, THE NEW STRING CAN BE TYPED IN AGAIN.

10.0 ERROR INFORMATION

WHENEVER AN ERROR MESSAGE IS PRINTED OUT, ALL REGISTERS AND OTHER DATA PERTAININNG TO THE ERROR ARE ALSO GIVEN. RKDS, RKER...RKBA INDICATE THE CONTENTS OF THE CORRESPONDING REGISTERS AT THE TIME OF ERROR.

EVERY ERROR MESSAGE CONTAINS A PC. THIS PC INDICATES THE POSITION IN PROGRAM WHERE THE ERROR CALL IS LOCATED. THE ERROR MESSAGE, BECAUSE OF PRACTICAL CONSIDERATIONS IS MADE SHORT AND MEANINGFUL. THE USER IS ADVISED TO LOOK UP THE PC IN THE PROGRAM LISTING, WHERE HE WILL FIND MORE INFORMATION ABOUT THE ERROR. IN MANY INSTANCES, A SINGLE FAULT WILL GIVE RISE TO MORE THAN ONE ERROR REPORT. A LITTLE DELIBERATION AND CAREFUL EXAMINATION OF THE DATA GIVEN WILL BE CERTAINLY VERY HELPFUL. A BRIEF EXPLANATION OF WHAT IS BEING CHECKED IN THE SUBTEST IS GIVEN AT THE BEGINNING OF EVERY SUBTEST. ALL THE NUMBERS GIVEN WITH ERROR MESSAGES ARE IN OCTAL.

AT TIMES WHEN AN ERROR OCCURS BESIDES THE ERROR PRINTOUT MORE PRINTOUTS OCCUR. THEY ARE GIVEN TO HELP THE USER UNDERSTAND THE PROBLEM.

11.0 UNEXPECTED TIMEOUTS AND RK11 INTERRUPTS

WHEN AN UNEXPECTED TIMEOUT OCCURS, THE PC AT WHICH IME OUT OCCURRED IS TYPED OUT AND THE PROGRAM HALTS. IF IT IS INTACT, IT CAN BE RESTARTED BY PRESSING CONTINUE.

IF AN UNEXPECTED RK11 INTERRUPT OCCURS THE PROGRAM TYPES OUT THE PC AT WHICH THE INTERRUPT CAME IN AND THEN HALTS. PRESSING CONTINUE WOULD RESTART THE PROGRAM FROM BEGINNING.

12.0 COMMONLY USED SUBROUTINES

A BRIEF EXPLAINATION OF EVERY SUBROUTINE IS GIVEN IN THE LISTINGS (JUST BEFORE THE CODE FOR THAT SUBROUTINE). ALL SUB-ROUTINES ARE LISTED IN THE 'TABLE OF CONTENTS' FOUND AT THE BEGINNING OF LISTINGS. THESE ARE TWO WAYS IN WHICH ROUTINES ARE CALLED, 1. JSR PC,ROUTINE 2. THROUGH AN ENCODED TRAP INSTRUCTION. THE LOWER BYTE OF THE 'TRAP' INSTRUCTION IS USED TO INDEX THROUGH THE TRAP TABLE

($TRPAD) FOR THE STARTING ADDRESS OF THE DESIRED ROUTINE.

13.0 SAMPLE GRAPH AND SEEK TIMER OUTPUTS

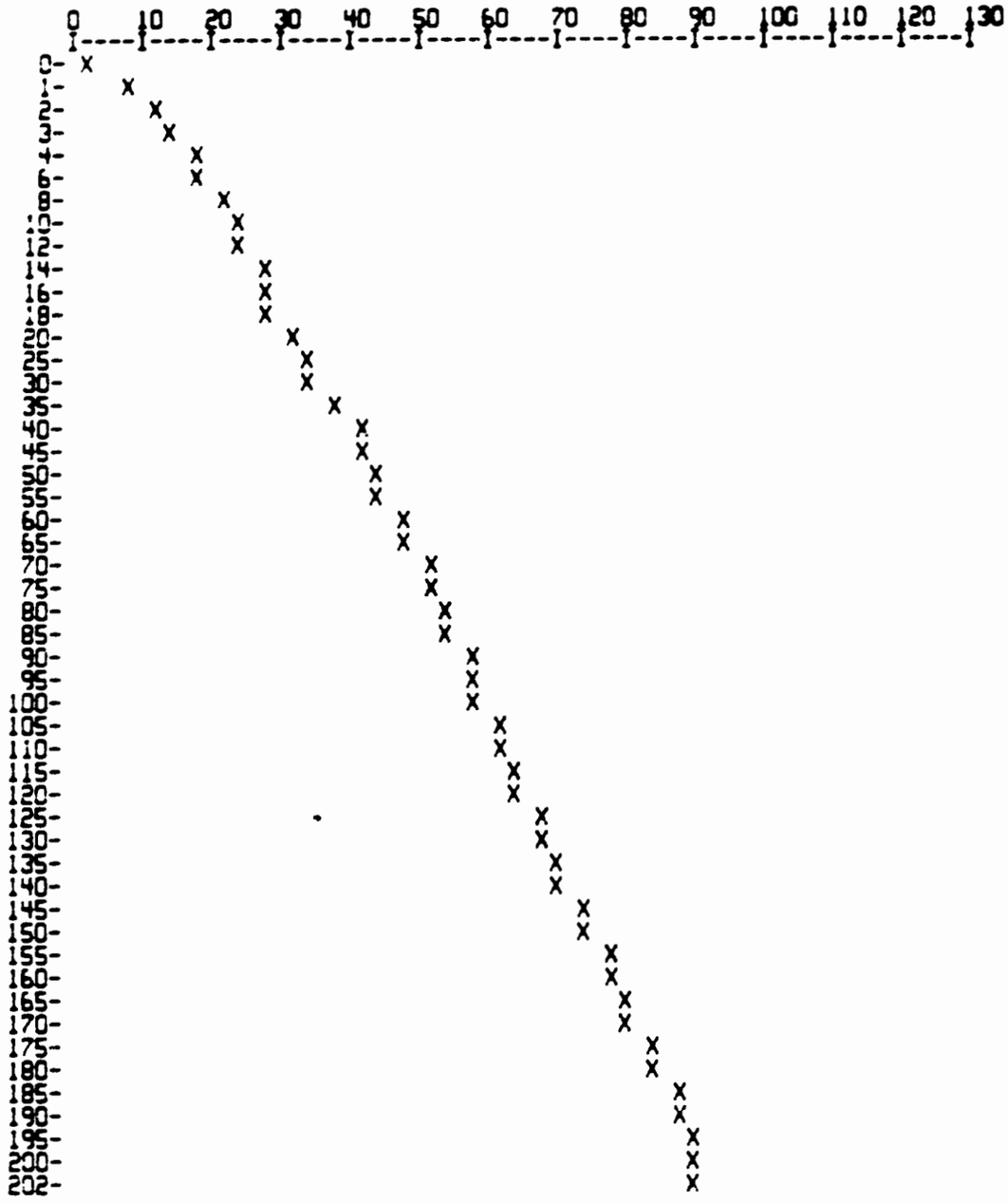
'# OF SEEKS' INDICATES THE NUMBER OF TIMES A PARTICULAR 'SEEK TIME' WAS OBTAINED. NOTE THAT TIMES ARE RECORDED FOR BOTH FORWARD AND REVERSE SEEKS, BETWEEN A SET OF CYLINDERS.

SEEK TIME SCALE FACTOR=0.01 MILI SECS

# OF SEEKS	SEEK TIME	# OF SEEKS	SEEK TIME
CYLS:0-202			
FRWRD		REVRSE	
100	9075	100	9075
CYLS:0-1			
FRWRD		REVRSE	
100	825	100	1155
CYLS:179-181			
FRWRD		REVRSE	
100	1155	100	1155
CYLS:0-3			
FRWRD		REVRSE	
100	1485	100	1485
CYLS:0-16			
FRWRD		REVRSE	
100	3135	100	3135
CYLS:0-32			
FRWRD		REVRSE	
100	3795	100	3795
CYLS:0-100			
FRWRD		REVRSE	
100	5775	100	5775

X AXIS - SEEK TIME - MILI SECS
Y AXIS - CYLINDER SEEKED FROM 0

SAMPLE OUTPUT



```
                        .TITLE  MD-11-DZRKL-E, RK11-RK05 DYNAMIC TEST
                        ;*COPYRIGHT (C) 1974,1977
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY JIM KAPADIA
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
                        ;*JANUARY 1975
                        ;*
                        ;*REVISED MARCH 1976 BY TOM SAWYER
                        ;*REVISED BY CHUCK HESS, AUGUST, 1976

                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH                  USE
                        ;*      ------          --------------------
                        ;*        15            HALT ON ERROR
                        ;*        14            LOOP ON TEST
                        ;*        13            INHIBIT ERROR TYPEOUTS
                        ;*        12            CYCLE ON ERROR TO PREVIOUS 'SCOPE'
                        ;*        10            BELL ON ERROR
                        ;*         9            LOOP ON ERROR
                        ;*         8            SELECT TEST TYPED IN BY USER
                        ;*         6            EXECUTE THE SEEK TIMER (TEST 11)
                        ;*         5            TYPE THE SEEK TIMER GRAPHS (TEST 11,
                        ;*         4            TYPE THE COMPLETE GRAPH (ALL SEEK TIMES
                        ;*                      NOTE, OTHERWISE YOU GET SMALL GRAPH
                        ;*         3            TERMINATE FUNCTION SELECTED BY USER
                        ;*                      (FOR FUNCTION SELECTION PROGRAM SA=210
                        ;*         2            DROP THE DRIVE AFTER MAXIMUM ALLOWABLE
                        ;*                      NUMBER OF ERRORS HAVE OCCURED
                        ;*         0            ASK FOR PATTERN TO BE WRITTEN (OR WRITE
                        ;*                      CHECKED), IN FUNCTION SELECTION PROGRAM
                        ;*        11            DUMP OUT ALL RK11 REGISTERS ON ERROR


                        ;*      YOU ARE ADVISED TO READ THE DOCUMENT FOR THIS PROGRAM.
                        ;*      FUNCTION SELECTION PROGRAM STARTS AT 210.
```

```
                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                  STACK=  1100
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0=     %0              ;;GENERAL REGISTER
000001                  R1=     %1              ;;GENERAL REGISTER
000002                  R2=     %2              ;;GENERAL REGISTER
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

                        ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7

                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
```

```
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

                        .SBTTL  TRAP CATCHER
```

```
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
                                .=174
000174  000000                DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  002462                JMP     @#START ;;JMP TO STARTING ADDRESS OF PROGRAM

000210                                .=210
000210  105237  001216                INCB    FFUNC           ;SET FLAG INDICATING SELECTION OF
000214  000137  002462                JMP     @#START         ;FUNCTION PROGRAM.
                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
000220                                $SVPC=.                 ;SAVE PC
000046                                .=46
000046  015330                        $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                                .=52
000052  000000                        .WORD   0               ;;2)SET LOC.52 TO ZERO
000220                                .=$SVPC                 ;; RESTORE PC
```

```
                                .SBTTL  COMMON TAGS

                                ;;*****************************************************************
                                ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
                                ;*USED IN THE PROGRAM.

                                        .=1100
001100                          $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                  $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102  000                     $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000                     $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                  $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000                     $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001                     $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                          .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                          .WORD   0
001134  000                     $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000                     $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                          .WORD   0
001140  177570                  SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                  DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560                  $TKS:   177560                  ;;TTY KBD STATUS
001146  177562                  $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564                  $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                  $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000                     $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002                     $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012                     $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000                     $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES.
001160  000000                  $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                                ;;WHICH  ($REG0) WAS OBTAINED
001162  000000                  $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001164  000000                  $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001166  000000                  $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001170  000000                  $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001172  000000                  $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001174  000000                  $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001176  000000                  $REG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
001200  000000                  $REG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
001202  000000                  $REG10: .WORD   0               ;;CONTAINS (($REGAD)+20)
001204  000000                  $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001206  177607  000377            $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001212  077                     $QUES:  .ASCII  /?/             ;;QUESTION MARK
001213  015                     $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001214  000012                  $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
```

```
                                ;IN CASE YOU WANT THE OUTPUT TO COME OUT ON LINE PRINTER, (IF YOU HAVE
                                ;ONE), MAKE THE FOLLOWING CHANGES ABOVE:

                                ;CHANGE CONTENTS OF '$TPS' TO 177514  (LPT VECTOR)
                                ;CHANGE CONTENTS OF '$TPB' TO 177516    ( " ")

                                ;TAGS AND GENERAL DATA AREA

001216  000000                FFUNC:  .WORD   0       ;FLAG SET, TO INDICATE ENTRY INTO FUNCTION PROGRAM
001220  000000                XXDPMD: .WORD   0       ;IF PROGRAM LOADED BY XXDP, THE
                                                        ;LOWER BYTE HAS THE DRIVE NUMBER
                                                        ;AND THE UPPER BYTE CONTAINS THE RK05 'XXDP' CODE
001222  000                   LUPSW:  .BYTE   0       ;FLAG, SET TO INDICATE THAT A
                                                        ;PARTICULAR TEST WAS SELECTED BY USER (SW 8)


001223  000                   DRVDON: .BYTE   0       ;CONTAINS NUMBER OF DRIVES THAT HAVE
                                                        ;BEEN ALREADY CHECKED
001224  000                   DRIVS:  .BYTE   0       ;CONTAINS TOTAL # OF DRIVES PRESENT

        001226                          .EVEN


001226  000000                DRVPTR: 0               ;CONTAINS POINTER TO INDICATOR STARTING
                                                        ;WHICH CHECKING SHOULD BE DONE FOR NEXT
                                                        ;AVAILABLE DRIVE
001230  000000                DRIVAD: 0               ;CONTAINS THE ADDRESS OF THE DRIVE
                                                        ;BEING TESTED

001232  000000                DRIV0:  000000          ;THESE ARE FLAGS TO INDICATE
001234  020000                DRIV1:  020000          ;THAT A PARTICULAR DRIVE IS
001236  040000                DRIV2:  040000          ;PRESENT.  BIT 0 IS SET TO
001240  060000                DRIV3:  060000          ;INDICATE THAT.  BITS 13, 14, 15
001242  100000                DRIV4:  100000          ;CONTAIN THE LOGICAL DRIVE
001244  120000                DRIV5:  120000          ;ADDRESS
001246  140000                DRIV6:  140000
001250  160000                DRIV7:  160000


001252  000000                RETRY1: 0               ;GENERAL REGISTERS
001254  000000                RETRY2: 0
001256  000000                RETRY3: 0


001260  000000                INADR:  0               ;CONTAINS INNER ADDRESS
001262  000000                OUTADR: 0               ;CONTAINS OUTER ADDRESS
001264  000000                TIMER:  0


001266  000015                BUFR:   .BLKW   13.     ;GENERAL BUFFERS
001320  000015                BUFR1:  .BLKW   13.
001352  000015                BUFR2:  .BLKW   13.
```

```
                                        ;IN CASE, YOU WANT TO USE BUFFERS STARTING AT SOME OTHER MEMORY
                                        ;ADDRESS YOU CAN DO SO BY CHANGING THE FOLLOWING POINTERS
                                        ;BOTH THE BUFFERS SHOULD BE 768 (DECIMAL) WORDS LONG.

001404  026446          PBUF0:  IOBUF0          ;POINTER TO THE STARTING ADDRESS OF THE
                                                ;BUFFER USED TO READ INTO FROM DIBK.
001406  031446          PBUF1:  IOBUF1          ;POINTER TO STARTING ADDRESS OF BUFFER
                                                ;IN WHICH PATTERNS ARE GENERATED. (WRITING
                                                ;IS DONE FROM THIS BUFFER)
001410  000000          BUFLG0: .WORD   0       ;FLAG FOR 'IOBUF0'
001412  000000          BUFLG1: .WORD   0       ;FLAG FOR 'IOBUF1'

001414  010106          PAT0:   PTGEN0          ;ADRES OF 'PATRN GENERATOR 0'
                                                ;ROUTINE
001416  010170          PAT1:   PTGEN1          ;ADRES OF 'PATRN GENERATOR 1'

001420  010272          PAT2:   PTGEN2          ;ADRES OF 'PATRN GENRATOR 2'

001422  010334          PAT3:   PTGEN3          ;ADRES OF 'PATRN GENRATOR 3'

001424  000000          PRSPAT: .WORD   0       ;CONTAINS THE POINTER TO THE
                                                ;ADRES OF 1 OF THE 3 'PATRN
                                                ;GENRATOR' ROUTINES
001426  000000          NXTPAT: .WORD   0       ;SAME AS ABOVE

001430  000000          PGSUBR: .WORD   0

001432  000000          DSKADR: .WORD   0       ;CONTAINS DISK ADRES (DA)

001434  000000          BUSADR: .WORD   0       ;CONTAINS BUS ADRES (BA)

001436  000000          WRDCNT: .WORD   0       ;CONTAINS WORD COUNT

001440  000000          WDSKAD: .WORD   0       ;CONTAINS DISK ADRES

001442  000000          WBUSAD: .WORD   0       ;CONTAINS BUS ADRES

001444  000000          WWRDCN: .WORD   0       ;CONTAINS WORD COUNT

001446  000000          BUFNO:  .WORD   0       ;CONTAINS STARTING ADRES

001450  000000          ADRES:  .WORD   0       ;OF A BUFFER

                                        ;RK11 REGISTERS
                                        ;IF FOR ANY REASON THE REGISTER ADDRESSES ARE DIFFERENT FROM
                                        ;THESE (BELOW), THE CONTENTS OF THE APPROPRIATE POINTERS SHOULD
                                        ;BE MODIFIED SO THAT THE CORRECT REGISTER ADDRESS IS USED.


001452  177400          RKDS:   177400
001454  177402          RKER:   177402
001456  177404          RKCS:   177404
001460  177406          RKWC:   177406
001462  177410          RKBA:   177410
```

```
1173  001464  177412                RKDA:   177412
1174  001466  177416                RKDB:   177416
1175
1176  001470  000200                RKPRI:  200             ;CONTAINS THE CPU LEVEL (4) AT WHICH
1177                                                         ;RK11 NORMALLY INTERRUPTS.  THIS WORD
1178                                                         ;SHOULD BE CHANGED IF RK11 IS DESIGNATED
1179                                                         ;A BR LEVEL OTHER THAN 5.EXP: IF IT
1180                                                         ;IS CHANGED TO 6, THE CPU LEVEL WOULD
1181                                                         ;BE 1 LESS (5) & HENCE THIS WORD
1182                                                         ;SHOULD BE 240 (BIT POSITIONS ARE
1183                                                         ;IDENTICAL TO THE PRIORITY BITS IN PSW)
1184  001472  000220                RKVEC:  220             ;CONTAINS THE NORMAL VECTOR ADDRESS
1185                                                         ;TO WHICH THE RK11 INTERRUPTS.  IF THE
1186                                                         ;VECTOR ADDRESS HAS BEEN CHANGED, MODIFY
1187                                                         ;THIS WORD.
1188
1189
1190
1191
1192  001474  000000                INDX1:  0               ;GENERAL INDEX REGISTERS
1193  001476  000000                INDX2:  0
1194  001500  000000                INDX3:  0
1195  001502  000000                INDX4:  0
1196
1197  001504  000000                ERCNT1: 0               ;GENERAL REGISTERS
1198  001506  000000                ERCNT2: 0               ;GENERAL REGISTERS
1199  001510  000000                ERCNT3: 0               ;GENERAL REGISTERS
1200  001512  000000                ERCNT4: 0
1201  001514  000000                ERCNT5: 0
1202  001516  000000                ERCNT6: 0
1203  001520  000000                ERCNT7: 0
1204  001522  000000                ERCNT8: 0
1205
1206
1207                                ;*THE FOLLOWING TABLE CONTAINS THE CYLINDERS BETWEEN WHICH THE SEEKS WILL BE
1208                                ;*TIMED. THEY HAVE BEEN SELECTED TO GIVE SOME TYPICAL SEEKS TIMES FOR THE
1209                                ;*3 SEEK SPEEDS. IF FOR  ANY REASON YOU WANT  TO TIME SEEKS BETWEEN ANY
1210                                ;*OTHER SET OF CYLINDERS, MAKE CHANGES IN THE CORRESPONDING SEEK CYLINDER
1211                                ;*ADDRESSES.
1212
1213                                ;*OUTER CYLINDER ADDRESS, FROM WHERE SEEK WILL BE DONE
1214  001524  000000                SOAD:   0               ;CYLINDER 0
1215  001526  000000                        0               ;  "  0
1216  001530  013140                        13140           ;  "  179
1217  001532  000000                        0               ;  "  0
1218  001534  000000                        0               ;  "  0
1219  001536  000000                        0               ;  "  0
1220  001540  000000                        0               ;  "  0
1221
1222                                ;*INNER ADDRESS, TO WHICH SEEK WILL BE DONE
1223  001542  014500                SIAD:   14500           ;CYLINDDER 202, LAST
1224  001544  000040                        40              ;  "  1
1225  001546  013240                        13240           ;  "  181
1226  001550  000140                        140             ;  "  3
1227  001552  001000                        1000            ;  "  16
1228  001554  002000                        2000            ;  "  32
```

```
1229  001556  006200                          6200            ;  "  100
1230
1231
1232
1233                                          ;FOLLOWING POINTERS ARE USED TO TRANSFER CONROL TO THE
1234                                          ;TEST SELECTED BY USING SW 8.  IF ANY MORE TESTS ARE
1235                                          ;ADDED TO THIS PROGRAM ADDITIONAL POINTERS SHOULD BE INSERTED.
1236  001560  004262                          PT1:    TST1+2
1237  001562  004626                          PT2:    TST2+2
1238  001564  005124                          PT3:    TST3+2
1239  001566  005614                          PT4:    TST4+2
1240  001570  006622                          PT5:    TST5+2
1241  001572  007360                          PT6:    TST6+2
1242  001574  010440                          PT7:    TST7+2
1243  001576  012176                          PT10:   TST10+2
1244  001600  012740                          PT11:   TST11+2
1245
1246
1247                                          ;MESSAGES & ASCII STRINGS
1248  001602  005015  044523  000116  MSG1:   .ASCIZ  <15><12>/SIN/
1249
1250  001610  005015  045523  000105  MSG2:   .ASCIZ  <15><12>/SKE/
1251
1252  001616  005015  042524  052123  MSG3:   .ASCIZ  <15><12>/TEST # ABORTED:/
1253  001624  021440  040440  047502
1254  001632  052122  042105  000072
1255
1256  001640  005015  051120  043517  MSG4:   .ASCIZ  <15><12>/PROG ABORTED/
1257  001646  040440  047502  052122
1258  001654  042105     000
1259
1260  001657     015  051012  040505  MSG5:   .ASCIZ  <15><12>/READ HDRS OK FROM CYLB ABOVE/
1261  001664  020104  042110  051522
1262  001672  047440  020113  051106
1263  001700  046517  041440  046131
1264  001706  020102  041101  053117
1265  001714  000105
1266
1267  001716  054105  041520  042124  MSG6:   .ASCIZ  /EXPCTD HDR= /
1268  001724  044040  051104  020075
1269  001732     000
1270
1271  001733     040  050040  036503  MSG7:   .ASCIZ  /  PC= /
1272  001740  000040
1273
1274  001742  005015  047103  051124  MSG10:  .ASCIZ  <15><12>/CNTRL RDY DIDN'T SET/
1275  001750  020114  042122  020131
1276  001756  044504  047104  052047
1277  001764  051440  052105     000
1278
1279  001771     123  041505  051124  MSG11:  .ASCIZ  /SECTR  EXPC P-HDR  RECV P-HDR/
1280  001776  020040  054105  041520
1281  002004  050040  044055  051104
1282  002012  020040  042522  053103
1283  002020  050040  044055  051104
1284  002026     000
```

```
1285
1286  002027     015  051012  053457  MSG12:  .ASCIZ  <15><12>"R/W/S RDY NOT SET"
1287  002034  051457  051040  054504
1288  002042  047040  052117  051440
1289  002050  052105     000
1290
1291  002053     040  052040  054522  MSG13:  .ASCIZ  /  TRY #:/
1292  002060  021440  000072
1293
1294
1295  002064  005015  051104  053111  MSG14:  .ASCIZ  <15><12>/DRIVE /
1296  002072  020105     000
1297
1298  002075     040  020040          BLNK13: .ASCII  /   /
1299  002100     040                  BLNK10: .ASCII  / /
1300  002101     040                  BLNKS9: .ASCII  / /
1301  002102     040                  BLNKS8: .ASCII  / /
1302  002103     040                  BLNKS7: .ASCII  / /
1303  002104     040                  BLNKS6: .ASCII  / /
1304  002105     040                  BLNKS5: .ASCII  / /
1305  002106     040                  BLNKS4: .ASCII  / /
1306  002107     040                  BLNKS3: .ASCII  / /
1307  002110     040                  BLNKS2: .ASCII  / /
1308  002111     040     000          BLNKS1: .ASCIZ  / /
1309
1310        002114                          .EVEN
1311  002114  000000                FDRIVE: 0
1312  002116  000000                FDRVE1: 0
1313  002120  000000                DRHOLD: 0
```

ERROR POINTER TABLE

```
                                .SBTTL  ERROR POINTER TABLE

                                ;#THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;#THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;#LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;#NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
                                ;#NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

                                ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                                ;*      DH              ;;POINTS TO THE DATA HEADER
                                ;*      DT              ;;POINTS TO THE DATA
                                ;*      DF              ;;POINTS TO THE DATA FORMAT


002122                          $ERRTB:

                                        ;ERROR ITEMS TABLE
                                        ;
                                        ;
                                ;ITEM   1

002122  024334                          EM1     ;CNTRL RDY DIDN'T SET AFTER SEEK
002124  025500                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002126  026334                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002130  000000                          0

                                ;ITEM   2

002132  024373                          EM2     ;SIN ON SEEK
002134  025500                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002136  026334                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002140  000000                          0

                                ;ITEM   3

002142  024407                          EM3     ;DRE ON SEEK
002144  025500                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002146  026334                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002150  000000                          0

                                ;ITEM   4

002152  024423                          EM4     ;'ERR' ON SEEK
002154  025500                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002156  026334                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002160  000000                          0

                                ;ITEM   5

002162  024441                          EM5     ;'DRU' ON SEEK; PUT DRIVE ON 'LOAD' BACK TO 'RUN'
002164  025500                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002166  026334                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002170  000000                          0
```

ERROR POINTER TABLE

```
1370                                    ;ITEM   6
1371
1372  002172  024510                            EM6     ;R/W/S RDY NOT SET AFTER SEEK
1373  002174  025500                            DH1     ;PC     RKCS    RKER    RKDS    RKDA
1374  002176  026334                            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1375  002200  000000                            0
1376
1377                                    ;ITEM   7
1378
1379  002202  024545                            EM7     ;SIN ON WRITE FMT
1380  002204  025576                            DH7     ;PC     RKCS    RKER    RKDS    RKDA    CYLINDER
1381  002206  026350                            DT7     ;$ERRPC $REG0   $REG1   $REG2   $REG3   $REG4
1382  002210  000000                            0
1383
1384                                    ;ITEM   10
1385
1386  002212  024564                            EM10    ;'ERR' ON DOING WRITE FMT
1387  002214  025576                            DH7     ;PC     RKCS    RKER    RKDS    RKDA    CYLINDER
1388  002216  026350                            DT7     ;$ERRPC $REG0   $REG1   $REG2   $REG3   $REG4
1389  002220  000000                            0
1390
1391                                    ;ITEM   11
1392
1393  002222  024615                            EM11    ;SIN ON READ FMT
1394  002224  025653                            DH11    ;PC     RKCS    RKER    RKDS    RKDA:
1395                                                    ;DRV#   CYL     SUR     SEC
1396  002226  026366                            DT11    ;$ERRPC $REG0   $REG1   $REG2
1397                                                    ;$REG4  $REG5   $REG6   $REG7
1398  002230  000000                            0
1399
1400                                    ;ITEM   12
1401
1402  002232  024633                            EM12    ;'ERR' ON READ FMT
1403  002234  025576                            DH7     ;PC     RKCS    RKER    RKDS    RKDA    CYLINDER
1404  002236  026350                            DT7     ;$ERRPC $REG0   $REG1   REG2    $REG3   $REG4
1405  002240  000000                            0
1406
1407                                    ;ITEM   13
1408
1409  002242  024655                            EM13    ;WRONG HEADERS FROM 'SEC #
1410  002244  025750                            DH13    ;SECTOR #       HEADER RECVD
1411  002246  000000                            0
1412  002250  015456                            ESR13   ;USE THIS SUBROUTINE FOR TYPING OUT ERROR DATA
1413
1414                                    ;ITEM   14
1415
1416  002252  024704                            EM14    ;ERROR ON IMPLIED SEEK FROM CYLA TO CYLB
1417  002254  025767                            DH14    ;PC     CYLA    CYLB    RKER    RKDS    TRY#
1418  002256  026350                            DT7     ;$ERRPC $REG0   $REG1   $REG2   $REG3   $REG4
1419  002260  000000                            0
1420
1421                                    ;ITEM   15
1422
1423  002262  024757                            MS15    ;READ WRONG HDRS FROM CYLB ABOV
1424  002264  025750                            DH13    ;SEC#   HEADER RECVD
1425  002266  000000                            0
```

```
 1426  002270  015364                          ESR15   ;GO TO "ESR15" FOR TYPING OUT
 1427
 1428                                  ;ITEM   16
 1429
 1430  002272  025021                          EM16    ;READ WRONG, FIRST WORD FROM SECTOR 0, 'CYLB' ON IMPLIED SEEK FROM 'CYLA
 1431  002274  026046                          DH16    ;PC     CYLA    CYLB    EXPCT   RECVD   TRY#
 1432  002276  026350                          DT7     ;SERRPC $REG0   $REG1   $REG2   $REG3   $REG4
 1433  002300  000000                          0
 1434
 1435                                  ;ITEM   17
 1436
 1437  002302  025126                          MS17    ;READ FIRST WORD FROM SECTOR 1, 'CYLB' ABOVE
 1438  002304  026123                          DH17    ;PC     CYLB    EXPCT   RECVD
 1439  002306  026410                          DT17    ;SERRPC $REG0   $REG1   $REG2
 1440  002310  000000                          0
 1441
 1442                                  ;ITEM   20
 1443
 1444  002312  025174                          EM20    ;READ WRONG HEADER ON IMPLIED SEEK FROM 'CYLA' TO 'CYLB'
 1445  002314  025750                          DH13    ;SECTOR # HEADER RECVD
 1446  002316  000000                          0
 1447  002320  015532                          ESR20   ;USE THIS SUBROUTINE FOR TYPING OUT ERROR DATA
 1448
 1449
 1450                                  ;ITEM   21
 1451
 1452  002322  025262                          EM21    ;EROR ON DOING WRITE ON DSK
 1453  002324  025500                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
 1454  002326  026334                          DT1     ;SERRPC $REG0   $REG1   $REG2   $REG3
 1455  002330  000000                          0
 1456
 1457                                  ;ITEM   22
 1458
 1459  002332  025316                          EM22    ;SIN ON DOING WRITE
 1460  002334  025500                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
 1461  002336  026324                          DT1     ;SERRPC $REG0   $REG1   $REG2   $REG3
 1462  002340  000000                          0
 1463
 1464                                  ;ITEM   23
 1465
 1466  002342  025341                          EM23    ;HE ON DOING READ
 1467  002344  025653                          DH11    ;PC     RKCS    RKER    RKDS    RKDA:
 1468                                                  ;DRV#   CYL     SUR     SEC
 1469  002346  026366                          DT11    ;SERRPC $REG0   $REG1   $REG2
 1470                                                  ;$REG4  $REG5   $REG6   $REG7
 1471  002350  000000                          0
 1472
 1473                                  ;ITEM   24
 1474
 1475  002352  025362                          EM24    ;CSE ON READ
 1476  002354  026161                          DH24    ;PC     TRY#    RKCS    RKER    RKDS    RKDA:
 1477                                                  ;DRV#   CYL     SUR     SEC
 1478  002356  026422                          DT24    ;SERRPC $REG10  $REG0   $REG1   $REG2
 1479                                                  ;$REG4  $REG5   $REG6   $REG7
 1480  002360  000000                          0
 :48:
```

```
1482                                    ;ITEM   25
1483
1484    002362  025376                            EM25    ;DATA ERROR ON READ FROM DISK ADDRESS
1485    002364  026266                            DH25    ;WORD#  EXPCT   RECVD   CYL     SUR     SEC
1486    002366  000000                            0
1487    002370  015620                            ESR25   ;USE THIS ROUTINE FOR ERROR REPORTING
1488
1489                                    ;ITEM   26
1490
1491    002372  025440                            EM26    ;ME ON WRT CHK
1492    002374  025653                            DH11    ;PC     RKCS    RKER    RKDS    RKDA:
1493                                                    ;DRV#   CYL     SUR     SEC
1494    002376  026366                            DT11    ;$ERRPC $REG0   $REG1   $REG2
1495                                                    ;$REG4  $REG5   $REG6   $REG7
1496    002400  000000                            0
1497
1498                                    ;ITEM   27
1499
1500    002402  025456                            EM27    ;WRT CHK EROR
1501    002404  026161                            DH24    ;PC     TRY#    RKCS    RKER    RKDS    RKDA:
1502                                                    ;DRV#   CYL     SUR     SEC
1503    002406  026422                            DT24    ;$ERRPC $REG10  $REG0   $REG1   $REG2
1504                                                    ;$REG4  $REG5   $REG6   $REG7
1505    002410  000000                            0
1506
1507                                    ;ITEM   30
1508
1509    002412  025473                            EM30    ;ERROR
1510    002414  025500                            DH1     ;PC     RKCS    RKER    RKDS    RKDA
1511    002416  026334                            DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
1512    002420  000000                            0
1513
1514
1515
1516
1517
1518
```

```
1519
1520                                    ;THIS IS THE HANDLER FOR UNEXPECTED TIME OUT. PRESSING CONTINUE WILL
1521                                    ;RESTART THE PROGRAM.
1522
1523
1524  002422  011600                    BADTMO: MOV     (SP),R0 ;SAVE PC WHERE TIME OUT OCCURED
1525  002424  005740                            TST     -(R0)
1526  002426  022626                            CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
1527  002430  104401  002436                    TYPE    ,65$            ;;TYPE ASCIZ STRING
1528  002434  000407                            BR      64$             ;;GET OVER THE ASCIZ
1529                                    ;;65$:  .ASCIZ  <15><12>/TIMOUT:PC=/
1530  002454                            64$:
1531  002454  010046                            MOV     R0,-(SP)        ;SET UP FOR TYPING OUT PC
1532  002456  104402                            TYPOC                   ;GO TYPE OUT OCTAL PC
1533  002460  000000                            HALT
1534
1535  002462  000005                    START:  RESET                   ;CLEAR THE BUS
1536                                    ;;GIVE DRIVES TIME TO RELOAD HEADS IN CASE OF AN APT START
1537  002464  023737  000042  000046            CMP     @#42,@#46               ;ARE WE IN ACT11 AUTO MODE?
1538  002472  001016                            BNE     STARTA                  ;NO, SKIP DELAY
1539  002474  005077  176764                    CLR     @RKDA                   ;SELECT UNIT 0
1540  002500  012700  000250                    MOV     #250,R0                 ;WAIT FOR..
1541  002504  032777  000200  176740    20$:    BIT     #200,@RKDS              ;DRIVE READY..
1542  002512  001006                            BNE     STARTA                  ;IN CASE..
1543  002514  005001                            CLR     R1                      ;OF APT..
1544  002516  005301                            DEC     R1                      ;START, BUT..
1545  002520  001376                            BNE     .-2                     ;DON'T WAIT..
1546  002522  005300                            DEC     R0                      ;FOREVER.
1547  002524  001367                            BNE     20$
1548  002526  000000                            HALT                            ;RKDS BIT 7 (DIRVE READY) NEVER SET
1549  002530                            STARTA:
1550                                    .SBTTL  INITIALIZE THE COMMON TAGS
1551                                    ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
1552  002530  012706  001100                    MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
1553  002534  005026                            CLR     (R6)+           ;;CLEAR MEMORY LOCATION
1554  002536  022706  001140                    CMP     #SWR,R6 ;;DONE?
1555  002542  001374                            BNE     .-6             ;;LOOP BACK IF NO
1556  002544  012706  001100                    MOV     #STACK,SP       ;;SETUP THE STACK POINTER
1557                                    ;;INITIALIZE A FEW VECTORS
1558  002550  012737  017006  000020            MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
1559  002556  012737  000340  000022            MOV     #340,@#IOTVEC+2 ;;LEVEL 7
1560  002564  012737  017162  000030            MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
1561  002572  012737  000340  000032            MOV     #340,@#EMTVEC+2 ;;LEVEL 7
1562  002600  012737  022702  000034            MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
1563  002606  012737  000340  000036            MOV     #340,@#TRAPVEC+2;LEVEL 7
1564  002614  012737  023010  000024            MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
1565  002622  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;LEVEL 7
1566  002630  005037  001204                    CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
1567  002634  112737  000001  001115            MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
1568  002642  012737  002642  001106            MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1569  002650  012737  002650  001110            MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
1570                                    ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1571                                    ;;EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
1572  002656  013746  000004                    MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
1573  002662  012737  002716  000004            MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
1574  002670  012737  177570  001140            MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWITCH REGISTER
```

```
1575  002676  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
1576  002704  022777  177777  176226          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
1577  002712  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
1578                                                                  ;;AND  THE HARDWARE SWR IS NOT = -1
1579  002714  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
1580  002716  012716  002724          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
1581  002722  000002                          RTI
1582  002724  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
1583  002732  012737  000174  001142          MOV     #DISPREG,DISPLAY
1584  002740  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
1585
1586  002744  004737  020622                  JSR     PC,$TKINT       ;INITIALIZE THE TTY HANDLER
1587  002750  023737  000042  000046          CMP     @#42,@#46               ;ARE WE IN ACT11 AUTO MODE?
1588  002756  001416                          BEQ     69$                     ;YES, SKIP TITLE
1589                                  .SBTTL  TYPE PROGRAM NAME
1590                                  ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1591  002760  005227  177777                  INC     #-1             ;;FIRST TIME?
1592  002764  001043                          BNE     67$             ;;BRANCH IF NO
1593  002766  104401  003024                  TYPE    ,68$            ;;TYPE ASCIZ STRING
1594                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1595  002772  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
1596  002776  001006                          BNE     69$             ;;BRANCH IF YES
1597  003000  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
1598  003006  001005                          BNE     70$             ;;BRANCH IF NO
1599  003010  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
1600  003012  000403                          BR      70$
1601  003014  112737  000001  001134  69$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
1602  003022                          70$:
1603  003022  000424                          BR      67$             ;;GET OVER THE ASCIZ
1604                                  ;;68$:  .ASCIZ  <CRLF>/RK11 DYNAMIC TEST/<15><12>/MAINDEC-11-DZRKL-E/<CRLF>
1605  003074                          67$:
1606  003074  105737  001216          START1: TSTB    FFUNC           ;FUNCTION PROGRAM SELECTED?
1607  003100  001404                          BEQ     7$              ;NO
1608  003102  105037  001216                  CLRB    FFUNC           ;YES, CLEAR THE FLAG
1609  003106  000137  023172                  JMP     @#FUNBEG        ;GO TO 'FUNTION SELECTION PROGRAM'
1610  003112  012700  001220          7$:     MOV     #XXDPMD,R0      ;CLEAR FLAGS FROM
1611  003116  105020                  5$:     CLRB    (R0)+           ;'XXDPMD' TO 'DRIVAD'
1612  003120  020027  001232                  CMP     R0,#DRIVAD+2
1613  003124  001374                          BNE     5$
1614  003126  012701  177770                  MOV     #-10,R1
1615  003132  042720  000003          6$:     BIC     #3,(R0)+        ;CLEAR BIT 0'S IN 'DRIVE
1616  003136  005201                          INC     R1              ;PRESENT' FLAGS.
1617  003140  001374                          BNE     6$
1618
1619                                  ;THE FOLLOWING CODE FINDS OUT THE PROGRAM CONTROL MODE:
1620                                  ;PAPER TAPE (MANUAL), ACT11, RKDP CHAIN OR DUMP
1621
1622  003142  122737  000002  000041          CMPB    #2,41           ;LOADED FROM AN RK05 ?
1623  003150  001160                          BNE     ST2             ;BR IF NOT
1624  003152  013737  000040  001220          MOV     40,XXDPMD       ;GET DEVICE INDICATOR AND DRIVE ADDRESS OF
1625                                                                  ;LOADING RK05
1626  003160  122737  000010  001220          CMPB    #10,XXDPMD      ;DRIVE ADDRESS 7 OR LESS ?
1627  003166  101002                          BHI     2$              ;BR IF YES
1628  003170  105037  001220                  CLRB    XXDPMD          ;DRIVE ZERO LOADED THE PROGRAM
1629  003174  005737  000042          2$:     TST     42              ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
1630  003200  001424                          BEQ     3$              ;BR IF NEITHER
```

```
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/NOT TESTING DRIVE /
                                64$:
                                        CLR     -(SP)           ;CLEAR WORD ON STACK
                                        MOVB    XXDPMD,(SP)     ;GET DRIVE ADDRESS
                                        TYPOS                   ;TYPE THE ADDRESS
                                        .BYTE   1               ;ONLY 1 CHARACTER
                                        .BYTE   0               ;SUPRESS LEADING ZEROS
                                        BR      ST2             ;GET NUMBER OF DRIVES
                                3$:     INC     #-1             ;FIRST TIME THROUGH HERE ?
                                        BNE     ST2             ;BR IF NOT FIRST TIME
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <15><12>/TO TEST DRIVE /
                                66$:
                                        CLR     -(SP)           ;CLEAR WORD ON THE STACK
                                        MOVB    XXDPMD,(SP)     ;GET DRIVE ADDRESS
                                        TYPOS                   ;TYPE THE DRIVE ADDRESS
                                        .BYTE   1               ;ONLY 1 CHARACTER
                                        .BYTE   0               ;SUPRESS LEADING ZEROS
                                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  / HALT PROGRAM, REMOVE RKDP PACK AND REPLACE IT/<15><12>
                                68$:
                                        TYPE    ,71$            ;;TYPE ASCIZ STRING
                                        BR      70$             ;;GET OVER THE ASCIZ
                                ;;71$:  .ASCIZ  /WITH A WORK PACK, CLEAR LOCATION 40, AND RESTART PROGRAM
                                70$:

                        000004  ST2:    MOV     #BADTMO,ERRVEC  ;SET TIMEOUT VECTOR FOR
                                                                ;UNEXPECTED TIME OUT

                                ;THIS CODE FINDS WHICH DRIVES ARE PRESENT & PRINTS OUT THE DRIVE
                                ;DRIVE NUMBERS THAT WERE FOUND ON LINE.

                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/DRIVES PRESENT/
                                64$:
                                        CLR@    DRIVS           ;INITIALIZE NO. OF DRVS PRESENT
                                        CLR     R1
                                        MOV     #DRIVO,R2
                                        CLR     R3              ;INITIALIZE COUNT TO 0
                                1$:     TST     XXDPMD          ;LOADED FROM AN RK05 ?
                                        BEQ     6$              ;BR IF NOT
                                        CMPB    R3,XXDPMD       ;CHECKING THE LOAD DRIVE ?
                                        BEQ     2$              ;BR IF YES
                                6$:     MOV     R1,@RKDA        ;ADRES A DRIVE
                                        TSTB    @RKDS           ;IS IT PRESENT?
                                        BPL     2$              ;NO, BRANCH
                                        INCB    DRIVS           ;INCREMENT TOTAL # OF DRVS
                                        BIS     #1,(R2)         ;SET FLAG INDICATING THIS DRV PRSNT
                                2$:     TST     (R2)+
                                        INC     R3              ;INCREMENT COUNT
```

```
1687  003626  062701  020000            ADD     #20000,R1       ;ADRES THE NXT DRV
1688                                                              ;CHKD ALL 8 DRIVES?
1689  003632  001354                    BNE     1$              ;IF NOT, GO CHK IF NEXT DRV PRSNT
1690
1691  003634  004737  024250            JSR     PC,SIZEF        ;FIND WHICH ARE FS
1692  003640  105737  001224            TSTB    DRIVS           ;WERE ANY DRIVES FOUND?
1693  003644  001010                    BNE     3$              ;YES, BRANCH
1694  003646  104401  003654            TYPE    ,67$            ;;TYPE ASCIZ STRING
1695  003652  000403                    BR      66$             ;;GET OVER THE ASCIZ
1696                                ;;67$:  .ASCIZ  / NONE/
1697  003662                            66$:
1698  003662  000137  015246            JMP     $EOP            ;IF NONE WERE FOUND, GO
1699                                                            ;TO THE END OF PROGRAM
1700  003666  005002            3$:     CLR     R2              ;DRIVE NUMBER
1701  003670  012700  001232            MOV     #DRIV0,R0       ;TABLE OF AVAIL DRIVES
1702  003674  105710            5$:     TSTB    (R0)            ;DRIVE HERE?
1703  003676  001414                    BEQ     4$              ;NO
1704  003700  104401                    TYPE
1705  003702  001213                    $CRLF
1706  003704  010246                    MOV     R2,-(SP)        ;PUSH NO ON THE STACK
1707  003706  104403                    TYPOS                   ;TO TYPE OCTAL NO.
1708  003710     001                    .BYTE   1               ;TYPE 1 DIGIT, SUPRESS LDG 0'S
1709  003711     000                    .BYTE   0
1710  003712  032710  000002            BIT     #2,(R0)         ;IS IT RK05F?
1711  003716  001404                    BEQ     4$              ;NO
1712  003720  104401  003726            TYPE    ,69$            ;;TYPE ASCIZ STRING
1713  003724  000401                    BR      68$             ;;GET OVER THE ASCIZ
1714                                ;;69$:  .ASCIZ  /F/
1715  003730                            68$:
1716  003730  005202            4$:     INC     R2              ;POINT TO NEXT DRIVE #
1717  003732  005720                    TST     (R0)+           ;NEXT DRIVE IN TABLE
1718  003734  020027  001251            CMP     R0,#DRIV7+1     ;ALL DONE?
1719  003740  002755                    BLT     5$              ;NO, CHECK REST
1720
1721                                ;FIND OUT THE FIRST (FROM 0-7) DRIVE THAT IS PRESENT.  PUT THE ADDRESS
1722                                ;OF THAT DRIVE IN 'DRIVAD'.  INDICATE THAT DRIVE # WILL
1723                                ;BE TESTED.
1724
1725  003742  012737  001232  001226 ST3:    MOV     #DRIV0,DRVPTR
1726  003750  105037  001223            CLRB    DRVDON
1727  003754  005037  001230            CLR     DRIVAD
1728  003760  105037  001102            NXTDRV: CLRB    $TSTNM          ;RESET TEST NUMBER TO 1
1729  003764  005037  001112                    CLR     $ERTTL          ;CLEAR ERROR COUNT FOR THIS DRIVE
1730  003770  013701  001226            MOV     DRVPTR,R1
1731  003774  032721  000001            1$:     BIT     #1,(R1)+        ;IS THIS DRIVE PRESENT?
1732  004000  001005                    BNE     2$              ;YES, BRANCH
1733  004002  020127  001252            4$:     CMP     R1,#DRIV7+2     ;CHECKED THE WHOLE LIST?
1734  004006  001372                    BNE     1$              ;NO
1735  004010  000137  015246            JMP     $EOP            ;YES, EXIT
1736  004014  010137  001226            2$:     MOV     R1,DRVPTR       ;NO, GO AHEAD
1737  004020  014104                    MOV     -(R1),R4        ;GET DRIVE NO. TO BE TESTED
1738  004022  005037  002116            CLR     FDRVE1
1739  004026  005037  002114            CLR     FDRIVE          ;SHOWS F IF -1
1740  004032  032704  000002            BIT     #2,R4           ;RK-05F?
1741  004036  001410                    BEQ     7$              ;NO
1742  004040  005237  002116            INC     FDRVE1          ;SHOWS F
```

GET VALUE FOR SOFTWARE SWITCH REGISTER

```
1743  004044  032704  020000            BIT     #20000,R4       ;EVEN DRIVE?
1744  004050  001003                    BNE     7$              ;NO
1745  004052  012737  177777  002114    MOV     #-1,FDRIVE      ;RK05F AND EVEN
1746  004060  042704  000003        7$: BIC     #3,R4
1747  004064  010437  001230            MOV     R4,DRIVAD       ;SET UP DRIVE ADRES
1748  004070  104401  002064            TYPE    ,MSG14
1749  004074  000241                    CLC
1750  004076  006104                    ROL     R4              ;TYPE OUT THE DRIVE NO.
1751  004100  006104                    ROL     R4
1752  004102  006104                    ROL     R4
1753  004104  006104                    ROL     R4
1754  004106  010446                    MOV     R4,-(SP)
1755  004110  104403                    TYPOS
1756  004112     001                    .BYTE   1
1757  004113     000                    .BYTE   0
1758
1759  004114  005737  002116            TST     FDRVE1          ;RK-05F?
1760  004120  001404                    BEQ     6$              ;NO
1761  004122  104401  004130            TYPE    ,65$            ;;TYPE ASCIZ STRING
1762  004126  000401                    BR      64$             ;;GET OVER THE ASCIZ
1763                            ;;65$:  .ASCIZ  /F/
1764  004132                    64$:
1765  004132                    6$:
1766                            ;IF SW 8 IS SET THEN FIND OUT WHICH TEST NUMBER IS
1767                            ;SELECTED AND JUMP TO THAT TEST.
1768
1769  004132  105037  001222            CLRB    LUPSW           ;CLEAR FLAG INDICATING SW8 SET
1770  004136  032777  000400  174774    BIT     #SW8,@SWR       ;SW 8 SET?
1771  004144  001445                    BEQ     TST1            ;NO, BRANCH
1772  004146                    5$:
1773  004146  104401  004154            TYPE    ,67$            ;;TYPE ASCIZ STRING
1774  004152  000410                    BR      66$             ;;GET OVER THE ASCIZ
1775                            ;;67$:  .ASCIZ  <15><12>/OCTAL TEST#?/
1776  004174                    66$:
1777  004174  104412                    RDOCT
1778  004176  012600                    MOV     (SP)+,R0
1779  004200  001762                    BEQ     5$
1780  004202  020027  000011            CMP     R0,#11          ;CHECK TYPED IN TEST #
1781  004206  003357                    BGT     5$              ;IS LEGAL, IF NOT ASK
1782  004210  110037  001102            MOVB    R0,$TSTNM
1783  004214  005300                    DEC     R0              ;FORM POINTERS FOR THE TEST #
1784  004216  006300                    ASL     R0
1785  004220  016037  001560  001106    MOV     PT1(R0),$LPADR  ;ADJUST POINTERS FOR SCOPE
1786  004226  013737  001106  001110    MOV     $LPADR,$LPERR   ;LOOP, ETC.
1787  004234  105237  001222            INCB    LUPSW           ;SET FLAG INDICATING TEST #
1788                                                            ;SELECTED
1789  004240  000177  174642            JMP     @$LPADR         ;GO TO THE TEST SELECTED
1790
1791
1792
1793
1794
1795                            ;ON RECOVERY FROM POWER FALIURE RETURN HERE
1796
1797  004244  005000            PWRFL:  CLR     R0
1798  004246  005001                    CLR     R1
```

```
1799  004250  005201                    1$:     INC     R1
1800  004252  001376                            BNE     1$
1801  004254  105200                            INCB    R0
1802  004256  001374                            BNE     1$
1803
1804
1805
1806                                    ;;*****************************************************************
1807                                    ;*TEST 1         CHECK INNER LIMIT SWITCH & ELECTROMECHANICAL INTEGRITY
1808                                            ;*THIS TEST PERFORMS 200(8) PURE SEEKS FROM CYLINDER 0
1809                                            ;*TO CYLINDER 312 AND BACK TO 0.  IT IS SUPPOSED TO
1810                                            ;*CHECK THE INNER LIMIT SWITCH AND THE MECHANICAL
1811                                            ;*INTEGRITY OF THE DRIVE.  NOTE THAT A VISUAL CHECK FOR
1812                                            ;ANY MECHANICAL FAILURES WOULD BE VERY HELPFUL.
1813                                    ;;*****************************************************************
1814  004260  000004                    TST1:   SCOPE
1815  004262  005000                            CLR     R0              ;INITIALIZE COUNT
1816  004264  005001                            CLR     R1              ;INITIALIZE COUNT FOR # OF SEEKS
1817  004266  005002                            CLR     R2              ;CONTAINS SEEK ADRES
1818  004270  012737  004322  001110            MOV     #20$,$LPERR     ;SET RETURN ADRES FOR LUPING
1819                                                                    ;ON ERROR
1820  004276  012703  001266                    MOV     #BUFR,R3        ;INITIALIZE POINTER TO THE TABLE
1821  004302  012704  177767                    MOV     #-11,R4         ;ALLOW ONLY 9 CNTRL-RDY, SIN, OR R/W/S RDY ERORS
1822  004306  012705  177770                    MOV     #-10,R5         ;ALLOW ONLY 8 DRU+DRE+ERR+DRY ERRORS
1823  004312  000402                            BR      2$
1824
1825  004314  005703                    1$:     TST     R3              ;WAS THERE ANY ERROR?
1826  004316  001403                            BEQ     3$              ;NO, BRANCH
1827  004320  005003                    2$:     CLR     R3              ;CLR EROR FLAG
1828  004322  104415                    20$:    CON.RESET               ;GO DO CNTRL RESET. SUB ROUTINE
1829                                                                    ;AT 'CNT.RST'
1830  004324  104416                            DRV.RESET               ;GO TO 'DRV.RST' & DO DRV RESET
1831
1832  004326  013777  001230  175130  3$:     MOV     DRIVAD,@RKDA    ;ADRES THE DRIVE
1833  004334  050277  175124                    BIS     R2,@RKDA        ;SET SEEK ADRES
1834  004340  105777  175106                    TSTB    @RKDS           ;DRIVE RDY?
1835  004344  100406                            BMI     21$             ;YES
1836  004346  004737  016162                    JSR     PC,GT4RG        ;NO, GET RKCS, ER, DS, DA
1837  004352  104030                            ERROR   30              ;DRIVE RDY BIT IS NOT SET
1838                                                                    ;IN RKDS
1839  004354  005203                            INC     R3              ;SET ERROR FLAG
1840  004356  005205                            INC     R5              ;ALLOW ONLY 5 ERORS, IF MORE
1841  004360  001515                            BEQ     18$             ;ABORT
1842
1843  004362  012777  000011  175066  21$:     MOV     #11,@RKCS       ;GO, SEEK
1844  004370  005200                    4$:     INC     R0              ;WAIT FOR CNTRL RDY
1845  004372  001007                            BNE     5$              ;WAITED LONG?
1846                                                                    ;IF YES, ERROR
1847  004374  004737  016162                    JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
1848  004400  104001                            ERROR   1               ;CNTRL RDY DIDN'T SET AFTER
1849                                                                    ;SEEK WAS DONE TO CYLINDER
1850                                                                    ;SHOWN IN RKDA. GO TO 'RK11 LOGIC TESTS'
1851  004402  005203                            INC     R3              ;SET ERROR FLAG
1852  004404  005204                            INC     R4              ;EXIT THIS TEST IF THERE R 5 OR MORE ERRORS
1853  004406  001502                            BEQ     18$
1954  004410  000403                            BR      6$
```

```
1855  004412  105777  175040          5$:   TSTB    @RKCS           ;DID CNTRL RDY SET?
1856  004416  100364                        BPL     4$              ;IF NOT WAIT FOR IT
1857
1858  004420  005000                  6$:   CLR     R0              ;INITIALIZE COUNT
1859  004422  032777  000100  175022        BIT     #100,@RKDS      ;R/W/S RDY SET?
1860  004430  001010                        BNE     7$              ;YES
1861  004432  005200                        INC     R0              ;WAIT FOR R/W/S RDY
1862  004434  001372                        BNE     6$+2
1863  004436  004737  016162                JSR     PC,GT4RG        ;GET RKCS, ER, DS, DA
1864  004442  104006                        ERROR   6               ;R/W/S RDY DID NOT SET WHEN SEEK
1865                                                                ;WAS DONE TO CYLINDER INDICATED IN RKDA
1866  004444  005203                        INC     R3              ;SET ERROR FLAG
1867  004446  005204                        INC     R4              ;IF MAXM EROR COUNT, ABORT
1868  004450  001461                        BEQ     18$
1869  004452  032777  001000  174772  7$:   BIT     #1000,@RKDS     ;SIN ERROR?
1870  004460  001406                        BEQ     8$              ;NO, BRANCH
1871  004462  004737  016162                JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
1872  004466  104002                        ERROR   2               ;SIN ERROR, ON DOING SEEK TO
1873                                                                ;CYL AS SHOWN IN RKDA
1874  004470  005203                        INC     R3              ;SET ERROR FLAG
1875  004472  005204                        INC     R4              ;IF MAXM EROR COUNT REACHED,
1876  004474  001447                        BEQ     18$             ;ABORT THE TEST
1877  004476  005777  174752          8$:   TST     @RKER           ;DRE ERROR?
1878  004502  100006                        BPL     10$             ;NO, BRANCH
1879
1880  004504  004737  016162                JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
1881  004510  104003                        ERROR   3               ;DRE ON DOING SEEK TO CYLINDER
1882                                                                ;AS SHOWN IN RKDA
1883  004512  005203                        INC     R3              ;SET ERROR FLAG
1884  004514  005205                        INC     R5              ;IF MAXM EROR COUNT REACHED,
1885  004516  001767                        BEQ     8$              ;ABORT THE TEST
1886
1887  004520  005777  174732          10$:  TST     @RKCS           ;'ERR' BIT IN RKCS SET?
1888  004524  100006                        BPL     12$             ;NO, BRANCH
1889  004526  004737  016162                JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
1890  004532  104004                        ERROR   4               ;'ERR' IN RKCS SET, ON DOING SEEK
1891                                                                ;TO CYL AS SHOWN IN RKDA. NOTE
1892                                                                ;WHICH BIT IN RKER SET?
1893  004534  005203                        INC     R3              ;SET ERROR FLAG
1894  004536  005205                        INC     R5              ;IF MAXM EROR COUNT REACHED,
1895  004540  001425                        BEQ     18$             ;ABORT THE TEST
1896
1897  004542  032777  002000  174702  12$:  BIT     #2000,@RKDS     ;DRU SET?
1898  004550  001406                        BEQ     15$             ;NO, BRANCH
1899  004552  004737  016162                JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
1900  004556  104005                        ERROR   5               ;DRU SET, THIS IS AN IRRECOVERABLE
1901                                                                ;ERROR.  HENCE PUT THE DRIVE ON
1902                                                                ;LOAD, BACK TO RUN.  DRU ERROR
1903                                                                ;SHOULD BE CLEARED, IF IT IS NOT
1904                                                                ;1) THE HEAD POSITION TRANSDUCER LAMP
1905                                                                ;IS INOPERATIVE
1906                                                                ;2) OR ERASE OR WRT CURRENT PRESENT
1907                                                                ;WITHOUT 'WRT GATE'
1908  004560  005203                        INC     R3              ;SET EROR FLAG
1909  004562  005205                        INC     R5              ;ALLOW ONLY 5 ERRORS
1910  004564  001413                        BEQ     18$             ;IF MORE THAN 5
```

```
                                                        ;GO TO THE END OF THE PROGRAM

004566  005702          15$:  TST   R2                ;WAS SEEKING TO 0 OR 312?
004570  001402                BEQ   16$               ;TO 0, BRANCH
                                                      ;TO 312,
004572  005002                CLR   R2                ;SEEK NXT TIME TO 0
004574  000647                BR    1$                ;GO BAK & SK TO 0

004576  012702  014500    16$:  MOV   #14500,R2       ;SEEK NXT TIME TO 312

004602  005201                INC   R1                ;DONE SEEKS 200 TIMES?
004604  022701  000200          CMP   #200,R1

004610  001241                BNE   1$                ;IF NOT, GO BAK
004612  000404                BR    TST2              ;;EXIT

004614  104401  001640    18$:  TYPE  ,MSG4
004620  000137  015224          JMP   TST12

                        ;;*****************************************************************
                        ;*TEST 2        FORMAT THE DISK
                                ;*THIS PROGRAM ASSUMES AN UNFORMATTED DISK AND ITS
                                ;*FORMATTING IS DONE IN THIS TEST. A SECTOR IS FORMATTED
                                ;*AT A TIME.  THE FIRST OWRD OF EVERY SECTOR IS WRITTEN
                                ;*TO BE A PSUEDO-HEADER CONTAINING THE DRIVE #, CYLINDER
                                ;*#, SURFACE AND SECTOR #.  THE FOLLOWING IS CHECKED
                                ;*1.  'SIN' IF 'SIN' OCCURS, A DRIVE RESET IS DONE
                                ;*AND THE SAME SECTOR IS FORMATTED AGAIN.  THREE
                                ;*RETRIES ARE DONE BEFORE AN ERROR MESSAGE IS PRINTED.
                                ;*2.  'ERR' ON FINDING THAT THE 'ERR' BIT SET, RKER
                                ;*SCANNED TO FIND OUT WHAT CAUSED IT AND THE
                                ;*ERROR IS REPORTED.
                        ;;*****************************************************************
004624  000004          TST2: SCOPE
004626  013737  001230  002120      MOV   DRIVAD,DRHOLD   ;SAVE DRIVE NUMBER
004634  005737  002114            TST   FDRIVE          ;SEE IF EVEN RK-05F DRIVE
004640  001003                BNE   11$               ;YES
004642  005737  002116            TST   FDRVE1          ;ODD RK-05F?
004646  001125                BNE   TST3              ;DO NOT FORMAT IF ODD RK-05F
004650                  11$:
004650  012702  177152            MOV   #-626,R2        ;203 CYLINDERS, (406 TRAKS)
004654  012703  177764            MOV   #-14,R3         ;12 SECTORS
004660  012701  177773            MOV   #-5,R1          ;ALLOW ONLY 5 'SIN' ERRORS
004664  012705  177773            MOV   #-5,R5          ;ALLOW ONLY 5 'ERR'S
004670  013704  001230            MOV   DRIVAD,R4       ;STORE ADRES OF DRIVE.
004674  104415          4$:   CON.RESET
004676  104416                DRV.RESET               ;GO TO 'DR-RST' & DO DRIVE RESET
004700  005000          1$:   CLR   R0                ;KEEP COUNT OF 'SIN' ERORS
                                                      ;ALLOW 3 RETRIES ON SIN
004702  005777  174550            TST   @RKCS           ;ERR?
004706  100001                BPL   3$                ;NO

004710  104415                CON.RESET               ;GO TO 'CN-RST' & DO CONTROL RESET

004712  005046          3$:   CLR   -(SP)
```

```
1967  004714  012746  004722            MOV     #12$,-(SP)
1968  004720  000002                    RTI                     ;SET PRIORITY TO ZERO
1969  004722  010437  026446    12$:    MOV     R4,OUTBUF       ;WRITE THIS WORD
1970  004726  012777  026446  174526    MOV     #OUTBUF,@RKBA   ;FROM THIS ADRES
1971  004734  010477  174524            MOV     R4,@RKDA        ;ON THIS DISK SECTOR
1972  004740  012777  177777  174512    MOV     #-1,@RKWC       ;WRITE 1 WORD
1973  004746  012737  004674  001110    MOV     #4$,$LPERR      ;SET RETURN ADDRESS FOR
1974                                                            ;LUPING ON ERROR
1975
1976  004754  012777  002003  174474    MOV     #2003,@RKCS     ;GO WRT FMT
1977
1978  004762  104421            CON.RDY                         ;WAIT FOR CONTROL READY
1979  004764  032777  001000  174460  5$:     BIT     #1000,@RKDS     ;WAS THERE A SIN?
1980  004772  001413            BEQ     6$                      ;NO, SKIP DOING DRV RESET
1981  004774  004737  016136            JSR     PC,GTSRG        ;GO, GET RKCS, ER, DS, DA, CYLINDER
1982  005000  104007            ERROR   7                       ;SIN ERROR ON TRYING TO
1983                                                            ;WRT FMT ON CYLINDER AS
1984                                                            ;INDICATED IN RKDA.  3 RETRIES
1985                                                            ;ARE DONE
1986                                                            ;NOTE THAT BEFORE
1987  005002  104415            CON.RESET                       ;RETRYING A DRIVE RESET WAS DONE
1988  005004  104416            DRV.RESET                       ;GO TO 'DR-RST' & DO DRV RESET
1989  005006  005200            INC     R0                      ;INCRMNT SIN COUNT
1990  005010  022700  000003            CMP     #3,R0           ;ALLOW 3 RETRIES WERE THERE 3?
1991  005014  001332            BNE     1$+2                    ;IF NOT, GO & RETRY
1992
1993  005016  005201            INC     R1                      ;ALLOW ONLY 12 SIN ERRORS
1994                                                            ;IF MORE THAN 5 EXIT THIS TEST
1995  005020  001436            BEQ     9$                      ;IF MORE THAN 5 EXIT THIS TEST
1996  005022  005777  174430    6$:     TST     @RKCS           ;DID 'ERR' BIT SET IN RKCS?
1997  005026  100005            BPL     7$                      ;NO, BRANCH
1998  005030  004737  016136            JSR     PC,GTSRG        ;GO, GET RKCS, ER, DS,DA,CYL
1999  005034  104010            ERROR   10                      ;'ERR' OCCURED WHILE DOING
2000                                                            ;WRT FMT ON SECTOR, CYLINDER
2001                                                            ;AS INDICATED IN RKDA.
2002  005036  005205            INC     R5                      ;ALLOW ONLY 5 'ERR'S. IF
2003  005040  001426            BEQ     9$                      ;MORE THAN 5 EXIT THIS TEST
2004  005042  005204    7$:     INC     R4                      ;INCRMNT DISK ADRES TO NXT SCTR
2005  005044  005203            INC     R3                      ;ALL 12 SECTORS DONE?
2006  005046  001314            BNE     1$                      ;IF NOT, GO BAK & FMT NXT SCTR
2007                                                            ;IF YES
2008  005050  012703  177764            MOV     #-14,R3         ;RESET COUNT FOR 12 SECTORS
2009  005054  042704  000017            BIC     #17,R4          ;CLR OUT SEC BITS
2010
2011  005060  062704  000020    8$:     ADD     #20,R4          ;ADRES THE NXT TRAK TO B FMTED
2012  005064  005202            INC     R2                      ;ALL TRAKS FMTED?
2013  005066  001304            BNE     1$                      ;IF NOT GO BAK B FMT NXT CYL, SUR 0
2014  005070  005237  002114            INC     FDRIVE          ;EVEN RK05F?
2015  005074  001004            BNE     10$                     ;NO
2016  005076  062737  020000  001230    ADD     #20000,DRIVAD   ;FORMAT ODD DRIVE OF F
2017  005104  000661            BR      11$
2018  005106  013737  002120  001230  10$:    MOV     DRHOLD,DRIVAD   ;RESTORE DRIVE ADDR
2019  005114  000402            BR      TST3                    ;;EXIT
2020
2021  005116  004737  016772    9$:     JSR     PC,ABRT
2022
```

```
                                ;;*****************************************************************
                                ;*TEST 3         READ FORMAT OF THE DISK
                                        ;* IN THIS TEST, THE HEADERS FROM ALL THE SECTORS ARE READ
                                        ;* & CHECKED IF THEY ARE CORRECT.  THE FOLLOWING IS THE
                                        ;* TEST SEQUENCE.
                                        ;* 1.  READ 12 SECTORS (HDRS ONLY) AT A TIME
                                        ;* 2.  IF THERE IS A 'SIN' ERROR RETRY ONCE MORE, IF SAW AGAIN
                                        ;* REPORT ERROR & READ HEADER FROM NEXT CYLINDER
                                        ;* 3.  IF THERE IS 'ERR' IN RKCS, DO A CONTROL RESET, REPORT
                                        ;* ERROR & READ HEADER FROM NEXT CYLINDER.  IF THERE ARE
                                        ;* MORE THAN 5 ERRORS OF THIS KIND, THIS TEST WILL BE EXITED
                                        ;* 4.  THE 12 HEADERS ARE CHECKED.  IF THEY ARE CORRECT THE
                                        ;* NEXT CYLINDER IS READ.
                                        ;* IF THEY ARE NOT CORRECT, A RETRY IS DONE; IF AGAIN CORRECT
                                        ;* HEADERS ARE NOT RECIEVED, AN ERROR IS REPORTED.  THE
                                        ;* SECTOR #'S GIVING THE BAD HEADERS, & THE BAD HEADERS ARE
                                        ;* STORED.
                                        ;* 5.  IF INHIBIT TYPEOUT' SWITCH IS NOT SET, THE FIRST WORDS OF
                                        ;* THE 12 SECTORS (PSUEDO-HEADERS) ARE READ.  IN A PREVIOUS
                                        ;* TEST THE FIRST WORD OF EVERY SECTOR WAS WRITTEN
                                        ;* AS A SOFTWARE HEADER (CONSISTING OF DRIVE #, CYL#, SUR,SEC#,
                                        ;* THEN THE SECTOR # GIVING BAD HEADER, EXPECTED PSUEDO-HEADER,
                                        ;* & THE PS-HEADER RECIEVED ARE TYPED OUT.  THIS WOULD
                                        ;* WRONG, HEADER WAS READ WRONG, ETC.
                                        ;* 6.  THE NEXT CYLINDER IN LINE IS READ.  ORDER OF READING IS
                                        ;* CYL0,SUR0    CYL0,SUR1       CYL312,SUR1
                                ;;*****************************************************************
005122  000004                TST3:   SCOPE

005124  012737  177773  001504         MOV     #-5,ERCNT1      ;ALLOW ONLY 5 ERRORS (OF BAD HEADER
                                                                ;KIND FROM 5 CYLINDERS)
005132  012737  177766  001506         MOV     #-12,ERCNT2     ;ALLOW ONLY 12 'ERR'S
005140  012737  177773  001510         MOV     #-5,ERCNT3      ;ALLOW ONLY 5 ERRORS
005146  013705  001230                 MOV     DRIVAD,R5       ;SET DRIVE #,CYL ADRES=0
005152  012737  177152  001476         MOV     #-626,INDX2     ;313 CYLS (626 TRAKS) TO B READ
005160  104415              4$:       CON.RESET               ;GO DO CONTROL RESET
005162  104416                        DRV.RESET               ;GO DO DRIVE RESET

005164  005037  001254      1$:       CLR     RETRY2          ;ALLOW 2 RETRIES IF HDRS READ WRONG
005170  005037  001252      2$:       CLR     RETRY1          ;ALLOW 2 RETRIES FOR 'SINS'

005174  012777  026446  174260  3$:   MOV     #OUTBUF,@RKBA   ;RD HDRS INTO LOC STARTING AT THIS
005202  010577  174256                MOV     R5,@RKDA        ;FROM THIS DSK ADRES
005206  012777  177764  174244        MOV     #-14,@RKWC      ;12 HDRS TO BE READ
005214  012737  005160  001110        MOV     #4$,$LPERR      ;SET RETURN ADRES FOR LUPING ON ERROR

005222  012777  002005  174226        MOV     #2005,@RKCS     ;GO, RD FMT OF THIS CYLINDER

005230  104421                       CON.RDY                 ;WAIT FOR CNTRL RDY TO SET

005232  032777  001000  174212  5$:   BIT     #1000,@RKDS     ;'SIN' ERROR?
005240  001420                       BEQ     6$              ;NO, BRANCH
005242  004737  016214                JSR     PC,GETINF

005246  104011                       ERROR   11              ;'SIN' OCCURED WHEN DOING RD FMT
                                                             ;FROM CYL SHOWN IN RKDA.  IT
```

```
2079  005250  104415                        CON.RESET               ;DO CNTRL RESET
2080  005252  104416                        DRV.RESET               ;GO, DO DRIVE RESET
2081  005254  005237  001252                INC     RETRY1          ;ALLOW ONLY 2 RETRIES FOR THIS  ERROR
2082  005260  022737  000002  001252        CMP     #2,RETRY1       ;IF TRIED 2 TIMES REPORT
2083  005266  001342                        BNE     3$              ;ERROR, OTHERWISE GO BAK & RETRY
2084                                                                ;WAS TRIED TWICE, BUT 'SIN'.
2085  005270  005237  001510                INC     ERCNT3          ;ALLOW 5 ERRORS AT MOST
2086  005274  001002                        BNE     6$
2087  005276  000137  005606                JMP     16$
2088
2089  005302  005777  174150        6$:     TST     @RKCS           ;'ERR' IN RKCS?
2090  005306  100010                        BPL     7$              ;NO, BRANCH
2091  005310  004737  016136                JSR     PC,GTSRG        ;GO, GET RKCS, ER,DS,DA,CYLNDR
2092  005314  104012                        ERROR   12              ;'ERR' SET WHILE DOING RD FMT
2093                                                                ;FROM CYL SHOWN IN RKDA
2094  005316  104415                        CON.RESET               ;GO DO CNTRL RESET
2095                                                                ;ALLOW ONLY 12 ERRORS OF THIS
2096  005320  005237  001506                INC     ERCNT2          ;KIND, IF MORE THAN FIVE ERRORS
2097                                                                ;SKIP THIS TEST
2098  005324  001532                        BEQ     TST4            ;EXIT
2099  005326  000520                        BR      14$             ;GO SET UP TO RD FMT FROM NXT
2100                                                                ;CYL IN LINE
2101                                                                ;CHECK THAT CORRECT HEADERS WERE RECVD.
2102                                                                ;SECTR # HAVING BAD HDR IS STORED ALONG
2103                                                                ;WITH BAD HDR
2104
2105  005330  004737  007260        7$:     JSR     PC,CHKHDRS      ;GO CHECK IF CORRECT HEADERS WERE READ
2106
2107  005334  005737  001500                TST     INDX3           ;WAS THERE A MISCOMPARISON?
2108  005340  001513                        BEQ     14$             ;IF NOT, GO SET UP TO RD FMT
2109                                                                ;NXT CYL IN LINE
2110  005342  012737  005170  001110        MOV     #2$,$LPERR
2111  005350  104013                        ERROR   13              ;CORRECT HDRS WERE NOT RECVD
2112                                                                ;FROM SECTRS AS TYPED OUT.
2113                                                                ;THE SAME CYLINDER WAS READ TWICE
2114  005352  005237  001254                INC     RETRY2          ;RETRY RD FMT ON SAME CYL AGAIN
2115  005356  022737  000002  001254        CMP     #2,RETRY2       ;TRIED RDING SAME CYL TWICE
2116  005364  001301                        BNE     2$              ;IF NOT, GO RD AGAIN
2117                                                                ;YES, REPORT ERROR
2118  005366  005237  001504                INC     ERCNT1          ;ALLOW ONLY 5 ERRORS OF THE
2119                                                                ;ABOVE TYPE.  IF MORE THAN 12
2120                                                                ;EXIT THIS TEST
2121  005372  001505                        BEQ     16$
2122
2123  005374                        20$:                            ;THE PSUEDO-HEADERS (FIRST WORD OF EVERY
2124                                                                ;SECTOR) FROM THIS CYLINDER (ABOVE,
2125                                                                ;THE CYLINDER THAT GAVE WRONG HEADERS)
2126                                                                ;WILL BE READ, NOW.  FOLLOWING WILL B TYPD OUT:
2127                                                                ;SEC#, EXPCTD PSUEDO-HDR, RECVD PHDR.
2128                                                                ;IF "INHIBIT TYPEOUT" SW IS SET THIS ENTIRE
2129                                                                ;READING & TYPING WILL BE SKIPPED
2130  005374  032777  020000  173536        BIT     #20000,@SWR     ;INHIBIT TYPEOUT?
2131  005402  001072                        BNE     14$             ;YES, SKIP THE FOLLOWING & GO
2132                                                                ;SET UP TO RD FMT NXT CYL IN LINE
2133
2134  005404  012701  177764                MOV     #-14,R1         ;READ FROM 12 SECTRS
```

```
                                        MOV     R5,@RKDA        ;FROM THIS DSK-ADRES
                                        MOV     #OUTBUF,@RKBA   ;INTO THIS BUS-ADRES
                                10$:    MOV     #-1,@RKWC       ;RD 1 WRD

                                        MOV     #5,@RKCS        ;GO, RD
                                        CON.RDY                 ;WAIT FOR CNTRL RDY
                                        TST     @RKCS           ;ANY ERR?
                                        BPL     15$             ;NO, PROCEED
                                        CON.RESET               ;CLEAR THE ERR
                                        BR      14$             ;ERR, SO COULDN'T READ PSUEDO-HDRS

                                15$:    INC     R1              ;READ FROM ALL 12 SECS
                                        BNE     10$             ;IF NOT GO RD THE NXT ONE

                                                                ;TYPE OUT  SUEDO-HDRS CORRESPONDING TO
                                                                ;THE SECTORS WHICH GAVE BAD HEADERS
                                                                ;TYPE: SEC #, EXPC P-HDR, RECVD P-HDR

                                        TYPE                    ;TYPE OUT
                                        MSG11
                                        MOV     #BUFR,R1        ;SEC #'S ARE STORED HERE

                                11$:    TYPE                    ;TYPE CR, LF
                                        $CRLF

                                        MOV     (R1),R2
                                        MOV     #OUTBUF,R3      ;PSUEDO-HEADERS WHICH WERE
                                                                ;READ ARE STORED HERE

                                12$:    TST     R2              ;IS THIS SEC # CORRESPONDING TO THE
                                        BEQ     13$             ;ONE IN ERROR
                                        DEC     R2              ;R2 CONTAINS THE SEC #
                                        TST     (R3)+
                                        BR      12$

                                13$:    MOV     (R1),-(SP)      ;GO TYPEOUT SEC # GIVING
                                        TYPOS                   ;MISCOMPARISON OF HEADERS
                                        .BYTE   2
                                        .BYTE   0               ;SUPRES LDG 0'S

                                        TYPE                    ;TYPE 2 BLANKS
                                        BLNKS5

                                        MOV     R5,-(SP)        ;GO TYPE EXPCTD PSUEDO HEADER
                                        BIS     (R1),(SP)
                                        TYPOC

                                        TYPE                    ;TYPE 2 BLNKS
                                        BLNKS7

                                        MOV     (R3),-(SP)      ;GO TYPE PSUEDO-HEADER RECVD
                                        TYPOC

                                        TST     (R1)+           ;TYPED OUT ALL SEC #'S IN ERROR.
                                        CMP     (R1),#177777
                                        BNE     11$             ;IF NOT GO BAK & TYPE NXT
```

```
2191
2192  005552  104401  001733          TYPE    ,MSG7           ;TYPE OUT PC
2193  005556  012746  005374          MOV     #20$ -(SP)
2194  005562  104402                  TYPOC
2195  005564  104401  001213          TYPE    ,$CRLF
2196                                                          ;TYPE ROUTINE ENDS HERE
2197
2198                                                          ;FIND OUT NXT TRAK TO B READ
2199                                                          ;FORMATTED
2200
2201  005570  062705  000020  14$:    ADD     #20,R5          ;SET ADRES FOR SUR 0, NXT CYL IN LINE
2202  005574  005237  001476          INC     INDX2           ;READ ALL 313 CYLINDERS (626 TRAKS)?
2203  005600  001404                  BEQ     TST4            ;;EXIT
2204  005602  000137  005164          JMP     1$              ;IF NOT, GO BAK & READ NXT
2205
2206  005606  004737  016772  16$:    JSR     PC,ABRT
2207
2208                                  ;;*****************************************************************
2209                                  ;*TEST 4         SEEK PATTERNS: 0-312-0-311-...,USING IMPLIED SEEK
2210
2211                                          ;****TEST 2 (WRITING PSUEDO-HEADERS) SHOULD HAVE BEEN DONE BEFORE****
2212                                          ;**** DOING THIS TEST****
2213                                          ;*THIS TEST PERFORMS SEEKS (IMPLIED SEEKS USING 'READS') IN THE
2214                                          ;*FOLLOWING PATTERN.
2215                                          ;*0-312-0-311-0-310-.....0-1-0-0
2216
2217                                          ;*THE FIRST WORD OF EVERY SECTOR IS A PSEUDO-HEADER (WRITTEN IN
2218                                          ;*A PREVIOUS TEST) CONSISTING OF DRIVE NO, CYLINDER NO., SURFACE
2219                                          ;*AND SECTOR NO.  AN IMPLIED SEEK IS DONE BY ISSUING A 'READ' FOR
2220                                          ;*THE PSEUDO-HEADER OF SECTOR 0, SURFACE 0.
2221
2222                                          ;*IF A 'SIN' OCCURS TWO TRIES ARE DONE BEFORE ABORTING.  IF A 'SKE'
2223                                          ;*OCCURS IT COULD MEAN THAT 1) EITHER THE HEADERS WAS READ WRONG
2224                                          ;*OR 2) THE HEADS GOT POSITIONED ON THE WRONG CYLINDER.  IN
2225                                          ;*ORDER TO PROVIDE A FURTHER INSIGHT INTO THE PROBLEM, THE FOLLOWING
2226                                          ;*IS DONE:
2227                                          ;*THE HEADERS ARE READ FROM THE CYLINDER THAT GAVE 'SKE'.  IF THE HEADERS
2228                                          ;*ARE CORRECT IT IS SO REPORTED.  IF THE HEADERS ARE INCORECT, THEN THE
2229                                          ;*EXPECTED HEADERS AND THE RECEIVED ONES ARE REPORTED.  ONE MORE TRY IS
2230                                          ;*DONE (THE IMPLIED SEEK IS TRIED AGAIN BETWEEN THE CYLINDERS THAT GAVE RISE
2231                                          ;*TO 'SKE')
2232
2233                                          ;*THE FOLLOWING ACTION IS TAKEN WHEN THERE IS NO 'SKE' OR 'SIN' BUT STILL THE
2234                                          ;*PSEUDO-HEADER IS READ WRONG:
2235                                          ;*FIRST THE HEADERS ARE READ FROM THAT CYLINDER AND CHECKED.  IF THEY ARE
2236                                          ;**CORRECT, IT IS SO REPORTED.  IF THEY ARE WRONG THEN THE EXPECTED AND RECEIVED
2237                                          ;*HEADERS ARE REPORTED.  THEN THE PSEUDO-HEADER FROM SECTOR 1 IS READ AND REPORT
2238                                          ;*ONE MORE TRY IS DONE BY REPEATING THE WHOLE PROCESS. (IMPLIED SEEK
2239                                          ;*BETWEEN THE TWO CYLINDERS AND READING PSEUDO-HEADER FROM SECTOR 0 OF THE DEST
2240                                          ;*CYLINDER).
2241
2242                                          ;*UP TO 12 ERRORS OF EACH KIND (SIN, SKE, BAD PSEUDO-HEADER) ARE ALLOWED.
2243                                          ;*IF ANY ERROR OCCURS MORE THAN 12 TIMES THE TEST IS ABORTED.
2244                                  ;;*****************************************************************
2245  005612  000004  TST4:   SCOPE
2246  005614  104415          CON.RESET               ;GO DO CONTROL RESET
```

```
2247  005616  104416                        DRV.RESET               ;GO DO DRIVE RESET
2248
2249  005620  005004                        CLR     R4              ;FLAG, CLR IF DOING IMPLIED
2250                                                                  ;SEEK IN FROM 0 TO 'INADR'
2251                                                                  ;=1, IF GOING FROM 'INADR'
2252                                                                  ;OUT TO CYL 0
2253  005622  012737  177465  001476          MOV     #-313,INDX2     ;313 SEEK PATTERNS
2254  005630  012700  177764                  MOV     #-14,R0
2255  005634  010037  001504                  MOV     R0,ERCNT1       ;ALLOW ONLY 12 ERRORS
2256  005640  010037  001506                  MOV     R0,ERCNT2       ;OF THESE KINDS
2257  005644  010037  001510                  MOV     R0,ERCNT3
2258
2259  005650  012737  014500  001260          MOV     #14500,INADR    ;'INADR' CONTAINS THE INER
2260                                                                  ;CYL TO WHICH IMPLIED SEEK WILL
2261                                                                  ;BE DONE
2262
2263  005656  005704                  1$:     TST     R4              ;GOING IN OR OUT?
2264  005660  001005                          BNE     2$              ;GOING OUT, BRANCH
2265  005662  013705  001260                  MOV     INADR,R5        ;SET CYL ADRES BITS FOR GOING IN
2266  005666  053705  001230                  BIS     DRIVAD,R5       ;FORM DISK ADRES FOR INNER
2267  005672  000402                          BR      3$              ;CYLINDER
2268  005674  013705  001230          2$:     MOV     DRIVAD,R5       ;FORM DISK ADRES FOR OUTER
2269                                                                  ;CYLINDER - 0
2270                                                                  ;ALLOW 2 TRIES WHEN
2271  005700  012737  177776  001254  3$:     MOV     #-2,RETRY2      ;ERRORS OCCUR
2272  005706  012737  177776  001252  13$:    MOV     #-2,RETRY1
2273  005714  012737  177777  001256  4$:     MOV     #-1,RETRY3
2274  005722  000404                          BR      5$
2275  005724  104415                  6$:     CON.RESET
2276  005726  104416                          DRV.RESET
2277  005730  004737  006526                  JSR     PC,SBR1         ;REPOSITION HEADS TO PRE-ERROR CYL
2278
2279  005734  012777  177777  173516  5$:     MOV     #-1,@RKWC       ;READ 1 WORD
2280  005742  010577  173516                  MOV     R5,@RKDA        ;FROM THIS CYLINDER, SEC 0
2281  005746  012777  026446  173506          MOV     #OUTBUF,@RKBA   ;INTO THIS BUS ADRES
2282  005754  012737  005724  001110          MOV     #6$,$LPERR      ;SET RETURN ADRES FOR LUPING
2283                                                                  ;ON 'ERROR'
2284
2285  005762  012777  000005  173466          MOV     #5,@RKCS        ;GO, READ
2286
2287  005770  104421                          CON.RDY                 ;WAIT FOR CNTRL RDY
2288
2289  005772  032777  001000  173452          BIT     #1000,@RKDS     ;SIN?
2290  006000  001434                          BEQ     8$              ;NO, BRANCH
2291                                                                  ;YES, THERE WAS A SIN
2292  006002  004737  016376                  JSR     PC,ERR1         ;GO GET, CYLS BETW'N WHICH SK WAS TRIED
2293  006006  017737  173440  001170          MOV     @RKDS,$REG3
2294  006014  017737  173434  001166          MOV     @RKER,$REG2
2295  006022  104420  001602                  TYPMSG  ,MSG1
2296  006026  013737  001256  001172          MOV     RETRY3,$REG4    ;SAVE TRY # ON 'SIN'
2297  006034  062737  000002  001172          ADD     #2,$REG4
2298  006042  104014                          ERROR   14              ;AN IMPLIED SEEK WAS TRIED
2299                                                                  ;FROM 'CYLA' TO 'CYLB' (INDICATED
2300                                                                  ;IN EROR MESAGE), 'SIN' OCCURRED.
2301                                                                  ;2 TRIES ARE DONE BEFORE
2302                                                                  ;ABORTING
```

```
2303  006044  005737  001256          TST     RETRY3          ;DONE RETRIES
2304  006050  001403                  BEQ     7$              ;YES, BRANCH
2305  006052  005237  001256          INC     RETRY3          ;GO DO 2ND TRY
2306  006056  000722                  BR      6$
2307
2308  006060  005237  001504  7$:     INC     ERCNT1          ;ALLOW LESS THAN 12 ERORS OF THIS TYPE
2309  006064  001103                  BNE     19$             ;IF MORE SKIP THIS TEST
2310  006066  000137  006614          JMP     EXT4            ;EXIT THIS TEST
2311
2312  006072  032777  010000  173354  8$:     BIT     #10000,@RKER    ;SKE?
2313  006100  001506                  BEQ     20$
2314  006102  004737  016376  000166  15$:    JSR     PC,ERR1         ;GO GET 2 CYL NOS. BETWEEN WHICH
2315  006106  017737  173342  001166          MOV     @RKER,$REG2     ;IMPLIED SEEK WAS DONE
2316  006114  017737  173332  001170          MOV     @RKDS,$REG3
2317  006122  013737  001252  001172          MOV     RETRY1,$REG4    ;GET TRY # ON 'SKE'
2318  006130  062737  000003  001172          ADD     #3,$REG4
2319  006136  104420  001610                  TYPMSG  .MSG2           ;GO PRINT 'SKE'
2320  006142  104014                          ERROR   14              ;IMPLIED SEEK WAS TRIED FROM
2321                                                                  ;'CYLA' TO 'CYLB' (INDICATED
2322                                                                  ;IN EROR MESAGE); 'SKE' OCCURRED.
2323                                                                  ;2 TRIES ARE DONE.
2324  006144  104415                          CON.RESET               ;DO CONTROL RESET
2325
2326  006146  004737  006552  9$:     JSR     PC,SBR2         ;GO READ 12 HEADERS FROM
2327                                                          ;THIS CYLINDER & COMPARE
2328                                                          ;THEM.  NOTE R5 CONTAINS THE
2329                                                          ;DISK ADRES THAT WILL BE USED.
2330
2331  006152  012777  000015  173276          MOV     #15,@RKCS       ;GO DO DRIVE RESET
2332                                                                  ;WHILE THE DRIVE IS DOING RESET
2333                                                                  ;THE HDRS THAT WERE READ
2334                                                                  ;ABOVE ARE CHECKED, PRINTED
2335
2336  006160  005737  001500          TST     INDX3           ;WAS THERE A MISCOMPARISON
2337                                                          ;IN ANY HEADER?
2338  006164  001006                  BNE     10$             ;IF INDX3>0, THERE WAS.
2339                                                          ;NO, THERE WASN'T. HDRS OK
2340  006166  005237  001252          INC     RETRY1          ;ONLY 2 TRIES FOR SKE
2341  006172  001414                  BEQ     12$             ;BRANCH IF THIS WAS A 2ND TRY
2342  006174  104420  001657          TYPMSG  .MSG5           ;TYPE OUT THAT HDRS WERE READ
2343                                                          ;CORRECTLY.  THIS WAS TRY # 1
2344
2345  006200  000405                  BR      11$
2346
2347                                                          ;HDRS WERE READ INCORRECT.
2348  006202  005237  001252  10$:     INC     RETRY1          ;ALLOW 2 TRIES FOR SKE
2349  006206  001411                  BEQ     14$             ;BRANCH, IF THIS WAS 2ND TRY
2350
2351                                                          ;THERE WAS SKE ON DOING IMPLIED
2352  006210  104417  000015          MESAGE  .15             ;SEEK TO 'CYL B'. THEN HDRS WERE
2353                                                          ;READ FROM CYL B, WRONG HDRS
2354                                                          ;RECVD
2355
2356  006214  104423          11$:    RESDON                  ;WAIT FOR PREVIOUS DRIVE RESET
2357                                                          ;TO BE DONE
2358  006216  004737  006526          JSR     PC,SBR1         ;GO, REPOSITION HEADS
```

```
2359  006222  000634                        BR      4$
2360
2361                                                          ;2ND TRY, SKE THIS TIME ALSO. BU
2362                                                          ;READ HDRS CORRECTLY FROM
2363                                                          ;CYLINDER THAT GAVE SKE
2364                                                          ;NOTE THIS WAS THE 2ND TRY
2365  006224  104420  001657          12$:    TYPMSG  ,MSG5   ;TYPE OUT THAT HDRS WERE
2366                                                          ;READ CORRECTLY.
2367
2368  006230  000402                        BR      16$
2369
2370                                                          ;2ND TRY, SKE THIS TIME ALSO.
2371  006232  104417  000015          14$:    MESAGE  .15     ;READ HDRS FROM CYL THAT
2372                                                          ;GAVE SKE, THEY WERE INCORRECT.
2373
2374  006236  104423                  16$:    RESDON          ;WAIT FOR PREVIOUS DRIVE RESET
2375                                                          ;TO BE DONE
2376
2377  006240  005237  001506                INC     ERCNT2  ;ALLOW ONLY LESS THAN 10 ERRORS OF
2378                                                          ;THIS TYPE (SKE)
2379  006244  001002                        BNE     17$
2380  006246  000137  006614                JMP     EXT4    ;EXIT THIS TEST IF MORE
2381
2382  006252  005704                  17$:    TST     R4      ;WENT LAST TIME IN OR OUT?
2383  006254  001007                        BNE     19$     ;OUT
2384                                                          ;IN
2385  006256  005703                        TST     R3      ;WERE HDRS CORRECT?
2386  006260  001005                        BNE     19$     ;NO
2387                                                          ;YES
2388
2389  006262  005204                  18$:    INC     R4
2390  006264  004737  006526                JSR     PC,SBR1 ;GO POSITION HEADS BAK ON INNER
2391                                                          ;CYL
2392  006270  000137  005656                JMP     1$      ;GO BAK & SEEK OUT NOW
2393
2394  006274  005237  001476          19$:    INC     INDX2   ;ALL SEEK PATTERNS DONE?
2395  006300  001547                        BEQ     TSTS    ;;EXIT
2396
2397  006302  162737  000040  001260        SUB     #40,INADR       ;SET ADDRESS FOR THE NXT
2398                                                          ;INNER CYLINDER
2399  006310  005004                        CLR     R4      ;INDICATE THAT NOW SEEK IS GOING
2400                                                          ;TO BE IN
2401  006312  000137  005656                JMP     1$      ;GO BAK & SEEK IN TO 'INADR'
2402
2403  006316                           20$:                   ;IF THERE WAS NO SIN OR SKE
2404                                                          ;ENTER HERE
2405
2406  006316  012737  005700  001110        MOV     #3$,$LPERR      ;SET RETURN ADRES FOR LUPING
2407                                                          ;ON ERROR
2408  006324  020537  026446                CMP     R5,OUTBUF       ;CORRECT PSUEDO-HEADER READ?
2409  006330  001471                        BEQ     24$     ;YES, BRANCH
2410  006332  013737  001254  001172        MOV     RETRY2,$REG4    ;GET TRY #
2411  006340  062737  000003  001172        ADD     #3,$REG4
2412  006346  004737  016376                JSR     PC,ERR1 ;GO GET CYL #'S BETW'N
2413                                                          ;WHICH IMPLIED SEEK (READ)
2414                                                          ;WAS DONE
```

```
2415  006352  010537  001166          MOV     R5,$REG2        ;GET EXPCTD PSUEDO-HDR
2416  006356  013737  026446  001170  MOV     OUTBUF,$REG3    ;GET PSUEDO-HDR RECVD
2417  006364  104016                  ERROR   16              ;IMPLIED SEEK FROM CYLA TO CYLB WAS DONE.
2418                                                          ;READ PSEUDO-HEADER OF SEC 0,
2419                                                          ;CYLB (IN EROR MESAGE), BUT
2420                                                          ;THE WRONG PSEUDO-HEADER WAS
2421                                                          ;RECEIVED
2422  006366  005237  001254          INC     RETRY2
2423  006372  001402                  BEQ     21$
2424  006374  000137  005706          JMP     13$
2425
2426
2427  006400  004737  006552  21$:    JSR     PC,SBR2         ;GO READ HEADERS (12) FROM
2428                                                          ;THIS CYLINDER, & CHECK THEM.
2429                                                          ;IF MISCOMPARISON INDX3 WILL
2430                                                          ;BE > 0.
2431  006404  005737  001500          TST     INDX3
2432  006410  001003                  BNE     22$
2433  006412  104420  001657          TYPMSG  .MSG5           ;WRONG PSUEDO-HDR WAS READ
2434                                                          ;BUT WHEN HDRS WERE READ
2435                                                          ;FROM THE SAME CYLINDER, THEY
2436  006416  000402                  BR      23$             ;WERE CORRECT
2437
2438  006420  104417  000015  22$:    MESAGE  ,15             ;WRONG PSUEDO-HDR WAS READ
2439                                                          ;FROM 'CYLB' (IN ERROR MESAGE).
2440                                                          ;THEN HEADERS WERE READ FROM THE
2441                                                          ;SAME CYLINDER.  THEY WERE ALSO
2442                                                          ;WRONG.
2443  006424  010500          23$:    MOV     R5,R0           ;NOW READ THE PSUEDO-HEADER
2444  006426  005200                  INC     R0              ;FROM THE NEXT SECTOR (1)
2445  006430  010077  173030          MOV     R0,@RKDA        ;SAME CYLINDER
2446  006434  012777  026446  173020  MOV     #OUTBUF,@RKBA
2447  006442  012777  177777  173010  MOV     #-1,@RKWC
2448  006450  012777  000005  173000  MOV     #5,@RKCS
2449  006456  104421                  CON.RDY
2450  006460  010537  001162          MOV     R5,$REG0
2451  006464  004737  016434          JSR     PC,GCYL         ;GO GET CYL # & STORE IT IN $REG0
2452  006470  010037  001164          MOV     R0,$REG1        ;GET EXPCT PSUEDO-HDR FROM SEC 1
2453  006474  013737  026446  001166  MOV     OUTBUF,$REG2
2454  006502  104417  000017          MESAGE  ,17             ;PSUEDO-HEADER FROM SEC 1, CYLB
2455                                                          ;(IN MESAGE) WAS READ.  THE EXPCTD
2456                                                          ;& RECVD DATA WORDS ARE REPORTED.
2457  006506  005237  001510          INC     ERCNT3          ;ALLOW ONLY LESS THAN 10 ERRORS
2458                                                          ;OF THIS TYPE (WRONG PS-HDRS)
2459  006512  001440                  BEQ     EXT4
2460
2461  006514  005704          24$:    TST     R4              ;SEEKED IN OR OUT LAST TIME?
2462  006516  001266                  BNE     19$             ;IF OUT, GO SEEK NXT INNER CYL
2463                                                          ;IF IN, GO SEEK BAK TO 0
2464  006520  005204                  INC     R4              ;INDICATE THAT SEEK OUT (0)
2465  006522  000137  005656          JMP     1$              ;WILL BE DONE NOW
2466
2467
2468                                  ;THIS ROUTINE IS USED IN THIS TEST ONLY.
2469                                  ;R4=0 INDICATES SEEK BEING DONE FROM
2470                                  ;CYL 0 TO INNER CYL.
```

```
2471                                                    ;R4=1 INDICATES SEEK BEING DONE FROM
2472                                                    ;INNER CYL TO 0. THIS ROUTINE POSITIONS
2473                                                    ;THE HEADS ON 'INADR' CYL IF R4=1
2474
2475  006526  005704                SBR1:   TST     R4
2476  006530  001407                        BEQ     1$
2477  006532  013777  001260  172724      MOV     INADR,@RKDA
2478  006540  012777  000011  172710      MOV     #11.,@RKCS
2479  006546  104422                        ISI.RWS
2480  006550  000207                1$:     RTS     PC
2481
2482                                                    ;THIS ROUTINE IS USED IN THIS TEST
2483                                                    ;ONLY. IT READS 12 HEADERS FROM CYLINDER
2484                                                    ;WHOSE ADRES IS IN R5. THEN IT CHECKS
2485                                                    ;IF THE EXPECTED HEADER IS RECEIVED.
2486                                                    ;IF IT IS NOT, INDX3 IS INCREMENTED INDICATING
2487                                                    ;THE ERROR
2488
2489  006552  012700  177764        SBR2:   MOV     #-14,R0
2490  006556  012701  026446                MOV     #OUTBUF,R1
2491  006562  010077  172672                MOV     R0,@RKWC        ;READ 12 HDRS
2492  006566  010177  172670                MOV     R1,@RKBA        ;INTO THIS ADRES
2493  006572  010577  172666                MOV     R5,@RKDA        ;FROM THIS CYLINDER
2494  006576  012777  002005  172652        MOV     #2005,@RKCS     ;RD FMT, GO
2495  006604  104421                        CON.RDY
2496
2497  006606  004737  007260                JSR     PC,CHKHDRS      ;GO CHECK IF CORRECT HEADERS WERE READ
2498
2499  006612  000207                        RTS     PC              ;EXIT
2500
2501  006614  004737  016772        EXT4:   JSR     PC,ABRT
2502
2503                                ;;*****************************************************************
2504                                ;*TEST 5         PERFORM CONVERGING-DIVERGING (IMPLIED) SEEKS
2505                                        ;*THIS TEST PERFORMS A CONVERGING-DIVERGING SEEK PATTERN
2506                                        ;*USING IMPLIED SEEK (READ FORMAT).  THE SEEK SEQUENCE IS:
2507                                        ;*0-312-1-311-2-310-3-307-------310-2-311-1-312
2508                                        ;ALL READ FORMATS ARE DONE FROM SURFACE 0.
2509                                        ;*THE CYLINDER ADDRESSES, BETWEEN WHICH THE IMPLIED SEEK IS
2510                                        ;*PERFRORMED, ARE CONTAINED IN 'OUTADR' & 'INADR'.  IF 'SIN' OCCURS
2511                                        ;*AN ERROR IS REPORTED AND A RETRY IS DONE.  ON READING INCORRECT
2512                                        ;*HEADERS AN ERROR IS REPORTED AND A RETRY IS DONE.  NOTE THAT IF
2513                                        ;*ALL THE 12 HEADERS ARE INCORRECT, IT COULD MEAN THAT THE HEADS
2514                                        ;*COULD NOT POSITION CORRECTLY.  THIS WOULD BE CONFIRMED IF IN
2515                                        ;*PREVIOUS TESTS BAD HEADERS WERE NOT RECIEVED FROM THE SAME
2516                                        ;*CYLINDER.  IF THAT CYLINDER GAVE BAD HEADERS IN ALL THE PREVIOUS
2517                                        ;*TESTS THE PROBLEM COULD BE DIFFERENT.
2518                                        ;*MAXIMUM 12 ERRORS OF ANY KIND ARE ALLOWED.
2519                                        ;*IF MORE THAN 12 ERRORS OCCUR THE TEST IS ABORTED.
2520                                ;;*****************************************************************
2521  006620  000004                TST5:   SCOPE
2522  006622  104415                        CON.RESET               ;GO,DO CONTROL RESET
2523  006624  104416                        DRV.RESET               ;GO,DO DRIVE RESET
2524
2525  006626  005004                        CLR     R4              ;(R4)=0 SEEKING FROM 'OUTADR' TO 'INADR'
2526                                                                ;(R4)=1 SEEKING FROM 'INADR' TO 'OUTADR'
```

```
2527
2528  006630  012737  177466  001476          MOV     #-312,INDX2     ;SET COUNT FOR DOING 312 TIMES
2529  006636  012700  177764                  MOV     #-14,R0
2530  006642  010037  001504                  MOV     R0,ERCNT1       ;ALLOW ONLY 12 ERRORS
2531  006646  010037  001506                  MOV     R0,ERCNT2
2532
2533  006652  005037  001262                  CLR     OUTADR          ;INITIALIZE 'OUTADR' TO 0
2534  006656  012737  014500  001260          MOV     #14500,INADR    ;INITIALIZE 'INADR' TO 312
2535
2536  006664  005704                  1$:     TST     R4              ;GOING IN OR OUT?
2537  006666  001005                          BNE     2$              ;GOING OUT,BRANCH
2538  006670  013705  001260                  MOV     INADR,R5        ;SET CYL ADRES BITS FOR GOING IN
2539  006674  053705  001230                  BIS     DRIVAD,R5       ;FORM DISK ADRES FOR INNER CYLINDER
2540  006700  000404                          BR      3$
2541
2542  006702  013705  001262          2$:     MOV     OUTADR,R5       ;SET CYL ADRES BITS FOR GOING OUT
2543  006706  053705  001230                  BIS     DRIVAD,R5       ;FORM DISK ADRES FOR GOING OUT
2544
2545  006712  005037  001254          3$:     CLR     RETRY2          ;ALLOW 2 RETRIES
2546  006716  012737  177777  001252  4$:     MOV     #-1,RETRY1      ;WHEN ERRORS OCCUR
2547  006724  000404                          BR      7$
2548  006726  104415                  5$:     CON.RESET
2549  006730  104416                          DRV.RESET
2550  006732  004737  007224                  JSR     PC,SBR3         ;GO REPOSITION HEADS
2551
2552  006736  012777  177764  172514  7$:     MOV     #-14,@RKWC      ;READ ALL HDRS FROM THIS CYLINDER
2553  006744  010577  172514                  MOV     R5,@RKDA        ;FROM THIS CYL, SEC 0
2554  006750  012777  026446  172504          MOV     #OUTBUF,@RKBA   ;INTO THIS BUS ADRES
2555  006756  012737  006726  001110          MOV     #5$,$LPERR      ;SET RETURN ADRES FOR LOOPING ON ERROR
2556
2557  006764  012777  002005  172464          MOV     #2005,@RKCS     ;READ FORMAT,GO
2558
2559  006772  104421                          CON.RDY                 ;WAIT FOR CONTRL RDY
2560
2561  006774  032777  001000  172450          BIT     #1000,@RKDS     ;SIN?
2562  007002  001443                          BEQ     8$              ;NO, BRANCH
2563  007004  017737  172444  001166          MOV     @RKER,$REG2     ;SAVE RKER
2564  007012  017737  172434  001170          MOV     @RKDS,$REG3     ;SAVE RKDS
2565  007020  013737  001252  001172          MOV     RETRY1,$REG4    ;GET RETRY #
2566  007026  062737  000002  001172          ADD     #2,$REG4
2567  007034  004737  016320                  JSR     PC,ERR2         ;GET CYL #'S BELOW 'N WHICH
2568                                                                  ;SEEK WAS TRIED
2569  007040  104420  001602                  TYPMSG  ,MSG1           ;TYPE 'SIN'
2570  007044  104014                          ERROR   14
2571                                                                  ;'SIN' OCCURRED ON DOING IMPLIED
2572                                                                  ;SEEK FROM 'CYLA' TO 'CYLB' (IN
2573                                                                  ;EROR MESAGE).
2574  007046  005737  001252                  TST     RETRY1          ;DONE 2 TRIES?
2575  007052  001403                          BEQ     6$              ;YES, BRANCH
2576  007054  005237  001252                  INC     RETRY1          ;NO, RETRY
2577  007060  000722                          BR      5$
2578  007062  104415                  6$:     CON.RESET
2579  007064  104416                          DRV.RESET
2580
2581
2582  007066  005237  001504                  INC     ERCNT1          ;ALLOW LESS THAN 12 ERORS OF THE
```

```
2583  007072  001527                        BEQ     EXT5            ;ABOVE KIND
2584                                                                  ;IF MORE SKIP THIS TEST
2585                                                                  ;SIN OCCURED WHEN GOING TO CYL (IN
2586                                                                  ;R5).  A DRVE RESET HAS BEEN DONE.
2587                                                                  ;NOW TRY POSITIONING HEADS ON
2588                                                                  ;THAT CYL.
2589  007074  010577  172364                MOV     R5,@RKDA        ;SET CYL ADRES
2590  007100  012777  000011  172350        MOV     #11.,@RKCS      ;SEEK,GO
2591  007106  104422                        TST.RWS
2592  007110  000424                        BR      11$
2593                                                                  ;IF NO SIN, ENTER HERE
2594  007112  004737  007260        8$:     JSR     PC,CHKHDRS      ;GO CHECK IF CORRECT HEADERS WERE READ
2595
2596  007116  012737  006716  001110        MOV     #4$,$LPERR      ;SET LUP ADDRESS
2597  007124  005737  001500                TST     INDX3           ;WAS THERE A BAD HDR?
2598  007130  001414                        BEQ     11$             ;NO, BRANCH
2599  007132  104020                        ERROR   20              ;WRONG HEADERS WERE READ FROM
2600                                                                  ;'SEC #'S, ON DOING AN IMPLIED
2601                                                                  ;SEEK (RDFMT) FROM 'CYLA' TO 'CYLB'
2602  007134  005237  001254                INC     RETRY2          ;ALLOW 2 TRIES
2603  007140  022737  000002  001254        CMP     #2,RETRY2
2604  007146  001263                        BNE     4$              ;GO TRY 2ND TIME
2605  007150  005237  001506                INC     ERCNT2          ;ALLOW ONLY 12 ERRORS
2606  007154  001002                        BNE     11$
2607  007156  000137  007352                JMP     EXT5            ;IF MORE, EXIT THIS TEST
2608
2609  007162  005704                11$:    TST     R4              ;GOING WHICH WAY?
2610  007164  001006                        BNE     12$             ;'INADR' TO 'OUTADR', BRANCH
2611                                                                  ;'OUTADR' TO 'INADR'
2612  007166  005204                        INC     R4              ;INDICATE THAT NXT TIME GOING
2613                                                                  ;FROM 'INADR' TO 'OUTADR'
2614  007170  062737  000040  001262        ADD     #40,OUTADR      ;INCREMENT CYCLINDER ADRES
2615  007176  000137  006664                JMP     1$              ;GO BAK & DO IMPLIED SEEK
2616                                                                  ;FROM 'INADR' TO 'OUTADR'
2617
2618  007202  005004                12$:    CLR     R4              ;INDICATE THAT NXT TIME GOING
2619                                                                  ;FROM 'OUTADR' TO 'INADR'
2620  007204  162737  000040  001260        SUB     #40,INADR       ;DECREMENT CYLINDER ADRES
2621  007212  005237  001476                INC     INDX2           ;DONE ALL 312 FORWARD-BACKWARD
2622                                                                  ;SEEK PATTERNS
2623  007216  001457                        BEQ     TST6            ;;IF YES, EXIT
2624
2625  007220  000137  006664                JMP     1$              ;IF NOT, GO BAK & DO IMPLIED
2626                                                                  ;SEEK FROM 'OUTADR' TO 'INADR'
2627
2628                                ;THIS SUBROUTINE IS ENTERED AFTER A 'SIN' ERROR OCCURED ON GOING FROM
2629                                ;'CYLA' TO 'CYLB' AS INDICATED IN THE ERROR MESSAGE. BEFORE RETRYING THE
2630                                ;HEADS HAVE TO BE POSITIONED BACK TO 'CYLA'. NOTE THAT A DRIVE RESET
2631                                ;WAS DONE TO CLEAR SIN.
2632                                ;R4=0, INDICATES SEEK IS BEING DONE FROM 'OUTADR' CYLINDER TO 'INADR'CYLINDER.
2633                                ;R4=1, INDICATES THAT SEEK IS BEING DONE FROM 'INADR' TO 'OUTADR'.
2634  007224  005704                SBR3:   TST     R4              ;GOING WHICH WAY?
2635  007226  001404                        BEQ     1$              ;IF FROM 'OUTADR' TO 'INADR', BRANCH.
2636  007230  013777  001260  172226        MOV     INADR,@RKDA
2637  007236  000403                        BR      2$
2638  007240  013777  001262  172216  1$:   MOV     OUTADR,@RKDA
```

```
 2639  007246  012777  000011  172202  2$:     MOV     #11,@RKCS
 2640  007254  104422                          TST.RWS
 2641  007256  000207                          RTS     PC
 2642
 2643
 2644                                  ;CHKHDRS
 2645                                  ;THIS ROUTINE CHECKS THAT THE HEADERS READ PREVIOUSLY WERE CORRECT.
 2646                                  ;IF NOT, THE BAD HEADERS AND THE SECTOR #'S FROM WHICH THEY WERE READ
 2647                                  ;ARE STORED.
 2648                                  ;ON ENTRY:
 2649                                  ;R5 CONTAINS DISK ADDRESS FROM WHERE HEADERS WERE READ.
 2650                                  ;OUTBUF - 12 HEADRERS READ PREVIOUSLY ARE STORED STARTING 'OUTBUF'.
 2651                                  ;ON EXIT:
 2652                                  ;INDX3=0, IF THE HEADERS WERE CORRECT
 2653                                  ;INDX3=1, IF THE HEADERS WERE INCORRECT
 2654                                  ;BUFR - SECTOR #'S GIVING BAD HEADRERS ARE STORED STARTING AT 'BUFR'.
 2655                                  ;BUFR1 - BAD HEADERS FOR THE ABOVE SECTORS ARE STORED STARTING 'BUFR1'.
 2656                                  ;THE BAD SECTOR TABLE IS TERMINATED BY A '177777' WORD.
 2657
 2658  007260  005000                  CHKHDRS: CLR    R0              ;INITIALIZE FOR 14 HDRS
 2659  007262  012701  026446                  MOV     #OUTBUF,R1      ;INITIZLIZE PTR TO HDRS RECVD
 2660  007266  012702  001266                  MOV     #BUFR,R2        ;INITIALIZE PTR TO SECTOR TABLE
 2661  007272  012703  001320                  MOV     #BUFR1,R3       ;INITIALIZE PTR TO BAD HDR TABLE
 2662  007276  010537  001120                  MOV     R5,$GDADR
 2663  007302  042737  160037  001120          BIC     #160037,$GDADR  ;GET EXPCTD HEADER
 2664  007310  005037  001500                  CLR     INDX3           ;CLR FLG INDICATING BAD HDRS
 2665
 2666  007314  023711  001120          9$:     CMP     $GDADR,(R1)     ;HEADER OK?
 2667  007320  001406                          BEQ     10$             ;YES,BRANCH
 2668  007322  011123                          MOV     (R1),(R3)+      ;SAVE BAD HDR
 2669  007324  010022                          MOV     R0,(R2)+        ;SAVE BAD SECTR #
 2670
 2671  007326  012712  177777                  MOV     #177777,(R2)    ;PUT TERMINATR ON SECTR TABLE
 2672  007332  005237  001500                  INC     INDX3           ;SET FLG INDICATING BAD HDR
 2673  007336  005721                  10$:    TST     (R1)+           ;INCRMNT PTR TO NXT HDR
 2674  007340  005200                          INC     R0              ;ALL HDRS CHKD?
 2675  007342  022700  000014                  CMP     #14,R0
 2676  007346  001362                          BNE     9$              ;IF NOT, LUP BAK
 2677  007350  000207                          RTS     PC
 2678
 2679  007352  004737  016772          EXT5:   JSR     PC,ABRT
 2680
 2681                                  ;;*****************************************************************
 2682                                  ;*TEST 6         WRITE PATTERNS ON THE DISK
 2683                                          ;*IN THIS TEST DIFFERENT PATTERNS ARE WRITTEN ON THE ENTIRE DISK.  768
 2684                                          ;*WORDS (3 SECTORS) ARE WRITTEN AT A TIME.  TWO 768 WORDS LONG
 2685                                          ;*BUFFERS HAVE BEEN USED IN THIS TEST.  WHILE ONE BUFFER IS
 2686                                          ;*USED TO WRITE ON THE DISK, THE OTHER BUFFER GETS FILLED UP WITH
 2687                                          ;*PATTERNS TO BE USED NEXT!  THIS OVERLAPPING IS DONE TO SAVE TIME.
 2688
 2689                                          ;*THREE PATTERN-GENERATOR SUB-ROUTINES ARE USED:
 2690                                          ;*1.  PTGEN0    2.  PTGEN1      3.  PTGEN2      4.  PTGEN3
 2691                                          ;*THESE THREE ROUTINES ARE CALLED IN A CYCLIC ORDER:
 2692                                          ;*      PTGEN0-PTGEN1-PTGEN2-PTGEN3-PTGEN0-PTGEN1.....
 2693                                          ;*THE FOLLOWING ORDER IS MAINTAINED IN WRITING BLOCKS (EACH
 2694                                          ;*3 SECTORS LONG) USING ONE OF THE FOUR PATTERN GENERATORS:
```

```
2695
2696                                                    ;#CYL   SECTORS       CYL   SECTORS       CYL   SECTORS       CYL   SECTORS
2697                                                    ;#      SUR           SUR           SUR           SUR           ROUTI
2698                                                    ;# 0    0    0-2  PTGEN0  0    0    6-10  PTGEN1  0    0    3-5  PTGEN2  0    0  11-13 PTGEN3
2699                                                    ;# 0    1    0-2  PTGEN0  0    1    6-10  PTGEN1  0    1    3-5  PTGEN2  0    1  11-13 PTGEN3
2700                                                    ;# 1    0    0-2  PTGEN0  1    0    6-10  PTGEN1  1    0    3-5  PTGEN2  1    0  11-13 PTGEN3
2701                                                    ;# 1    1    0-2  PTGEN0  1    1    6-10  PTGEN1  1    1    3-5  PTGEN2  1    1  11-13 PTGEN3
2702                                                    ;# 2    0    0-2  PTGEN0  2    0    6-10  PTGEN1  2    0    3-5  PTGEN2  2    0  11-13 PTGEN3
2703                                                    ;#ETC, ETC.....
2704                                                    ;#THE ABOVE SHOWN STAGGERING (OF SECTORS TO BE WRITTEN) IS DONE TO
2705                                                    ;#SAVE ONE REV (40MS) EVERY TIME A BLOCK (3 SECTORS) IS WRITTEN.
2706                                                    ;#IF YOU WANT TO USE BUFFERS STARTING AT SOME OTHER MEMORY ADDRESS
2707                                                    ;#MAKE THE FOLLOWING CHANGES:
2708                                                    ;#CHANGE 'PBUF0' TO STARTING ADDRESS OF THE FIRST 768 WORDS LONG BUFFER.
2709                                                    ;#CHANGE 'PBUF1' TO STARTING ADDRESS OF SECOND 768 WORDS LONG BUFFER.
2710
2711                                                    ;#IF YOU WANT TO WRITE YOUR OWN PATTERNS USING PATTERN GENERATOR 'PTGEN0'
2712                                                    ;#CHANGE 'PTRN01' AND 'PTRN02' TO THE PATTERNS YOU WANT.
2713
2714                                                    ;#TO WRITE THE SAME TWO (OR ONE) PATTERNS ON THE ENTIRE DISK CHANGE
2715                                                    ;#LOCATION 'PAT1' TO 'PTGEN0' THE STARTING ADDRESS OF PAT-GENERATOR 0
2716                                                    ;#LOCATION 'PAT2' TO 'PTGEN0' THE STARTING ADDRES OF PAT-GENERATOR 0
2717                                                    ;#LOCATION 'PAT3' TO 'PTGEN0' THE STARTING ADDRESS OF PAT-GENERATOR 0
2718                                                ;;*****************************************************************
2719  007356  000004                            TST6: SCOPE
2720  007360  012737  177764  001504                  MOV     #-14,ERCNT1
2721  007366  012737  177764  001506                  MOV     #-14,ERCNT2
2722  007374  012737  177152  001474                  MOV     #-626.,INDX1        ;SET COUNT FOR 313X2 TRACKS
2723  007402  005037  001410                         CLR     BUFLG0              ;CLR FLAG FOR BUFR 0
2724  007406  005037  001412                         CLR     BUFLG1              ;CLR FLAG FOR BUFR 1
2725                                                                               ;BIT 7 OF ABOVE FLAGS WHEN SET
2726                                                                               ;INDICATES, THAT BUFR TO BE USED
2727                                                                               ;FOR WRITING ON DSK
2728  007412  013737  001230  001432                  MOV     DRIVAD,DSKADR       ;GET DRIVE #, DISK ADRES
2729  007420  012737  001414  001424                  MOV     #PAT0,PRSPAT        ;INITIALIZE PTR TO THE FIRST
2730                                                                               ;PATRN GENERATOR
2731  007426  004737  010044                         JSR     PC,GETBUF
2732  007432  017737  171766  001430                  MOV     @PRSPAT,PGSUBR
2733  007440  004777  171764                         JSR     PC,@PGSUBR          ;GO GENERATE PATRNS FOR
2734                                                                               ;3 SECTOR (400X3)
2735  007444  005005                         1$:     CLR     R5                  ;INITIALIZE COUNT FOR 4 BLOCKS
2736  007446  005737  001410                  2$:     TST     BUFLG0              ;FIND OUT WHICH BUFR TO USE
2737  007452  100407                                  BMI     3$                  ;FOR WRITING ON DSK
2738  007454  013737  001406  001434                  MOV     PBUF1,BUSADR        ;USE 'IOBUF1' FOR TRANSFER
2739                                                                               ;OR THE ONE INDICATED BY THE USER
2740  007462  052737  000200  001412                  BIS     #BIT7,BUFLG1        ;SET FLAG TO INDICATE THAT
2741                                                                               ;WRTING ON DSK WILL B DONE FROM
2742                                                                               ;THIS BUFR (BUF 1)
2743  007470  000406                                  BR      13$
2744
2745  007472  013737  001404  001434  3$:     MOV     PBUF0,BUSADR        ;USE 'IOBUF0' FOR TRANSFER
2746                                                                               ;OR THE ONE INDICATED BY THE USER
2747
2748  007500  052737  000200  001410                  BIS     #BIT7,BUFLG0        ;INDICATE THAT 'IOBUF0' WILL
2749                                                                               ;B USED FOR WRTING ON DISK
2750  007506  012737  007514  001110  13$:     MOV     #4$,$LPERR
```

```
2751  007514  013777  001432  171742  4$:     MOV     DSKADR,@RKDA    ;SET RKDA
2752  007522  013777  001434  171732          MOV     BUSADR,@RKBA
2753  007530  012777  176400  171722          MOV     #-1400,@RKWC
2754  007536  012777  000003  171712          MOV     #3,@RKCS        ;WRITE THE 4 SECTORS ON
2755                                                                  ;DISK
2756  007544  013737  001424  001426          MOV     PRSPAT,NXTPAT   ;WHILE THE PATRNS R BEING WRITTEN
2757  007552  023727  001424  001422          CMP     PRSPAT,#PAT3    ;GO GENERATE THE NXT PATRNS
2758  007560  001004                          BNE     5$              ;TO B WRITTEN
2759  007562  012737  001414  001426          MOV     #PAT0,NXTPAT    ;KEEP GENERATING PATRNS IN THIS
2760  007570  000403                          BR      6$              ;WAY "PAT0"-"PAT1"-"PAT2"-"PAT3"-"PAT0"-
2761  007572  062737  000002  001426  5$:     ADD     #2,NXTPAT
2762  007600  004737  010044          6$:     JSR     PC,GETBUF
2763  007604  017737  171616  001430          MOV     @NXTPAT,PGSUBR
2764  007612  004777  171612                  JSR     PC,@PGSUBR      ;GO GENERATE THESE PATRNS.
2765                                                                  ;(3 X 400) WORDS
2766  007616  104421                  CON.RDY
2767  007620  032777  140000  171630          BIT     #140000,@RKCS   ;ANY ERROR?
2768  007626  001411                          BEQ     12$             ;GET RKCS,ER,DS,DA
2769  007630  004737  016162                  JSR     PC,GT4RG
2770  007634  104021                          ERROR   21              ;ERROR ON DOING WRITE
2771  007636  104415                          CON.RESET               ;CLEAR IT
2772  007640  005237  001504                  INC     ERCNT1          ;ALLOW 12 ERRORS AT MOST
2773  007644  001002                          BNE     12$
2774  007646  000137  010432                  JMP     EXT6            ;IF MORE, EXIT
2775  007652  032777  001000  171572  12$:    BIT     #BIT9,@RKDS     ;SIN, ON DOING WRITE?
2776  007660  001412                          BEQ     7$
2777  007662  004737  016162                  JSR     PC,GT4RG
2778  007666  104022                          ERROR   22              ;SIN ERROR ON DOING WRITE
2779  007670  104415                          CON.RESET
2780  007672  104416                  DRV.RESET
2781  007674  005237  001506                  INC     ERCNT2          ;ALLOW 12 ERRORS AT MOST
2782  007700  001002                          BNE     7$
2783  007702  000137  010432                  JMP     EXT6
2784                                                                  ;FIGURE OUT WHICH BUFFER IS
2785                                                                  ;AVAILABLE FOR USE
2786  007706  105737  001410          7$:     TSTB    BUFLG0          ;WAS PREVIOUS DSK-WRITE DONE
2787  007712  100003                          BPL     8$              ;USING BUFR 0?
2788  007714  005037  001410                  CLR     BUFLG0          ;YES, CLR FLAG INDICATING IT'S
2789                                                                  ;AVAILABLE NOW
2790  007720  000402                          BR      9$
2791  007722  005037  001412          8$:     CLR     BUFLG1          ;CLR FLAG INDICATING BUFR1
2792                                                                  ;IS AVAILABLE NOW
2793  007726  013737  001426  001424  9$:     MOV     NXTPAT,PRSPAT   ;'PRSPAT'S PATRNS WILL BE USED
2794                                                                  ;ON NEXT WRITE
2795  007734  010500                          MOV     R5,R0           ;FORM SEC # TO BE USED NXT TIME
2796  007736  116000  010426                  MOVB    SECPTR(R0),R0   ;GET SEC #
2797  007742  042737  000017  001432  10$:    BIC     #17,DSKADR      ;MASK SECTOR BITS FROM DSK-ADRES
2798  007750  050037  001432                  BIS     R0,DSKADR       ;FORM NXT DSK-ADRES
2799  007754  005205                          INC     R5              ;DONE WITH 12 SECTRS (3 BLOCKS)?
2800  007756  022705  000004                  CMP     #4,R5
2801  007762  001231                          BNE     2$              ;ON THIS SURFACE? IF NOT, GO
2802                                                                  ;DO NXT 4 SECTRS
2803  007764  032737  000001  001474          BIT     #BIT0,INDX1     ;WHICH SURFACE WAS DONE, 0 OR 1?
2804  007772  001415                          BEQ     11$             ;IF 0, GO DO SURFACE 1
2805  007774  005237  001474                  INC     INDX1           ;COUNT # OF TRACKS
2806  010000  001002                          BNE     .+6
```

```
 2807  010002  000137  010436                    JMP     TST7            ;EXIT IF DONE
 2808  010006  042737  000020  001432            BIC     #20,DSKADR      ;GO TO SUR 0, NEX" CYLINDER
 2809  010014  062737  000040  001432            ADD     #40,DSKADR
 2810  010022  000137  007444                    JMP     1$              ;GO BACK AND DO WRITE
 2811  010026  005237  001474            11$:    INC     INDX1           ;COUNT # OF TRACKS
 2812  010032  052737  000020  001432            BIS     #20,DSKADR      ;SET BIT FOR SUR 1
 2813  010040  000137  007444                    JMP     1$              ;GO, WRITE PATRNS ON SURFACE 1
 2814                                    ;*GET BUF#
 2815                                    ;*THIS ROUTINE FINDS OUT  WHICH BUFFER (IOBUF0, IOBUF1) OR ONE OF
 2816                                    ;*THE TWO BUFFERS SELECTED BY THE USER) TO USE
 2817                                    ;*FOR GENERATING PATTERNS. THEN IT SETS UP A FLAG INDICATING THAT
 2818                                    ;*BUFFER HAS TO BE FILLED UP. THE STARTING ADDRESS OF THE
 2819                                    ;*BUFFER IS STORED IN 'BUF #'.
 2820
 2821  010044  005737  001410            GETBUF: TST     BUFLG0          ;BUFR 0 AVAILABLE FOR USE?
 2822  010050  100007                            BPL     1$              ;YES, BRANCH
 2823  010052  052737  100000  001412            BIS     #BIT15,BUFLG1   ;NO, USE BUFR 1. INDICATE SO.
 2824  010060  013737  001406  001446            MOV     PBUF1,BUFNO     ;SAVE STARTING ADRES OF BUFR1
 2825  010066  000207                            RTS     PC
 2826  010070  052737  100000  001410  1$:       BIS     #BIT15,BUFLG0   ;INDICATED, USING BUF 0
 2827  010076  013737  001404  001446            MOV     PBUF0,BUFNO     ;SAVE STARTING ADRES OF BUFR 0
 2828  010104  000207                            RTS     PC              ;RETURN
 2829                                    ;*PTGEN0
 2830                                    ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS LONG
 2831                                    ;*BUFFER CONTAINING THE FOLLOWING PATTERNS
 2832                                    ;*FIRST BLOCK OF 256 WORDS:     125252
 2833                                    ;*SECOND BLOCK OF 256 WORDS:    052525 (COMPLEMENT OF ABOVE)
 2834                                    ;*THIRD BLOCK OF 256 WORDS:     010421
 2835                                    ;*YOU CAN USE ANY OTHER PATTERN/S (& ITS COMPLEMENT)
 2836                                    ;*MAKING THE CHANGES IN THE 2 LOCATIONS SHOWN BELOW.
 2837
 2838  010106  013700  001446            PTGEN0: MOV     BUFNO,R0                ;GET STARTING ADRES OF BUFR
 2839  010112  013701  010164                    MOV     PTRN01,R1       ;GET PATRN TO BE GENERATED
 2840  010116  012702  177400                    MOV     #-400,R2        ;IN THE FIRST 400 WORD BLOCK
 2841
 2842  010122  010120                    1$:     MOV     R1,(R0)+        ;GENERATE THE FIRST BLOCK
 2843  010124  005202                            INC     R2              ;WITH 'PAT01' PATRN
 2844  010126  001375                            BNE     1$              ;ALL DONE?
 2845
 2846  010130  012702  177400                    MOV     #-400,R2
 2847  010134  005101                            COM     R1              ;COMPLEMENT 'PAT01' PATAN
 2848  010136  010120                    2$:     MOV     R1,(R0)+        ;GENERATE 2ND BLOCK WITH
 2849  010140  005202                            INC     R2              ;'PAT01'S COMPLEMENT PATRN
 2850  010142  001375                            BNE     2$              ;ALL DONE?
 2851  010144  012702  177400                    MOV     #-400,R2
 2852  010150  013701  010166                    MOV     PTRN02,R1       ;GET PATRN TO BE GENERATED
 2853                                                                    ;FOR 3RD BLOCK
 2854  010154  010120                    3$:     MOV     R1,(R0)+        ;GENERATE 3RD BLOCK USING
 2855                                                                    ;'PAT02' PATRN
 2856  010156  005202                            INC     R2
 2857  010160  001375                            BNE     3$              ;ALL DONE?
 2858
 2859  010162  000207                            RTS     PC              ;RETURN
 2860
 2861  010164  125252            PTRN01: 125252                          ;CHANGE THESE LOCATIONS IF
 2862  010166  010421            PTRN02: 010421                          ;YOU WANT ANY OTHER PATTERNS
```

```
2863
2864                                    ;*PTGEN1
2865                                    ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS
2866                                    ;*LONG BUFFER CONTAINING THE FOLLOWING PATTERNS:
2867
2868                                    ;*FIRST BLOCK-256 WORDS 000001  FILL 1'S
2869                                    ;*                      000003
2870                                    ;*                        .
2871                                    ;*                      177777
2872
2873                                    ;*SECOND BLOCK          177776  FILL 0'S
2874                                    ;*                      177774
2875                                    ;*                        .
2876                                    ;*                      000000
2877
2878                                    ;*THIRD CLOCK           000001  FLOAT A 1
2879                                    ;*                      000020
2880                                    ;*                        .
2881                                    ;*                      100000
2882
2883                                    ;*'BUFNO' CONTAINS THE STARTING ADDRESS OF THE
2884                                    ;*BUFFER.
2885
2886  010170  012703  000001    PTGEN1: MOV     #1,R3           ;INITIALIZE PATRNS
2887  010174  012704  177776            MOV     #177776,R4
2888  010200  013700  001446            MOV     BUFNO,R0                ;GET STARTING ADRES OF BUFR
2889  010204  012702  177760            MOV     #-20,R2         ;SET COUNT
2890  010210  010301            1$:     MOV     R3,R1           ;INITIALIZE PATRN
2891  010212  010120            2$:     MOV     R1,(R0)+        ;GENERATE THE FIRST
2892  010214  000261                    SEC                     ;BLOCK USING "FILL 1'S"
2893  010216  006101                    ROL     R1
2894  010220  103374                    BCC     2$
2895  010222  005202                    INC     R2
2896  010224  001371                    BNE     1$              ;ALL DONE?
2897
2898  010226  012702  177760            MOV     #-20,R2
2899  010232  010401            3$:     MOV     R4,R1           ;INITIALIZE PATRN
2900  010234  010120            4$:     MOV     R1,(R0)+        ;GENERATE 2ND BLOCK
2901  010236  000241                    CLC                     ;USING "FILL 0'S"
2902  010240  006101                    ROL     R1
2903  010242  103774                    BCS     4$
2904  010244  005202                    INC     R2
2905  010246  001371                    BNE     3$              ;DONE?
2906
2907  010250  012702  177760            MOV     #-20,R2         ;SET COUNT
2908  010254  010301            5$:     MOV     R3,R1           ;INITIALIZE PATRN
2909  010256  010120            6$:     MOV     R1,(R0)+        ;GENERATE THE 3RD BLOCK
2910  010260  006301                    ASL     R1              ;USING "FLOAT A 1"
2911  010262  103375                    BCC     6$
2912  010264  005202                    INC     R2
2913  010266  001372                    BNE     5$              ;DONE?
2914  010270  000207                    RTS     PC              ;RETURN
2915
2916                                    ;*PTGEN2
2917                                    ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS LONG
2918                                    ;*BUFFER CONTAINING THE FOLLOWING PATTERNS:
```

```
2919
2920                                    ;*FIRST BLOCK-256 WORDS 000000  COUNT PATRN-LOWER BYTE
2921                                    ;*                      000001  0-377
2922                                    ;*                      000002
2923                                    ;*                      .
2924                                    ;*                      000377
2925
2926                                    ;*SECOND BLOCK          000000  COUNT PATRN-HIGHER BYTE
2927                                    ;*                      000400  0-377
2928                                    ;*                      001000
2929                                    ;*                      .
2930                                    ;*                      177400
2931
2932                                    ;*THIRD BLOCK           000000  COUNT PATRN-HIGHER & LOWER BYTE
2933                                    ;*                      000401  0-377, 0-377
2934                                    ;*                      .
2935                                    ;*                      177777
2936
2937                                    ;*'BUFNO' CONTAINS THE STARTING ADDRESS OF THE BUFFER.
2938
2939  010272  005001                    PTGEN2: CLR     R1              ;INITIALIZE PATRN
2940
2941  010274  013700  001446                    MOV     BUFNO,R0
2942  010300  010120                    1$:     MOV     R1,(R0)+        ;GENERATE 1ST BLOCK USING
2943  010302  105201                            INCB    R1              ;USING 'COUNT UP LOWER BYTE'
2944  010304  001375                            BNE     1$              ;DONE?
2945
2946  010306  005001                            CLR     R1
2947  010310  010120                    2$:     MOV     R1,(R0)+        ;GENERATE 2ND BLOCK
2948  010312  062701  000400                    ADD     #400,R1         ;USING 'COUNT UP HIGHER BYTE'
2949  010316  103374                            BCC     2$              ;DONE?
2950  010320  005001                            CLR     R1
2951  010322  010120                    3$:     MOV     R1,(R0)+        ;GENERATE 3RD BLOCK USING
2952  010324  062701  000401                    ADD     #401,R1         ;'COUNT UP HIGHER & LOWER BYTE'
2953  010330  103374                            BCC     3$              ;ALL DONE?
2954  010332  000207                            RTS     PC              ;RETURN
2955
2956
2957                                    ;PTGEN3
2958                                    ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS LONG
2959                                    ;*BUFFER CONTAINING THE FOLLOWING PATTERNS:
2960
2961                                    ;*FIRST BLOCK OF 256 WORDS:     167356 (COMPLEMENT OF 010421)
2962
2963                                    ;*SECOND BLOCK          177776  FLOAT A 0
2964                                    ;*                      177775
2965                                    ;*                      .
2966                                    ;*                      077777
2967
2968                                    ;*THIRD BLOCK           000377  COUNT UP HIGHER BYTE 0-377
2969                                    ;*                      000776  COUNT DOWN LOWER BYTE 377-0
2970                                    ;*                      .
2971                                    ;*                      177400
2972
2973                                    ;*'BUFNO' CONTAINS THE STARTING ADDRESS OF THE BUFFER.
2974
```

```
2975  010334  013700  001446          PTGEN3: MOV     BUFNO,R0
2976  010340  012702  177400                  MOV     #-400,R2
2977  010344  013701  010166                  MOV     PTRN02,R1       ;GET PATTERN
2978  010350  005101                          COM     R1              ;COMPLEMENT 'PAT02' PATRN
2979  010352  010120                  4$:     MOV     R1,(R0)+        ;GENERATE 1ST BLOCK
2980  010354  005202                          INC     R2
2981  010356  001375                          BNE     4$              ;ALL DONE?
2982
2983                                                                  ;2ND BLOCK
2984  010360  012702  177760                  MOV     #-20,R2
2985  010364  000261                  7$:     SEC
2986  010366  012701  177776                  MOV     #177776,R1
2987  010372  010120                  8$:     MOV     R1,(R0)+
2988  010374  006101                          ROL     R1
2989  010376  103775                          BCS     8$
2990  010400  005202                          INC     R2
2991  010402  001370                          BNE     7$              ;ALL DONE?
2992
2993                                                                  ;3RD BLOCK
2994  010404  012701  000377                  MOV     #377,R1
2995  010410  010102                          MOV     R1,R2           ;GENERATE 3RD BLOCK USING
2996  010412  010120                  9$:     MOV     R1,(R0)+        ;'COUNT DOWN LOWER BYTE'
2997  010414  060201                          ADD     R2,R1           ;'COUNT UP HIGHER BYTE'
2998  010416  022701  177777                  CMP     #-1,R1
2999  010422  001373                          BNE     9$              ;ALL DONE?
3000  010424  000207                          RTS     PC              ;RETURN
3001
3002                                  ;SECTOR POINTER TABLE. DATA TRANSFERS ARE DONE IN BLOCKS (3 SECTORS
3003                                  ;EACH) IN THE CYCLIC ORDER:  0-2, 6-10, 3-5, 11-13, 0-2, AND SO ON.
3004  010426      006                 SECPTR: .BYTE   6
3005  010427      003                         .BYTE   3
3006  010430      011                         .BYTE   11
3007  010431      000                         .BYTE   0
3008
3009  010432  004737  016772          EXT6:   JSR     PC,ABRT
3010
3011
3012                                  ;;*****************************************************************
3013                                  ;*TEST 7         READ, SOFTWARE COMPARE, WRITE CHECK OF THE PATTERNS
3014
3015                                          ;****TEST 6 (WRITING PATTERNS ON DISK) SHOULD BE DONE BEFORE DOING****
3016                                          ;****THIS TEST.****
3017
3018                                          ;*IN THIS TEST THE PATTERNS THAT WERE WRITTEN BEFORE (IN THE
3019                                          ;*PREVIOUS TEST) ARE READ BACK AND SOFTWARE COMPARED.
3020                                          ;*WHILE THE SOFTWARE COMPARISON IS TAKING PLACE, AN OVERLAPPING
3021                                          ;*'WRITE CHECK' IS DONE FOR THE SAME BLOCK.  THE READING
3022                                          ;*BACK OF EACH BLOCK (4 SECTORS) IS DONE IN THE SAME
3023                                          ;*MANNER AS DESCRIBED IN THE BEGINNING OF PREVIOUS TEST.
3024                                          ;*OVERLAPPING OPERATIONS AND STAGGERING OF BLOCKS ARE DONE IN
3025                                          ;*A SIMILIAR MANNER.
3026                                          ;*THE FOLLOWING IS A DESCRIPTION OF HOW ERRORS ARE HANDLED.
3027                                          ;*IF A 'SIN' OR 'HE' OCCURS IT IS REPORTED AND THAT BLOCK
3028                                          ;*IS SKIPPED.  IN THIS STAGE OF TESTING THESE ERRORS SHOULD NOT
3029                                          ;*NORMALLY OCCUR.
3030                                          ;*IF A CHECKSUM ERROR OCCURS, CONTROL IS TRANSFERRED TO
```

```
                                                        ;*'CSERROR'.  FIRST THE CSE IS REPORTED.  THE SECTOR GIVING
                                                        ;*CSE IS READ AGAIN.  IN ALL 3 TRIES ARE DONE.  IF STILL
                                                        ;*THE ERROR PRESISTS THAT SECTOR IS ABORTED AND THE REST
                                                        ;*OF THE SECTORS ARE READ AND CHECKED FOR CSE.  IF
                                                        ;*AGAIN SOME OTHER SECTOR GIVES CSE, REPORTING AND RETRIES
                                                        ;*ARE DONE AGAIN.
                                                        ;*NEXT SOFTWARE COMPARISON IS DONE OF THE DATA THAT WAS
                                                        ;*READ BACK.  ON GETTING A DATA MISCOMPARISON
                                                        ;*THE RELEVANT INFO(GOOD DATA, BAD DATA, ADDRESS ETC.) IS
                                                        ;*STORED.  AFTER THE WHOLE BLOCK HAS BEEN CHECKED, THE DATA
                                                        ;*ERROR/S IS/ARE REPORTED.  THEN THE BLOCK IS READ AGAIN.  IN ALL
                                                        ;*THREE TRIES ARE DONE.
                                                        ;*WHILE THE DATA COMPARISON (SOFTWARE) IS GOING ON THE 'WRITE
                                                        ;*CHECK' IS ALSO IN PROGRESS.  IF A WRITE CHECK ERROR OCCURS THE
                                                        ;*CONTROL IS TRANSFERRED TO 'WCEROR'.  WRITE CHECK OF THE SECTOR
                                                        ;*THAT GAVE WCE IS DONE AGAIN.  IN ALL THREE TRIES ARE DONE.
                                                        ;*NOTE THAT EVERY WCE IS REPORTED. IF ALL THE 4 SECTOR OF
                                                        ;*A BLOCK GAVE WCE'S, ALL 4 WILL BE REPORTED AND RETRIED.

                                                        ;*DEPENDING ON THE NATURE OF THE PROBLEM, THE ABOVE ERRORS
                                                        ;*CAN OCCUR IN ANY COMBINATION.  IT IS RECOMMENDED THAT ALL
                                                        ;*THE ERROR MESSAGE BE CAREFULLY CO-RELATED AND EXAMINED.  IT
                                                        ;*WILL PROVIDE A DEEP INSIGHT INTO THE PROBLEM.
                                                    ;;**************************************************************
010436  000004                                TST7:   SCOPE
010440  012737  177764  001510                        MOV     #-14,ERCNT3     ;ALLOW 12 ERRORS AT THE MOST
010446  012737  177773  001512                        MOV     #-5,ERCNT4      ;ALLOW 5 ERRORS AT THE MOST
010454  012737  177742  001514                        MOV     #-36,ERCNT5     ;ALLOW 10 ERRORS AT THE MOST
010462  012737  177742  001516                        MOV     #-36,ERCNT6     ;ALLOW 10 ERRORS AT THE MOST
010470  012737  177764  001520                        MOV     #-14,ERCNT7     ;ALLOW 12 ERRORS AT THE MOST
010476  012737  177742  001522                        MOV     #-36,ERCNT8     ;ALLOW 10 ERRORS AT THE MOST
010504  012737  177152  001474                        MOV     #-626,INDX1     ;SET COUNT FOR 626 TRACKS
010512  005037  001476                              CLR     INDX2           ;CLR COUNT FOR 4 BLOCKS ON A TRACK

010516  012737  001414  001424                      MOV     #PAT0,PRSPAT    ;INITLZE PTR TO PATRN GENRTR
010524  013737  001230  001450                      MOV     DRIVA0,ADRES    ;INITLZE DRV#,ADRES


010532  005037  001504                    BEGIN:  CLR     ERCNT1          ;IF > 0, MEANS THAT RETRIES
010536  005037  001506                            CLR     ERCNT2          ;DONE AFTER CSE OR CSE CHKD
010542  012737  177775  001252                      MOV     #-3,RETRY1      ;RETRY COUNT FOR CSE
010550  012737  177776  001254                      MOV     #-2,RETRY2      ;RETRY COUNT FOR SFTWRE MISCMP'N
010556  012737  000003  001256                      MOV     #3,RETRY3       ;RETRY COUNT FOR WCE

010564  013737  001450  001440                      MOV     ADRES,WDSKAD    ;DISK ADRES TO WRT CHK WITH
010572  013737  001406  001442                      MOV     PBUF1,WBUSAD    ;USE THIS BUFR 1 TO WRT CHK
                                                                            ;OR THE BUFR INDICATED BY THE USER
010600  012737  176400  001444                      MOV     #-1400,WWRDCN   ;WRT CHK 1 BLOCK=3SECS=1400 WRDS

010606  013737  001450  001432            READ:   MOV     ADRES,DSKADR    ;DISK ADRES TO READ FROM
010614  012737  176400  001436                      MOV     #-1400,WRDCNT   ;1 BLOCK = 3 SECTORS = 1400 WRDS
010622  013737  001404  001434                      MCV     PBUF0,BUSADR    ;USE 'IOBUF0' TO READ INTO
                                                                            ;OR THE BUFR INDICATED BY THE USER

010630  000404                                      BR      RDAGAIN
```

```
3087  010632  104415                  LUPSIN: CON.RESET
3088  010634  104416                          DRV.RESET
3089  010636  000401                          BR      RDAGAIN
3090
3091  010640  104415                  LUPHE:  CON.RESET
3092
3093  010642  013777  001432  170614  RDAGAIN:        MOV     DSKADR,@RKDA    ;READ FROM THIS DSK-ADRES
3094  010650  013777  001436  170602          MOV     WRDCNT,@RKWC    ;THIS # OF WORDS
3095  010656  013777  001434  170576          MOV     BUSADR,@RKBA    ;INTO THIS BUFR
3096
3097  010664  012777  000405  170564          MOV     #405,@RKCS      ;READ,SSE,GO
3098
3099
3100  010672  013737  001406  001446          MOV     PBUF1,BUFNO     ;SET UP STARTING ADRES
3101  010700  017737  170520  001430          MOV     @PRSPAT,PGSUBR
3102  010706  004777  170516                  JSR     PC,@PGSUBR      ;GO GENERATE A BUFFER
3103                                                                  ;OF 1400 WORDS USING THIS
3104                                                                  ;PATRN GENRATR
3105
3106  010712  104421                          CON.RDY                 ;DONE WITH PATRN GENRTNG,
3107                                                                  ;WAIT FOR CNT RDY TO SET
3108                                                                  ;(FROM PREVIOUS READ)
3109
3110                                                                  ;CNT RDY SET
3111  010714  032777  040000  170534          BIT     #BIT14,@RKCS    ;HARD ERROR?
3112  010722  001416                          BEQ     NOHE
3113
3114  010724  012737  010640  001110          MOV     #LUPHE,$LPERR
3115  010732  004737  016214                  JSR     PC,GETINF
3116  010736  104023                          ERROR   23              ;HARD ERROR
3117  010740  104415                          CON.RESET
3118  010742  005237  001510                  INC     ERCNT3          ;ALLOW 12 ERRORS AT MOST
3119  010746  001002                          BNE     1$
3120  010750  000137  012170                  JMP     EXT7            ;IF MORE, EXIT
3121  010754  000137  011512          1$:     JMP     FINISH
3122  010760  032777  001000  170464  NOHE:   BIT     #BIT9,@RKDS     ;SIN SET?
3123  010766  001417                          BEQ     NOSIN           ;NO
3124
3125  010770  012737  010632  001110          MOV     #LUPSIN,$LPERR
3126  010776  004737  016214                  JSR     PC,GETINF
3127  011002  104011                          ERROR   11              ;SIN ON READ
3128  011004  104415                          CON.RESET
3129  011006  104416                          DRV.RESET
3130  011010  005237  001512                  INC     ERCNT4          ;ALLOW 5 ERRORS AT MOST
3131  011014  001002                          BNE     1$
3132  011016  000137  012170                  JMP     EXT7            ;IF MORE, EXIT
3133  011022  000137  011512          1$:     JMP     FINISH
3134
3135  011026  005737  001504          NOSIN:  TST     ERCNT1          ;CHECKING CSE FOR 1ST TIME
3136  011032  001031                          BNE     WRTCHK          ;NO,BRANCH
3137  011034  005237  001504                  INC     ERCNT1          ;INDICATE THAT CSE HAS BEEN
3138                                                                  ;CHECKED ONCE
3139
3140  011040  032777  000002  170406          BIT     #BIT1,@RKER     ;CHECK SUM EROR?
3141  011046  001423                          BEQ     WRTCHK
3142  011050  012737  010532  001110          MOV     #BEGIN,$LPERR
```

```
3143  011056  004737  016214                  JSR     PC,GETINF
3144  011062  013737  001252  001202          MOV     RETRY1,$REG10   ;GET THE RETRY #
3145  011070  062737  000004  001202          ADD     #4,$REG10       ;SAVE IT FOR TYPEOUT
3146  011076  104024                          ERROR   24              ;CSE
3147  011100  005237  001514                  INC     ERCNT5          ;ALLOW 10 ERRORS AT MOST
3148  011104  001002                          BNE     1$
3149  011106  000137  012170                  JMP     EXT7            ;IF MORE, EXIT
3150  011112  000137  011654          1$:     JMP     CSERCR          ;GO, SERVICE CSE
3151
3152  011116  005037  001506          WRTCHK: CLR     ERCNT2          ;CLR FLAG INDICATING SOFTWARE
3153                                                                  ;COMPARE DONE
3154  011122  022737  000003  001256          CMP     #3,RETRY3       ;WRT CHK DONE BEFORE OR
3155  011130  001016                          BNE     SFTCMP          ;IT'S 1ST TIME?
3156                                                                  ;IF DONE,BRANCH OTHERWISE DO IT
3157  011132  013777  001440  170324  WCAGAIN: MOV     WDSKAD,@RKDA    ;WRT CHK FROM THIS DSK-ADRES
3158  011140  013777  001444  170312          MOV     WWRDCN,@RKWC    ;THIS # FO WORDS
3159  011146  013777  001442  170306          MOV     WBUSAD,@RKBA    ;WITH THIS BUFFER
3160
3161  011154  012777  000407  170274          MOV     #407,@RKCS      ;WRT CHK,GO,SSE
3162
3163  011162  005337  001256                  DEC     RETRY3          ;INDICATE WRT CHK DONE
3164
3165  011166  005737  001506          SFTCMP: TST     ERCNT2          ;SFTWARE CMPARE DONE ONCE BEFORE
3166  011172  001060                          BNE     WCREPT          ;IF SFTWARE CMPARE HAS BEEN DONE
3167                                                                  ;ONCE DON'T DO IT AGAIN. OTHERWISE,
3168  011174  005237  001506                  INC     ERCNT2          ;DO IT, INDICATE IT IS DONE.
3169                                                                  ;MORE THAN ONCE BEFORE.
3170                                                                  ;IF THIS IS 1ST TIME THRU &
3171                                                                  ;WRT CHK WAS DONE ONCE BEFORE
3172                                                                  ;DO SOFTWARE COMPARISON OF
3173                                                                  ;THE DATA THAT WAS READ FROM
3174                                                                  ;THE DISK
3175
3176  011200  012702  001266                  MOV     #BUFR,R2        ;INITLZE PTR TO BUFR STORING
3177                                                                  ;ADRES OF BAD DATA
3178  011204  012703  001320                  MOV     #BUFR1,R3       ;STORE EXPCTD DATA STARTING HERE
3179  011210  012704  001352                  MOV     #BUFR2,R4       ;STORE RECVD (BAD) DATA
3180                                                                  ;STARTING HERE
3181
3182  011214  005037  001500                  CLR     INDX3           ;CLR FLAG INDICATING MISCMPRE
3183
3184  011220  013700  001404          COMPAR: MOV     PBUF0,R0        ;INITLZE PTR TO 'RECVD DATA' BUFR
3185  011224  012737  177764  001502          MOV     #-14,INDX4      ;STORE AND REPORT 12 OR LESS DATA ERRORS
3186  011232  013701  001406                  MOV     PBUF1,R1        ;INITLZE PTR TO 'EXPCTD DATA' BUFR
3187  011236  012737  176400  001260          MOV     #-1400,INADR    ;SET COUNT
3188  011244  021011                  CMPAGAN: CMP     (R0),(R1)       ;CORRECT WORD READ FROM DISK?
3189  011246  001402                          BEQ     1$
3190  011250  000137  012016                  JMP     MISCMP          ;BRANCH IF MISCMPRE ERROR
3191  011254  005720                  1$:     TST     (R0)+           ;INCRMNT PTRS
3192  011256  005721                          TST     (R1)+           ;TO NXT WORDS
3193  011260  005237  001260                  INC     INADR           ;DONE WITH CMPRISON?
3194  011264  001367                          BNE     CMPAGAN         ;IF NOT, COMPARE THE RES
3195
3196  011266  005737  001500                  TST     INDX3           ;WAS THERE A BAD DATA WORD
3197                                                                  ;(EVEN AFTR RETRYING)
3198  011272  001420                          BEQ     WCREPT          ;NO, BRANCH
```

```
                        ;T7          READ, SOFTWARE COMPARE, WRITE CHECK OF THE PATTERNS

3199  011274  012737  010606  001110  REPMSC: MOV     #READ,$LPERR
3200  011302  104025                  ERROR   25              ;DATA ERROR
3201  011304  005237  001516          INC     ERCNT6          ;ALLOW 10 ERRORS AT MOST
3202  011310  001002                  BNE     1$
3203  011312  000137  012170          JMP     EXT7            ;IF MORE, EXIT
3204  011316  005737  001254  1$:     TST     RETRY2
3205  011322  001404                  BEQ     WCREPT
3206  011324  005237  001254          INC     RETRY2
3207  011330  000137  010606          JMP     READ
3208
3209  011334  104421          WCREPT: CON.RDY                 ;WAIT FOR CNTRL RDY FROM
3210                                                          ;PREVIOUS WRT CHK
3211  011336  022737  177776  001254          CMP     #-2,RETRY2      ;IF THERE WAS A RETRY AFTER MISC
3212                                                          ;-OMPARISON, DO WRT CHK AGAIN
3213  011344  001417                  BEQ     ERWCHK
3214  011346  000401                  BR      LUPWCE+2
3215  011350  104415          LUPWCE: CON.RESET
3216  011352  013777  001440  170104          MOV     WDSKAD,@RKDA    ;WRT CHK WITH THIS DSK-ADRES
3217  011360  013777  001444  170072          MOV     WWRDCN,@RKWC    ;THIS # OF WORDS
3218  011366  013777  001442  170066          MOV     WBUSAD,@RKBA    ;THIS BUS ADRES
3219  011374  012777  000407  170054          MOV     #407,@RKCS
3220
3221  011402  104421                  CON.RDY
3222  011404  012737  011350  001110  ERWCHK: MOV     #LUPWCE,$LPERR
3223  011412  032777  040000  170032          BIT     #BIT14,@RKDS    ;HARD EROR?(FROM WRT CHK)
3224  011420  001410                  BEQ     XHE             ;NO,BRANCH
3225
3226  011422  004737  016214                  JSR     PC,GETINF
3227  011426  104026                  ERROR   26              ;HE ON WRT CHK
3228  011430  005237  001520          INC     ERCNT7          ;ALLOW 12 ERRORS AT MOST
3229  011434  001002                  BNE     XHE
3230  011436  000137  012170          JMP     EXT7            ;IF MORE, EXIT
3231
3232  011442  032777  000001  170004  XHE:    BIT     #BIT0,@RKER     ;WRITE CHECK EROR?
3233  011450  001420                  BEQ     FINISH
3234  011452  004737  016214                  JSR     PC,GETINF
3235  011456  012737  000003  001202          MOV     #3,$REG10       ;GET TRY #
3236  011464  163737  001256  001202          SUB     RETRY3,$REG10   ;SAVE IT FOR TYPEOUT
3237  011472  104027                  ERROR   27              ;WRT CHK EROR
3238  011474  005237  001522          INC     ERCNT8          ;ALLOW 10 ERRORS AT MOST
3239  011500  001002                  BNE     1$
3240  011502  000137  012170          JMP     EXT7            ;IF MORE, EXIT
3241  011506  000137  012062  1$:     JMP     WCEROR
3242
3243
3244                                                          ;THERE WAS NO WCE. DONE
3245                                                          ;WITH ALL CHECKING FOR SOFT
3246                                                          ;ERORS,ETC. MODIFY PARAMETERS
3247                                                          ;TO CHECK NXT BLOCK ON
3248                                                          ;THE DISK
3249
3250  011512  022737  001422  001424  FINISH: CMP     #PAT3,PRSPAT    ;FIND OUT THE NXT PATRN GENRATR
3251  011520  001404                  BEQ     1$              ;TO USE FOR GENRATING THE
3252  011522  062737  000002  001424          ADD     #2,PRSPAT       ;BUFR,STORE POINTER TO
3253  011530  000403                  BR      2$              ;GENRATR ROUTINE IN 'PRSPAT'
3254  011532  012737  001414  001424  1$:     MOV     #PAT0,PRSPAT    ;NOTE THER R 4 PAT-GENRATRS
```

```
3255
3256  011540  013701  001476          2$      MOV     INDX2,R1        ;PAT0-PAT1-PAT2-PAT3----PAT0-
3257  011544  116101  010426                  MOVB    SECPTR(R1),R1   ;FORM SECTR # TO BE USED NEXT
3258  011550  042737  000017  001450  3$:     BIC     #17,ADRES       ;GET SECTR #
3259  011556  050137  001450                  BIS     R1,ADRES        ;MASK SECTR BITS
3260                                                                  ;FORM THE NEW DISK-ADRES
3261                                                                  ;FORM THE NXT BLOCK TO BE
3262  011562  005237  001476                  INC     INDX2           ;CHECKED.
3263                                                                  ;DONE ALL 4 BLOCKS(3 SECS
3264  011566  022737  000004  001476          CMP     #4,INDX2        ;EACH) ON THIS TRACK?
3265  011574  001025                          BNE     GOBAK           ;IF NOT, GO BAK & DO THE
3266                                                                  ;NXT BLOCK ON THIS TRACK
3267
3268  011576  005037  001476          DONTRK: CLR     INDX2           ;REINITLZE COUNT FOR
3269                                                                  ;4 BLOCKS ON THIS TRACK
3270  011602  032737  000001  001474          BIT     #BIT0,INDX1     ;WHICH SUR TO DO NXT? 0 OR 1
3271  011610  001407                          BEQ     DOSURI          ;BRANCH, DO SUR 1 NXT
3272
3273  011612  062737  000040  001450          ADD     #40,ADRES
3274
3275  011620  042737  000020  001450          BIC     #20,ADRES       ;CLR SUR BIT, DO SUR 0 NXT
3276  011626  000403                          BR      DONE
3277
3278  011630  052737  000020  001450  DOSUR1: BIS     #20, ADRES      ;SET SUR BIT, DO SUR 1 NXT
3279
3280  011636  005237  001474          DONE:   INC     INDX1           ;DONE WITH ALL 626 TRACKS?
3281  011642  001002                          BNE     GOBAK
3282  011644  000137  012174                  JMP     TST10
3283
3284  011650  000137  010532          GOBAK:  JMP     BEGIN           ;IF NOT, GO BAK & CHECK
3285                                                                  ;THE NXT TRACK
3286
3287                                  ;CSEROR                         ;THIS IS THE ENTRY POINT
3288                                                                  ;FOR SERVICE ROUTINE FOR CHECK
3289  011654                          CSEROR:                         ;SUM EROR. CONTROL IS XFERED
3290                                                                  ;HERE ONCE CSE OCCURS
3291
3292  011654  017700  167604                  MOV     @RKDA,R0        ;GET RKDA AFTER CSE
3293  011660  010001                          MOV     R0,R1           ;SAVE RKDA
3294  011662  032700  000017                  BIT     #17,R0          ;FORM THE ADRES OF THE SECTR
3295  011666  001002                          BNE     1$              ;WHICH GAVE CSE
3296  011670  162700  000004                  SUB     #4,R0
3297  011674  005300                  1$:     DEC     R0              ;R0 CONTAINS DSK-ADRES WHERE
3298                                                                  ;CSE OCURED
3299  011676  020037  001432                  CMP     R0,DSKADR       ;DID A PREVIOUS RETRY (IF ANY)
3300                                                                  ;GIVE CSE ON SAME ADRES
3301  011702  001021                          BNE     2$              ;NO, THIS IS A FRESH CSE, BRANCH.
3302                                                                  ;IF THIS WAS A CSE ON A
3303  011704  005237  001252                  INC     RETRY1          ;RETRY INCMNT RETRY COUNT.
3304                                                                  ;BRANCH IF 3 RETRIES HAVEN'T
3305  011710  001021                          BNE     3$              ;BEEN DONE
3306                                                                  ;GO REPORT CSE
3307                                                                  ;IF CSE WAS IN THE LAST SECTR OF
3308                                                                  ;THE 3 SEC BLOCK, GO TO 'WRTCHK'
3309                                                                  ;OTHERWISE CHECK THE REST OF
3310                                                                  ;THE SECTORS FOR CSE.
```

```
3311  011712  010102                        MOV     R1,R2
3312  011714  042702  177760                BIC     #177760,R2
3313  011720  001002                  7$:   BNE     8$
3314  011722  000137  011116                JMP     WRTCHK
3315  011726  162702  000003          8$:   SUB     #3,R2
3316  011732  100372                        BPL     7$
3317
3318  011734  010100                  6$:   MOV     R1,R0
3319  011736  012737  177775  001252        MOV     #-3,RETRY1
3320  011744  000403                        BR      3$
3321
3322  011746  012737  177776  001252  2$:   MOV     #-2,RETRY1      ;ALLOW 2 MORE TRIES FOR
3323                                                                ;THE CSE ON SAME DISK-ADRES
3324
3325  011754  010037  001432          3$:   MOV     R0,DSKADR       ;SAVE DSK-ADRES FOR DOING
3326                                                                ;THE RETRY-READ
3327  011760  163700  001450                SUB     ADRES,R0        ;MODIFY THE
3328  011764  005200                        INC     R0              ;BUS ADRES & WORD COUNT
3329  011766  005300                  4$:   DEC     R0              ;TO BE USE ON RETRY-
3330  011770  001407                        BEQ     5$              ;READ
3331  011772  062737  001000  001434        ADD     #1000,BUSADR
3332  012000  062737  000400  001436        ADD     #400,WRDCNT
3333  012006  000767                        BR      4$
3334
3335  012010  104415                  5$:   CON.RESET               ;CLR THE CSE IN RKER
3336  012012  000137  010642                JMP     RDAGAIN         ;GO BACK, READ AGAIN
3337                                                                ;RETRY
3338
3339
3340
3341                                  ;MISCMP                       ;THIS IS THE ENTRY POINT
3342                                                                ;FOR SERVICE ROUTINE, FOR
3343                                                                ;DATA EROR (MISCOMPARISON)
3344                                                                ;ON SOFTWARE COMPARISON
3345                                                                ;OF DATA READ FROM THE DISK BLOCK
3346
3347
3348  012016  104421                  MISCMP: CON.RDY               ;WAIT FOR CNTRL RDY FROM
3349                                                                ;PREVIOUS WRT CHK
3350  012020  032777  040000  167430        BIT     #BIT14,@RKCS
3351  012026  001401                        BEQ     1$
3352  012030  104415                        CON.RESET               ;CLR WCE IN RKER
3353                                                                ;BUT STILL DATA EROR ENTER HERE
3354  012032  010022                  1$:   MOV     R0,(R2)+        ;STORE MEM ADRES WHERE
3355                                                                ;DATA EROR OCCURED
3356  012034  012123                        MOV     (R1)+,(R3)+     ;SAVE EXPCTD DATA
3357  012036  012024                        MOV     (R0)+,(R4)+     ;SAVE DATA RECVD (BAD)
3358  012040  005237  001500                INC     INDX3           ;INDICATE MISCMPRISON
3359
3360  012044  005237  001502                INC     INDX4           ;STORE ONLY 12(DEC) ERORS
3361  012050  001402                        BEQ     4$              ;IF 12 ERORS, GO REPORT THEM
3362
3363  012052  000137  011244                JMP     CMPAGAN         ;GO BACK & CMPARE THE REST
3364
3365  012056  000137  011274          4$:   JMP     REPMSC
3366
```

```
                                                        ;THIS THE ENTRY POINT FOR
                                                        ;EROR SERVICE ON GETTING A
                                                        ;WRITE CHECK EROR.
012062  005737  001256          WCEROR: TST     RETRY3          ;DONE 3 TRIES?
012066  003035                          BGT     CLRERR          ;IF NOT SKIP, OTHERWISE REPORT
                                                        ;W C EROR
012070  012737  000003  001256          MOV     #3,RETRY3       ;SET CONT FOR RETRIES
012076  017700  167362                  MOV     @RKDA,R0        ;IF WCE WAS IN THE LAST SECTOR
012102  010002                          MOV     R0,R2           ;OF THE BLOCK, NO MORE SECS
012104  042702  177760                  BIC     #177760,R2      ;TO CHECK, GO TO 'FINISH'
012110  001002                  3$:     BNE     4$              ;IF IT WASN'T LAST SECTOR OF THE
012112  000137  011512                  JMP     FINISH          ;BLOCK, THEN CHECK REMAINING
012116  162702  000003          4$:     SUB     #3,R2           ;SECTORS. (STARTING FROM THE
012122  100372                          BPL     3$              ;SEC AFTER THE ONE GIVING WCE)
012124  010001                  1$:     MOV     R0,R1           ;SAVE DISK ADRES
012126  163700  001440                  SUB     WDSKAD,R0
012132  010137  001440                  MOV     R1,WDSKAD       ;GET SAVED DISK ADRES
012136  005200                          INC     R0
012140  005300                  2$:     DEC     R0
012142  001407                          BEQ     CLRERR
012144  062737  001000  001442          ADD     #1000,WBUSAD    ;FORM THE NEW BUS ADRES
012152  062737  000400  001444          ADD     #400,WWRDCN     ;FORM THE NEW WORD COUNT
012160  000767                          BR      2$
012162  104415                  CLRERR: CON.RESET
012164  000137  011132                  JMP     WCAGAIN

012170  004737  016772          EXT7:   JSR     PC,ABRT


                                ;;*****************************************************************
                                ;*TEST 10        WRITE, WRITE CHECK ON CYLINDERS 127, 128
                                        ;*THIS TEST WRITES 12 UNIQUE PATTERNS (ONE FOR EACH
                                        ;*SECTOR) ON CYLINDERS 127 AND 128, THEN WRITE
                                        ;*CHECK IS DONE TO SEE IF THEY WERE WRITTEN
                                        ;*CORRECTLY.  IT SHOULD BE NOTED THAT THERE IS
                                        ;*CHANGE IN 'WRITE' CURRENT AT THIS CYLINDER.
                                        ;*PATTERNS ARE RELOCATED ON THE CYLINDERS AFTER EACH
                                        ;*WRITE/WRITE CHECK CYCLE.  THUS THE FIRST TIME
                                        ;*PATTERN 'SP1' IS WRITTEN ON SECTOR 0, PATTERN 'SP2' ON
                                        ;*SECTOR 2, ETC.  AFTER THIS WRITE/WRITE CHECK CYCLE
                                        ;*IS OVER PATTERN 'SP2' IS WRITTEN ON SECTOR 0, PATTERN
                                        ;*:'SP3' ON SECTOR 1 AND SO ON.  THIS WRITE/WRITE
                                        ;*CHECK CYCLE IS REPEATED 12 TIMES, THUS
                                        ;*THE LAST WRITE/WRITE CHECK IS DONE USING
                                        ;*PATTERN 'SP12' ON SECTOR 0, PATTERN 'SP1' IS
                                        ;*WRITTEN ON SECTOR 1, PATTERN 'SP2' ON SECTOR 2,
                                        ;*ETC.  IF YOU WANT TO WRITE ANY OTHER PATTERNS
                                        ;*FILL IN THE PATTERNS YOU WANT IN LOCATIONS
                                        ;*'SP1', 'SP2',...ETC.
                                ;;*****************************************************************
012174  000004                  TST10:  SCOPE
```

```
3423
3424  012176  012737  177764  001504          MOV     #-14,ERCNT1     ;ALLOW 12 ERRORS AT MOST
3425  012204  012737  177764  001506          MOV     #-14,ERCNT2     ;ALLOW 12 ERRORS AT MOST
3426  012212  012737  177723  001510          MOV     #-55,ERCNT3     ;ALLOW 15 ERRORS AT MOST
3427  012220  012702  012702                  MOV     #SP1,R2         ;INITIALIZE POINTER TO PATRN
3428  012224  010201                  DOWRT:  MOV     R2,R1
3429  012226  012700  007740                  MOV     #7740,R0        ;SET UP CYL ADRES BITS (127)
3430  012232  053700  001230                  BIS     DRIVAD,R0       ;SET UP DRIVE # BITS
3431  012236  005003                  WRL01:  CLR     R3
3432  012240  104415                  WRERR:  CON.RESET
3433
3434  012242  010077  167216          WRL0:   MOV     R0,@RKDA        ;ADRES THE DRIVE
3435  012246  012777  177400  167204          MOV     #-400,@RKWC     ;WRITE 1 SECTOR
3436  012254  010177  167202                  MOV     R1,@RKBA        ;USE THIS PATTERN
3437  012260  012777  004003  167170          MOV     #4003,@RKCS     ;WRITE, GO
3438  012266  104421                  CON.RDY
3439  012270  032777  140000  167160          BIT     #140000,@RKCS   ;ANY ERROR?
3440  012276  001414                  BEQ     4$
3441  012300  012737  012240  001110          MOV     #WRERR,$LPERR   ;SET ADRES FOR LOOPING ON ERROR
3442  012306  004737  016162                  JSR     PC,GT4RG        ;GET TKCS, ER, DS, DA
3443  012312  104021                  ERROR   21              ;ERROR OCCURRED ON DOING A WRITE
3444  012314  104415                  CON.RESET               ;CLEAR THE ERROR
3445  012316  005237  001504                  INC     ERCNT1          ;ALLOW 12 ERRORS ONLY
3446  012322  001002                  BNE     4$
3447  012324  000137  012732                  JMP     EXT10
3448
3449  012330  005200                  4$:     INC     R0
3450  012332  005203                  INC     R3              ;KEEP COUNT
3451  012334  022701  012730                  CMP     #SP12,R1        ;USE PATTERNS IN A CYCLIC FASHION
3452  012340  001002                  BNE     3$
3453  012342  012701  012700                  MOV     #SP1-2,R1
3454  012346  005721                  3$:     TST     (R1)+           ;INCREMENT POINTERS TO NEXT PATTERN
3455  012350  020327  000014                  CMP     R3,#14          ;DONE SURFACE 0?
3456  012354  002732                  BLT     WRL0            ;NO
3457  012356  001005                  BNE     2$              ;YES, IF CHANGING HEADS
3458  012360  010201                  MOV     R2,R1
3459  012362  042700  000017                  BIC     #17,R0          ;SET UP CORRECT ADDRESS ETC.
3460  012366  052700  000020                  BIS     #20,R0
3461
3462  012372  020327  000030          2$:     CMP     R3,#30          ;DONE WITH WRITING SURFACE 1?
3463  012376  001321                  BNE     WRL0            ;NO, BRANCH
3464
3465  012400  032700  007700          WRHI:   BIT     #7700,R0        ;DONE WITH BOTH CYLINDERS - 127 & 128?
3466  012404  001405                  BEQ     DOWCHK          ;YES
3467  012406  012700  010000                  MOV     #10000,R0       ;NO, DO CYLINDER 128
3468  012412  053700  001230                  BIS     DRIVAD,R0
3469  012416  000707                  BR      WRL01           ;GO BACK
3470                                                          ;CYLINDERS 127 AND 128 HAVE BEEN
3471
3472
3473  012420  010201                  DOWCHK: MOV     R2,R1           ;WRITTEN, NOW DO WRITE CHECK
3474  012422  012737  177775  001252          MOV     #-3,RETRY1      ;INITIALIZE POINTER TO FIRST PATTERN
3475  012430  012700  010000                  MOV     #10000,R0       ;RETRY COUNT
3476  012434  053700  001230                  BIS     DRIVAD,R0       ;DO CYLINDER 128 FIRST
3477  012440  005003                  WCHI1:  CLR     R3
3478  012442  010077  167016          WCERR:  MOV     R0,@RKDA        ;ADRES THE DRIVE
```

```
3479
3480  012446  012777  177400  167004  WCHI:   MOV     #-400,@RKWC     ;WRITE CHECK 1 SECTOR
3481  012454  010177  167002                  MOV     R1,@RKBA        ;WITH THIS PATTERN
3482  012460  012777  004007  166770          MOV     #4007,@RKCS     ;WRITE CHECK, GO
3483  012466  104421                          CON.RDY
3484
3485  012470  032777  040000  166754          BIT     #40000,@RKDS    ;HE?
3486  012476  001406                          BEQ     1$              ;NO
3487  012500  004737  016214                  JSR     PC,GETINF
3488  012504  104026                          ERROR   26              ;HE ON DOING WRT CHK
3489  012506  005237  001506                  INC     ERCNT2          ;ALLOW 12 ERRORS ONLY
3490  012512  001507                          BEQ     EXT10           ;IF MORE, EXIT
3491  012514  032777  000001  166732  1$:     BIT     #BIT0,@RKER     ;WCE?
3492  012522  001425                          BEQ     4$              ;NO
3493  012524  012737  012442  001110          MOV     #WCERR,$LPERR   ;SET ADRES FOR LOOPING ON ERROR
3494  012532  004737  016214                  JSR     PC,GETINF       ;GET INFO ON ERROR
3495  012536  013737  001252  001202          MOV     RETRY1,$REG10
3496  012544  062737  000004  001202          ADD     #4,$REG10       ;WCE ON DOING WRITE CHECK, WITH
3497  012552  104027                          ERROR   27              ;PATTERN STORED IN R1
3498  012554  005237  001252                  INC     RETRY1          ;DO 3 TIMES IN ALL
3499  012560  001330                          BNE     WCERR
3500  012562  005237  001510                  INC     ERCNT3          ;ALLOW 15 ERRORS ONLY
3501  012566  001461                          BEQ     EXT10           ;IF MORE, EXIT
3502  012570  012737  177775  001252          MOV     #-3,RETRY1
3503
3504  012576  005200                  4$:     INC     R0              ;KEEP TRACK OF DISK-ADRES
3505  012600  005203                          INC     R3              ;AND COUNT
3506  012602  022701  012730                  CMP     #SP12,R1        ;USE PATTERNS IN CYCLIC
3507  012606  001002                          BNE     3$
3508  012610  012701  012700                  MOV     #SP1-2,R1       ;FASHION
3509  012614  005721                  3$:     TST     (R1)+           ;INCREMENT POINTER TO NEXT PATTERN
3510  012616  020327  000014                  CMP     R3,#14          ;DONE SURFACE 0?
3511  012622  002711                          BLT     WCHI            ;NO
3512  012624  001005                          BNE     2$
3513  012626  010201                          MOV     R2,R1           ;IF CHANGING HEADS (0-1), SET CORRECT
3514  012630  042700  000017                  BIC     #17,R0          ;ADRES BITS
3515  012634  052700  000020                  BIS     #20,R0
3516
3517  012640  020327  000030          2$:     CMP     R3,#30          ;DONE WRITE CHECKING SURFACE 1?
3518  012644  001300                          BNE     WCHI            ;NO, GO BACK
3519
3520  012646  032700  007700          WCLO:   BIT     #7700,R0        ;DONE BOTH CYLINDERS - 127, 128?
3521  012652  001005                          BNE     REPEAT          ;YES, BRANCH
3522  012654  012700  007740                  MOV     #7740,R0        ;DO CYLINDER 127 NOW
3523  012660  053700  001230                  BIS     DRIVAD,R0
3524  012664  000665                          BR      WCHI1
3525
3526
3527  012666  005722                  REPEAT: TST     (R2)+           ;RELOCATE THE PATTERNS ON THE
3528  012670  020227  012732                  CMP     R2,#SP12+2      ;CYLINDERS AND DO IT AGAIN
3529  012674  001420                          BEQ     TST11           ;;EXIT
3530  012676  000137  012224                  JMP     DOWRT           ;THIS TEXT)
3531
3532
3533
3534                                  ;PATTERNS TO BE USED
```

```
3535
3536  012702  177777                SP1:    .WORD   177777          ;IF YOU WANT TO WRITE ANY
3537  012704  052525                SP2:    .WORD   052525          ;OTHER PATTERNS, CHANGE THESE
3538  012706  111111                SP3:    .WORD   111111          ;12 LOCATIONS TO ANY PATTERN
3539  012710  010421                SP4:    .WORD   010421          ;YOU WANT.
3540  012712  102041                SP5:    .WORD   102041
3541  012714  010101                SP6:    .WORD   010101
3542  012716  040201                SP7:    .WORD   040201
3543  012720  000401                SP8:    .WORD   000401
3544  012722  031463                SP9:    .WORD   031463
3545  012724  070707                SP10:   .WORD   070707
3546  012726  007417                SP11:   .WORD   007417
3547  012730  041020                SP12:   .WORD   041020
3548
3549  012732  004737  016772        EXT10:  JSR     PC,ABRT
3550
3551
3552
3553
3554
3555                                ;;*****************************************************************
3556                                ;*TEST 11        SEEK FUNCTION TIMER
3557                                ;*SEEK TIMER
3558                                ;*IN THIS PART OF THE PROGRAM SEEKS ARE TIMED BETWEEN A PARTICULAR SET
3559                                ;*OF CYLINDERS, BOTH IN THE FORWRAD  DIRECTION (0-312) AND REVERSE(312-0 .
3560                                ;******CAUTION******
3561                                ;*IT SHOULD BE NOTED THAT THE SECTOR COUNTER (IN RKDS) IS USED AS THE REAL
3562                                ;*TIME CLOCK TO DO THE SEEK TIMING. FOR THE TIMES TO BE RELIABLE, THE
3563                                ;*SECTOR COUNTER SHOULD BE ACCURATE WITHIN THE SPECIFICATIONS OF THE DISK
3564                                ;*SPEED:     1500 RPM = 40 MILI SECS/REV =3.33 MILI SECS/SECTOR.
3565                                ;*VARIATION: +-30  RPM
3566                                ;*****************************************************************
3567  012736  000004                TST11:  SCOPE
3568  012740  032777  000100  166172        BIT     #SW6,@SWR       ;INHIBIT TIMER?
3569  012746  001002                        BNE     .+6
3570  012750  000137  014106                JMP     PLTGRPH
3571  012754  104415                        CON.RESET
3572  012756  104416                        DRV.RESET
3573
3574  012760  012737  177771  001252        MOV     #-7,RETRY1      ;COUNT FOR 7 DIFRNT SEEK TIMES
3575                                                                ;TO BE RECORDED
3576
3577  012766  104401  012774                TYPE    ,65$            ;;TYPE ASCIZ STRING
3578  012772  000424                        BR      64$             ;;GET OVER THE ASCIZ
3579                                ;;65$:  .ASCIZ  <15><12>/SEEK TIME SCALE FACTOR=0.01 MILI SECS
3580  013044                        64$:
3581  013044  104401  013052                TYPE    ,67$            ;;TYPE ASCIZ STRING
3582  013050  000421                        BR      66$             ;;GET OVER THE ASCIZ
3583                                ;;67$:  .ASCIZ  <15><12><12>/ #  OF    SEEK   #  OF   SEEK/
3584  013114                        66$:
3585  013114  104401  013122                TYPE    ,69$            ;;TYPE ASCIZ STRING
3586  013120  000421                        BR      68$             ;;GET OVER THE ASCIZ
3587                                ;;69$:  .ASCIZ  <15><12>/ SEEKS    TIME   SEEKS   TIME/<15><12.
3588  013164                        68$:
3589
359C  013164  012737  001542  001260        MOV     #SIAD,INADR     ;INITLZE PTR TO INNER ADRES
```

```
3591  013172  012737  001524  001262          MOV     #SOAD,OUTADR    ;INITLZE PTR TO OUTER ADRES
3592
3593  013200  017777  166056  166256  REPTIM: MOV     @OUTADR,@RKDA   ;POSITION HEADS TO OUTER CYLINDER
3594                                          ;BEFORE STARTING TO TIME
3595  013206  053777  001230  166250          BIS     DRIVAD,@RKDA    ;SET DRIVE # BITS
3596  013214  012777  000011  166234          MOV     #11,@RKCS               ;SEEK, GO
3597  013222  104421                          CON.RDY                 ;WAIT FOR CNTRL RDY
3598  013224  104422                          TST.RWS                 ;WAIT FOR R/W/S RDY
3599
3600  013226  005037  001474                  CLR     INDX1           ;INDX1 = 0, GOING IN; OTHERWISE OUT
3601  013232  012704  027026                  MOV     #BUFR10,R4      ;STORE FWRD SEEK TIMES IN THIS BUFR
3602  013236  012705  027426                  MOV     #BUFR11,R5      ;STORE REVRSE  "  "
3603  013242  012737  177634  001476          MOV     #-144,INDX2     ;SET COUNT FOR # OF SEEKS
3604
3605
3606  013250  013702  001464          BEGSK:  MOV     RKDA,R2
3607  013254  005737  001474                  TST     INDX1           ;GOING FWRD OR REVRSE?
3608  013260  001005                          BNE     1$              ;REVRSE, BRANCH
3609
3610  013262  017712  165772                  MOV     @INADR,@R2      ;FWRD, SET INNER CYL ADRES
3611  013266  053712  001230                  BIS     DRIVAD,@R2      ;SET DRIVE # BITS
3612  013272  000404                          BR      2$
3613
3614  013274  017712  165762          1$:     MOV     @OUTADR,@R2     ;SET OUTER CYL ADRES
3615  013300  053712  001230                  BIS     DRIVAD,@R2      ;SET DRIVE # BITS
3616  013304  004737  014776          2$:     JSR     PC,@#TIMSEK     ;GO, TIME THE SEEK FROM CYLINDER
3617                                          ;0 TO THE ABOVE CYL. RETURN WITH
3618                                          ;R3 CONTAINING THE TIME (MS, SCALE
3619                                          ;FACTOR= 0.01) REQUIRED FOR THE SEEK.
3620  013310  005737  001474                  TST     INDX1
3621  013314  001004                          BNE     3$
3622  013316  010324                          MOV     R3,(R4)+        ;STORE TIME TAKEN FOR FRWRD SEEK
3623  013320  005237  001474                  INC     INDX1           ;SET FLAG FOR DOING REVRSE SEEK
3624  013324  000751                          BR      BEGSK           ;GO DO IT
3625
3626  013326  010325          3$:     MOV     R3,(R5)+        ;STORE TIME TAKEN FOR REVRSE SEEK
3627  013330  005037  001474                  CLR     INDX1           ;CLR FLG FOR DOING FRWRD SEEK
3628
3629  013334  005237  001476                  INC     INDX2           ;RECORDED 144 SEEK TIMES
3630  013340  001343                          BNE     BEGSK           ;IF NOT, GO BAK
3631
3632                                          ;AT THIS POINT 100 SEEKS HAVE BEEK PERFORMED BETWEEN TWO
3633                                          ;CYLINDERS (FORWARD & REVERSE DIRECTION).  FORWARD SEEK
3634                                          ;TIMES ARE STORED IN TABLE STARTING AT 'BUFR10' REVERSE
3635                                          ;STARTING AT 'BUFR11'.  THE FOLLOWING CODE FINDS OUT THE
3636                                          ;NUMBERS OF TIMES A PARTICULAR 'SEEK TIME' WAS OBTAINED.
3637                                          ;EXAMPLE: OUT OF 100 TIMES THE SEEK WAS DONE BETWEEN
3638                                          ;CYLINDER 0 & LAST.
3639                                          ;70 TIMES IT TOOK 95 MILI SECS
3640                                          ;20 TIMES IT TOOK 85 MILI SECS
3641                                          ;10 TIMES IT TOOK 100 MILI SECS
3642                                          ;THIS INDICATES HOW CONSISTENT WAS THE SEEKING TIME.
3643                                          ;NOTE THAT ONLY 5 DIFFERENT "SEEK TIMES" WILL BE TYPED
3644                                          ;OUT.
3645
3646                                          ;SORTING ROUTINE
```

```
3647
3648  013342  012705  177776          SORT:   MOV     #-2,R5          ;COUNT FOR FRWRD, REVRSE
3649  013346  012737  027026  001162          MOV     #BUFR10,$REG0   ;INTLZE PTR TO 'SEEK TIME'
3650  013354  012737  027030  001164          MOV     #BUFR10+2,$REG1
3651  013362  012737  026526  001166          MOV     #BUFR4,$REG2
3652  013370  012737  026606  001170          MOV     #BUFR5,$REG3    ;INTLZE PTR TO '# OF TIMES'
3653
3654  013376  013700  001162          1$:     MOV     $REG0,R0        ;PTR T 'SEEK TIME'
3655  013402  013701  001164                  MOV     $REG1,R1
3656  013406  012702  177635                  MOV     #-143,R2        ;COUNT FOR 143 ITEMS TO SOR'
3657  013412  005003                          CLR     R3
3658
3659  013414  021011                  2$:     CMP     (R0),(R1)       ;SORT THE ITEMS & PUT THEM
3660  013416  003404                          BLE     3$              ;IN DESCENDING ORDER
3661  013420  011004                          MOV     (R0),R4         ;LARGER ITEMS AT TOP OF LIS',
3662  013422  011110                          MOV     (R1),(R0)       ;SMALLER AT THE BOTTOM
3663  013424  010411                          MOV     R4,(R1)
3664  013426  005203                          INC     R3
3665  013430  005720                  3$:     TST     (R0)+
3666  013432  005721                          TST     (R1)+
3667  013434  005202                          INC     R2
3668  013436  001366                          BNE     2$
3669  013440  005703                          TST     R3              ;SORTED ALL ITEMS?
3670  013442  001355                          BNE     1$              ;IF NOT LOOP BACK
3671
3672  013444  013700  001162                  MOV     $REG0,R0        ;PTR TO 'SEEK TIME'
3673  013450  013701  001166                  MOV     $REG2,R1        ;SAVE 'SEEK TIME' HERE
3674  013454  013702  001170                  MOV     $REG3,R2        ;SAVE '# OF TIMES' HERE
3675  013460  010204                          MOV     R2,R4
3676  013462  005024                          CLR     (R4)+           ;CLR OUT 5 WORDS OF
3677  013464  005024                          CLR     (R4)+           ;'# OF TIMES'BUFR
3678  013466  005024                          CLR     (R4)+
3679  013470  005024                          CLR     (R4)+
3680  013472  005024                          CLR     (R4)+
3681  013474  012703  177773                  MOV     #-5,R3          ;SAVE ONLY 5 DIFRNT 'SEEK TIMES'
3682
3683  013500  011011                          MOV     (R0),(R1)       ;FIND OUT THE '# OF TIMES'
3684  013502  012703  177634                  MOV     #-144,R3        ;EACH 'SEEK TIME' WAS
3685  013506  022011                  4$:     CMP     (R0)+,(R1)
3686  013510  001411                          BEQ     5$              ;OBTAINED
3687
3688  013512  005721                          TST     (R1)+
3689  013514  016011  177776                  MOV     -2(R0),(R1)     ;SAVE 'SEEK TIME'
3690  013520  005722                          TST     (R2)+
3691  013522  012712  000001                  MOV     #1,(R2)         ;KEEP '# OF TIMES'
3692  013526  005203                          INC     R3
3693  013530  001404                          BEQ     6$
3694  013532  000765                          BR      4$
3695
3696  013534  005212                  5$:     INC     (R2)            ;INCRMNT '# OF TIMES'
3697  013536  005203                          INC     R3              ;ALL DONE?
3698  013540  001362                          BNE     4$              ;IF NOT, GO BAK
3699
3700  013542  005205                  6$:     INC     R5              ;SORTED BOTH FRWRD, REVRSE
3701  013544  001415                          BEQ     GOTYPE          ;'SEEK TIMES'. IF YES GO TYPE
3702
```

```
3703  013546  012737  027426  001162          MOV     #BUFR11,$REG0   ;IF NOT, INITLZE PTR TO 'SEEK TIME'
3704  013554  012737  027430  001164          MOV     #BUFR11+2,$REG1
3705  013562  012737  026666  001166          MOV     #BUFR6,$REG2    ;SAVE 'SEEK TIME'
3706  013570  012737  026746  001170          MOV     #BUFR7,$REG3    ;SAVE '# OF TIMES' HERE
3707
3708  013576  000677                          BR      1$              ;GO BAK & DO SORTING FOR REVRSE SEEK TIMES
3709
3710                                          ;TYPE OUT CYL #'S BETWEEN WHICH SEEK
3711                                          ;WAS TIMED.  'OUTADR' TO 'INADR' 'INADR' TO 'OUTADR'
3712  013600  104401                  COTYPE: TYPE
3713  013602  001213                          $CRLF
3714  013604  104401  013612                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3715  013610  000403                          BR      64$             ;;GET OVER THE ASCIZ
3716                                  ;;65$:  .ASCIZ  /CYLS:/
3717  013620                          64$:
3718
3719  013620  017700  165436                  MOV     @OUTADR,R0      ;GET OUTER CYL #
3720  013624  006200                          ASR     R0
3721  013626  006200                          ASR     R0
3722  013630  006200                          ASR     R0
3723  013632  006200                          ASR     R0
3724  013634  006200                          ASR     R0
3725  013636  010046                          MOV     R0,-(SP)
3726  013640  104424                          TYPDSS                  ;TYPE IT OUT IN DECIMAL
3727  013642  104401  013650                  TYPE    ,67$            ;;TYPE ASCIZ STRING
3728  013646  000401                          BR      66$             ;;GET OVER THE ASCIZ
3729                                  ;;67$:  .ASCIZ  /-/
3730  013652                          66$:
3731  013652  017701  165402                  MOV     @INADR,R1       ;GET INNER CYL #
3732  013656  006201                          ASR     R1
3733  013660  006201                          ASR     R1
3734  013662  006201                          ASR     R1
3735  013664  006201                          ASR     R1
3736  013666  006201                          ASR     R1
3737  013670  010146                          MOV     R1,-(SP)
3738  013672  104424                          TYPDSS                  ;TYPE IT OUT IN DECIMAL
3739  013674  104401  013702                  TYPE    ,69$            ;;TYPE ASCIZ STRING
3740  013700  000405                          BR      68$             ;;GET OVER THE ASCIZ
3741                                  ;;69$:  .ASCIZ  <15><12>/  FRWRD/
3742  013714                          68$:
3743  013714  104401  002101                  TYPE    ,BLNKS9
3744  013720  104401  013726                  TYPE    ,71$            ;;TYPE ASCIZ STRING
3745  013724  000404                          BR      70$             ;;GET OVER THE ASCIZ
3746                                  ;;71$:  .ASCIZ  /REVRSE/
3747  013736                          70$:
3748
3749                                          ;TYPE OUT THE '# OF SEEKS' & 'SEEK
3750                                          ;TIME' OBTAINED FOR EACH OF THOSE
3751                                          ;SEEKS
3752  013736  005000                  TYPTIM: CLR     R0
3753  013740  005005                          CLR     R5
3754  013742  104401                  1$:     TYPE
3755  013744  001213                          $CRLF
3756  013746  016046  026606                  MOV     BUFR5(R0),-(SP) ;GET '# OF SEEKS', IF NONE (0)
3757  013752  001424                          BEQ     3$              ;SKIP TYPING (FRWRD SEEK)
3758  013754  104405                          TYPDS                   ;GO TYPE OUT DECIMAL '# OF SEEKS'
```

```
3759  013756  104401                        TYPE
3760  013760  002110                        BLNKS2
3761  013762  016046  026526                MOV     BUFR4(R0),-(SP) ;GET 'SEEK TIME' FOR EACH OF
3762  013766  104405                        TYPDS                   ;OF THAT '# OF SEEKS'. 'GO
3763                                                                 ;TYPE OUT IN DECIMAL
3764
3765  013770  016046  026746          2$:   MOV     BUFR7(R0),-(SP) ;GET '# OF SEEKS', IF NONE ,C
3766  013774  001416                        BEQ     4$              ;SKIP TYPING (REVRSE SEEK)
3767  013776  005705                        TST     R5
3768  014000  001402                        BEQ     6$
3769  014002  104401  002075                TYPE    ,BLNK13
3770  014006  104405                  6$:   TYPDS                   ;TYPE OUT IN DECIMAL
3771  014010  104401                        TYPE
3772  014012  002110                        BLNKS2
3773  014014  016046  026666                MOV     BUFR6(R0),-(SP) ;GET 'SEEK TIME' & TYPE IT
3774  014020  104405                        TYPDS                   ;OUT IN DECIMAL
3775  014022  000406                        BR      5$
3776
3777  014024  005726                  3$:   TST     (SP)+           ;POP STACK
3778  014026  005205                        INC     R5
3779  014030  000757                        BR      2$
3780
3781  014032  005726                  4$:   TST     (SP)+           ;POP STACK
3782  014034  005705                        TST     R5
3783  014036  0C1004                        BNE     TIMDON
3784
3785  014040  005720                  5$:   TST     (R0)+           ;INCREMENT PTR TO TABLES
3786  014042  020027  000012                CMP     R0,#12          ;ALL DONE?
3787  014046  001335                        BNE     1$              ;IF NOT GO BAK
3788
3789  014050  062737  000002  001260  TIMDON: ADD     #2,INADR        ;INCRMNT POINTER TO NEXT
3790  014056  062737  000002  001262          ADD     #2,OUTADR       ;INNER & OUTER ADRES
3791  014064  005237  001252                INC     RETRY1          ;ALL DONE?
3792  014070  001406                        BEQ     PLTGRPH
3793  014072  032777  000100  165040        BIT     #SW6,@SWR       ;INHIBIT TIMER? FURTHER ?
3794  014100  001402                        BEQ     PLTGRPH         ;YES, BRANCH
3795  014102  000137  013200                JMP     REPTIM          ;GO, BACK AND TIME REST
3796                                                                 ;OF SEEKS
3797
3798
3799
3800                                  ;PLOT GRAPH OF 'SEEK TIME' V/S 'CYLIDERS SEEKED'
3801
3802                                          ;PERFORM SEEK FROM CYLINDER 0 TO EVERY OTHER CYLINDER AND TIME IT.
3803                                          ;0      0,0     1,----0 312. NOTE 'SECTOR COUNTER' IS USED AS A READ
3804                                          ;TIME CLOCK TO TIME THERE SEEKS.  AFTER OBTAINING THE SEEK TIMES A
3805                                          ;GRAPH IS PLOTTED OF 'SEEK TIME' V/S 'CYLINDER'.
3806
3807                                          ;TIME THE SEEKS
3808  014106  032777  000040  165024  PLTGRPH: BIT #SW5,@SWR         ;SKIP THE GRAPH?
3809  014114  001002                        BNE     .+6
3810  014116  000137  015224                JMP     TST12           ;YES, BRANCH
3811  014122  104415                        CON.RESET
3812  014124  104416                        DRV.RESET
3813  014126  012737  177465  001500        MOV     #-312,INDX3     ;PERFORM 313 SEEKS 0-0,0-1.0-312
3814  014134  012704  027026                MOV     #BUFR10,R4      ;STORE 'SEEK TIME' HERE
```

```
3815  014140  005037  001260                  CLR     INADR           ;CLR CYL ADRES BITS
3816
3817  014144  013777  001260  165312  1$:     MOV     INADR,@RKDA     ;ADRES THE RIGHT CYLINDER
3818  014152  053777  001230  165304          BIS     DRIVAD,@RKDA    ;ADRES THE RIGHT DRIVE
3819
3820  014160  004737  014776                  JSR     PC,TIMSEK       ;GO TIME THE SEEK FROM CYL 0
3821                                                                  ;TO THE ABOVE CYL. RETURN WITH
3822                                                                  ;R3 CONTAINING 'SEEK TIME' IN MS
3823                                                                  ;SCALE FACTOR OF 0.01
3824  014164  010324                          MOV     R3,(R4)+        ;STORE 'SEEK TIME'
3825  014166  042777  017777  165270          BIC     #17777,@RKDA            ;SEEK BACK TO CYL 0 FOR
3826  014174  012777  000011  165254          MOV     #11,@RKCS       ;TIMING NXT CYL SEEK
3827  014202  104421                          CON.RDY                 ;WAIT FOR CNTRL RDY?
3828  014204  104422                          TST.RWS                 ;WAIT FOR R/W/S RDY
3829  014206  062737  000040  001260          ADD     #40,INADR       ;FORM NXT CYL ADRES
3830  014214  005237  001500                  INC     INDX3
3831  014220  001351                          BNE     1$
3832
3833                                          ;PLOT A GRAPH USING 'SEEK TIMES' RECORDED BEFORE
3834
3835  014222                          PLOT:
3836  014222  104401  014230                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3837  014226  000422                          BR      64$             ;;GET OVER THE ASCIZ
3838                                  ;;65$:  .ASCIZ  <15><12><12><12>/X AXIS - SEEK TIME - MILI SECS/
3839  014274                          64$:
3840  014274  104401  014302                  TYPE    ,67$            ;;TYPE ASCIZ STRING
3841  014300  000423                          BR      66$             ;;GET OVER THE ASCIZ
3842                                  ;;67$:  .ASCIZ  <15><12>/Y AXIS - CYLINDER SEEKED FROM 0/<15><12><12>
3843  014350                          66$:
3844
3845  014350  104401                          TYPE
3846  014352  002103                          BLNKS7
3847  014354  005000                          CLR     R0              ;TYPE OUT THE TIME UNITS
3848  014356  010046                  1$:     MOV     R0,-(SP)        ;(MILI SECS) FOR THE X-AXIS
3849  014360  104424                          TYPDSS                  ;LIKE THIS:
3850  014362  005700                          TST     R0              ;0   20   30   40.....
3851  014364  001411                          BEQ     2$
3852  014366  022700  000144                  CMP     #144,R0
3853  014372  003010                          BGT     4$
3854  014374  022700  000170                  CMP     #170,R0
3855  014400  002412                          BLT     5$
3856  014402  104401                          TYPE
3857  014404  002110                          BLNKS2
3858  014406  000404                          BR      3$
3859  014410  104401                  2$:     TYPE
3860  014412  002111                          BLNKS1
3861  014414  104401                  4$:     TYPE
3862  014416  002107                          BLNKS3
3863  014420  062700  000012          3$:     ADD     #12,R0
3864  014424  000754                          BR      1$
3865
3866  014426  104401                  5$:     TYPE
3867  014430  001213                          $CRLF
3868  014432  104401                          TYPE
3869  014434  002103                          BLNKS7
3870
```

```
3871  014436  012700  177763          PLT1:   MOV     #-15,R0         ;TYPE OUT THE X-AXIS MARKERS
3872  014442                          1$:
3873  014442  104401  014450                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3874  014446  000403                          BR      64$             ;;GET OVER THE ASCIZ
3875                                  ;;65$:  .ASCIZ  /I----/
3876  014456                          64$:
3877  014456  005200                          INC     R0              ;I----I----I----
3878  014460  001370                          BNE     1$
3879
3880                                  ;IF SW 4 IS SET THEN TYPE THE COMPLETE GRAPH. IF NOT TYPE THE SMALL GRAPH.
3881
3882  014462  032777  000020  164450          BIT     #SW4,@SWR       ;TYPE COMPLETE GRAPH?
3883  014470  001054                          BNE     CMPGRP          ;YES BRANCH
3884                                                                  ;IF NOT, TYPE SMALL GRAPH
3885
3886
3887
3888  014472  005000                  SMGRP:  CLR     R0
3889  014474  032777  000040  164436  1$:     BIT     #SW5,@SWR       ;SKIP REST OF GRAPH?
3890  014502  001445                          BEQ     5$              ;YES
3891  014504  10440.                          TYPE                    ;IN THIS GRAPH SEEK TIMES ARE
3892  014506  001213                          $CRLF           ;PLOTTED ONLY FOR SELECTED
3893                                                        ;CYLINDERS (NOT ALL) SHOWN BELOW:
3894                                                        ;0,1,2,3,4,  6,8,10,12,14,16,18,20,
3895                                                        ;25,30,35,...,190,195,200,  203
3896  014510  010046                          MOV     R0,-(SP)                ;TYPE THE MARKERS
3897  014512  104405                          TYPDS
3898  014514  104401  014522                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3899  014520  000401                          BR      64$             ;;GET OVER THE ASCIZ
3900                                  ;;65$:  .ASCIZ  /-/
3901  014524                          64$:
3902  014524  010001                          MOV     R0,R1           ;FORM THE ADRES OF 'SEEK TIME'
3903  014526  006301                          ASL     R1
3904  014530  016103  027026                  MOV     BUFR10(R1),R3   ;GET THE SEEK TIME
3905  014534  004737  014736                  JSR     PC,PLTPT                ;GO PLOT IT
3906  014540  022700  000004                  CMP     #4,R0           ;PLOTTED UPTO CYL 4?
3907  014544  003402                          BLE     2$              ;YES
3908  014546  005200                          INC     R0
3909  014550  000751                          BR      1$
3910  014552  022700  000024          2$:     CMP     #24,R0  ;PLOTTED UPTO CYL 20?
3911  014556  003403                          BLE     3$
3912  014560  062700  000002                  ADD     #2,R0
3913  014564  000743                          BR      1$
3914  014566  022700  000310          3$:     CMP     #310,R0         ;PLOTTED UPTO CYL 200?
3915  014572  003403                          BLE     4$
3916  014574  062700  000005                  ADD     #5,R0
3917  014600  000735                          BR      1$
3918  014602  022700  000312          4$:     CMP     #312,R0         ;PLOTTED ALL CYLS?
3919  014606  001403                          BEQ     5$
3920  014610  062700  000002                  ADD     #2,R0
3921  014614  000727                          BR      1$
3922  014616  000137  015224          5$:     JMP     TST12
3923
3924
3925                                  ;IF SW 4 IS SET THE COMPLETE GRAPH IS PRINTED OUT. IT GIVES TIMES FOR
3926                                  ;SEEKS FROM CYLINDER 0 TO ALL OTHER CYLINDERS (0,1,2,3...202).
```

```
3927
3928  014622  005000                  CMPGRP: CLR     R0              ;INITLZE COUNT
3929  014624  012701  177773                  MOV     #-5,R1          ;INITLZE COUNT FOR Y-AXIS MARKER
3930  014630  012702  027026                  MOV     #BUFR10,R2      ;INITLZE PTR TO SEEK TIMES
3931  014634  104401                          TYPE
3932  014636  001213                          $CRLF
3933  014640  000412                          BR      3$
3934
3935  014642  032777  000040  164270  2$:     BIT     #SW5,@SWR               ;SKIP REST OF GRAPH?
3936  014650  001002                          BNE     .+6
3937  014652  000137  015224                  JMP     TST12
3938  014656  005201                          INC     R1              ;TYPE OUT Y-AXIS MARKER 'CYL #'
3939  014660  001005                          BNE     4$              ;IF REQUIRED
3940  014662  012701  177773                  MOV     #-5,R1
3941  014666  010046                  3$:     MOV     R0,-(SP)        ;TYPE 'CYL #' ON Y-AXIS
3942  014670  104405                          TYPDS                   ;(IN DECIMAL)
3943  014672  000402                          BR      5$
3944  014674  104401                  4$:     TYPE
3945  014676  002104                          BLNKS6
3946  014700                          5$:
3947  014700  104401  014706                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3948  014704  000401                          BR      64$             ;;GET OVER THE ASCIZ
3949                                  ;;65$:  .ASCIZ  /-/
3950  014710                          64$:
3951
3952  014710  012203                          MOV     (R2)+,R3                ;GET SEEK TIME
3953  014712  004737  014736                  JSR     PC,PLTPT        ;GO PLOT THE POINT
3954
3955
3956  014716  104401                          TYPE
3957  014720  001213                          $CRLF
3958  014722  005200                          INC     R0              ;ALL DONE?
3959  014724  022700  000312                  CMP     #312,R0
3960  014730  001344                          BNE     2$              ;IF NOT, GO BAK
3961  014732  000137  015224          6$:     JMP     TST12
3962
3963                                  ;PLTPT
3964                                  ;THE ROUTINE IS ENTERED WITH R3 CONTAINING HORIZONTAL AXIS
3965                                  ;COORDINATE- SEEK TIME
3966                                  ;PLOT THE ACTUAL TIME ON THE GRAPH.  IN KEEPING WITH NORMAL
3967                                  ;CONVENTION A NUMBER LESS THAN HALF THE CELL WIDTH IS
3968                                  ;CONSIDERED AS FALLING UNDER THE PREVIOUS CELL, A NUMBER
3969                                  ;GREATER THAN OR EQUAL TO HALF THE CELL WIDTH FALLS UNDER THE NEXT CELL
3970                                  ;EX: IF SEEK TIME IS 11.5 MS, IT'S BETW'N 10 & 12, BUT > 11
3971                                  ;HENCE IT WILL BE PLOTTED AS 12 IF SEEK TIME IS 10.8 MS,
3972                                  ;IT'S BETW'N 10 & 12, BUT < 11 HENCE IT WILL BE PLOTTED
3973                                  ;AS 10.0 MS
3974
3975  014736  162703  000310          PLTPT:  SUB     #310,R3         ;FIND OUT HOW MANY BLANKS TO
3976  014742  002403                          BLT     7$              ;INSERT TO PLOT THE POINT
3977                                                                  ;NOTE THE FIRST CELL = 0 MS
3978  014744  104401                          TYPE
3979  014746  002111                          BLNKS1
3980  014750  000772                          BR      PLTPT
3981  014752  062703  000144          7$:     ADD     #144,R3
3982  014756  002402                          BLT     8$
```

```
                                        TYPE
                                        BLNKS1

                                8$:
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /X/
                                64$:
                                        RTS     PC

                                ;TIMSEK
                                ;THIS ROUTINE FINDS OUT THE TIME REQUIRED TO SEEK TO THE CYLINDER
                                ;INDICATED IN RKDA.
                                ;***CAUTION*** SECTOR COUNTER IS USED AS A REAL TIME CLOCK TO KEEP TIME.
                                ;ENTRY: JSR     PC,TIMSEK
                                ;       RKDA CONTAINS THE CYLINDER TO BE SEEKED FROM THE PRESENT POSITION.
                                ;RETURN:        R3 CONTAINS THE SEEK TIME IN MILI SECS. SCALE FACTOR = 0.01


                                                                        ;R3 WILL COUNT REVOLUTIONS OF
                                TIMSEK: MOV     R2,-(SP)                ;DISK (FROM INDEX MARK TO INDEX MARK.
                                        CLR     R3                      ;40 MILI SECS FOR EACH REV
                                        MOV     RKDS,R1
                                1$:     MOV     @R1,R2
                                        BIT     #400,R2                 ;WAIT FOR SOK
                                        BEQ     1$

                                        BIT     #BIT3,R2                        ;WAIT FOR SECTOR 10 SO THAT
                                        BEQ     1$                      ;U CAN START WAITING FOR
                                                                        ;INDEX, SEC 0

                                2$:     MOV     @R1,R2
                                        BIT     #400,R2                 ;WAIT FOR SEC OK
                                        BEQ     2$
                                        CMP     @R1,R2
                                        BNE     2$
                                        BIT     #17,R2                          ;WAIT FOR SEC 0, INDEX MARK
                                        BNE     2$                      ;AS SOON AS IT IS SEC 0, ISSUE
                                                                        ;A SEEK & START TIMING
                                                                        ;
                                        MOV     #11,@RKCS               ;ISSUE A SEEK, START TIMING
                                                                        ;THE SEC COUNTER
                                        CON.RDY                         ;WAIT FOR CNTRL RDY

                                3$:     MOV     @R1,R2                  ;GET RKDS
                                        BIT     #400,R2                 ;WAIT FOR SOK
                                        BEQ     3$
                                        CMP     R2,@R1                  ;INFO CORRECT?
                                        BNE     3$                      ;NO
                                        BIT     #100,R2                 ;R/W/S RDY SET?
                                        BNE     SKDON                   ;IF YES, BRANCH
                                        BIT     #17,R2                  ;WAIT FOR SEC CNTR TO MOVE
                                        BEQ     3$                      ;FROM 0 TO 1

                                4$:     MOV     @R1,R2
                                        BIT     #400,R2                 ;WAIT FOR SOK
```

```
4039  015114  001774                        BEQ     4$
4040  015116  020211                        CMP     R2,@R1
4041  015120  001372                        BNE     4$
4042  015122  032702  000100                BIT     #100,R2         ;R/W/S RDY SET, SEEK DONE?
4043  015126  001005                        BNE     5$              ;YES, BRANCH
4044  015130  032702  000017                BIT     #17,R2                  ;IF NOT KEEP TRACK OF SEC
4045  015134  001364                        BNE     4$              ;COUNTER. INCREMENT R3 AT
4046                                                                ;EVERY INDEX MARK,EVERY
4047  015136  005203                        INC     R3              ;40 MILI SECS
4048  015140  000746                        BR      3$              ;GO BAK, KEEP TIME
4049
4050  015142  032702  000017          5$:   BIT     #17,R2          ;CHECK, IS IT INDEX MARK -SEC 0
4051  015146  001001                        BNE     SKDON           ;IF NOT, SKIP
4052  015150  005203                        INC     R3              ;IF YES, INCREMENT COUNT
4053
4054                                                                        ;SEEK DONE, SAVE RKDS-SEC COUNTER.
4055  015152                          SKDON:
4056  015152  012746  000014                MOV     #14,-(SP)               ;;PUT THE MULTIPLER ON THE STACK
4057  015156  010346                        MOV     R3,-(SP)                ;;PUT THE MULTIPLICAND ON THE STACK
4058  015160  004737  020500                JSR     PC,@#$MULT      ;;CALL THE MULTPLY ROUTINE
4059  015164  012616                        MOV     (SP)+,(SP)      ;;DISREGARD THE MSB'S
4060  015166  012603                        MOV     (SP)+,R3                ;;GET THE LSB'S OF THE PRODUCT
4061  015170  042702  177760                BIC     #177760,R2      ;SEEK. TOTAL TIME=(IN DECIMAL)
4062  015174  060203                        ADD     R2,R3           ;[(R3)X12+SEC COUNTER]X33.0X0.01
4063                                                                ;NOTE THERE IS A SCALE FACTOR
4064  015176  012746  000512                MOV     #512,-(SP)              ;;PUT THE MULTIPLER ON THE STACK
4065  015202  010346                        MOV     R3,-(SP)                ;;PUT THE MULTIPLICAND ON THE STACK
4066  015204  004737  020500                JSR     PC,@#$MULT      ;;CALL THE MULTPLY ROUTINE
4067  015210  012616                        MOV     (SP)+,(SP)      ;;DISREGARD THE MSB'S
4068  015212  012603                        MOV     (SP)+,R3                ;;GET THE LSB'S OF THE PRODUCT
4069  015214  062703  000245                ADD     #245,R3         ;ASSUMPTION THAT EACH SECTOR
4070                                                                ;TAKES 3.3 MILI SECS. IF THE
4071                                                                ;DISK SPEED IS VERY MUCH DIFRNT
4072                                                                ;FROM THE SPEC SPEED OF
4073                                                                ;1500 RPM (40 MS/REV), THEN
4074                                                                ;SEC COUNTER WOULD NOT BE AN
4075                                                                ;ACCURATE TIME CLOCK.
4076  015220  012602                        MOV     (SP)+,R2        ;POP R2 BAK
4077  015222  000207                        RTS     PC              ;RETURN
4078
4079
4080                                  ;;*****************************************************************
4081                                  ;*TEST 12        END OF PROGRAM
4082                                          ;*THIS IS NOT A TEST BUT IS JUST A LINKAGE
4083                                          ;*PROVIDED TO TEST ALL THE DRIVES.
4084                                  ;;*****************************************************************
4085  015224  000004                  TST12:  SCOPE
4086  015226  105237  001223                  INCB    DRVDON
4087  015232  123737  001223  001224  BTEOP:  CMPB    DRVDON,DRIVS
4088  015240  001402                        BEQ     .+6
4089  015242  000137  003760                JMP     NXTDRV
4090
4091                                  .SBTTL  END OF PASS ROUTINE
4092
4093                                  ;;*****************************************************************
4094                                  ;*INCREMENT THE PASS NUMBER ($PASS)
```

```
4095                                    ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
4096                                    ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER
4097                                    ;*IF THERES A MONITOR GO TO IT
4098                                    ;*IF THERE ISN'T JUMP TO ST3
4099
4100  015246                            $EOP:
4101  015246  000004                           SCOPE
4102  015250  005037  001102                   CLR     $TSTNM          ;;ZERO THE TEST NUMBER
4103  015254  005237  001100                   INC     $PASS           ;;INCREMENT THE PASS NUMBER
4104  015260  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
4105  015266  005327                           DEC     (PC)+           ;;LOOP?
4106  015270  000001                    $EOPCT: .WORD   1
4107  015272  003022                           BGT     $DOAGN          ;;YES
4108  015274  012737                           MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
4109  015276  000001                    $ENDCT: .WORD   1
4110  015300  015270                           $EOPCT
4111  015302  104401  015347                   TYPE    ,$ENDMG         ;;TYPE "END PASS #"
4112  015306  013746  001100                   MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
4113  015312  104405                           TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
4114  015314  104401  015344                   TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
4115  015320  013700  000042            $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
4116  015324  001405                           BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
4117  015326  000005                           RESET                   ;;CLEAR THE WORLD
4118  015330  004710                    $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
4119  015332  000240                           NOP                     ;;SAVE ROOM
4120  015334  000240                           NOP                     ;;FOR
4121  015336  000240                           NOP                     ;;ACT11
4122  015340                            $DOAGN:
4123  015340  000137                           JMP     @(PC)+          ;;RETURN
4124  015342  003742                    $RTNAD: .WORD   ST3
4125  015344     377       377       000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
4126  015347     015  042412  042116    $ENDMG: .ASCIZ  <15><12>/END PASS #/
4127  015354  050040  051501  020123
4128  015362  000043
4129
```

```
                                        END OF PASS ROUTINE

                                        ;COMMON SUBROUTINES AND HANDLERS



                                                .SBTTL  ESR15
                                        ;ESR15
                                        ;THIS ROUTINE IS USED TO TYPE OUT ERROR DATA FOR ITEM 15
                                        ;OF THE ERROR TABLE.  AT THE TIME OF ENTRY INTO THIS
                                        ;ROUTINE R5 CONTAINS THE DISK ADDRESS FROM WHICH THE 12
                                        ;HEADERS WERE READ, THE SECTOR #'S WHICH GAVE BAD HEADERS HAVE
                                        ;BEEN STORED STARTING AT 'BUFR'.  THE CORRESPONDING BAD HEADERS
                                        ;HAVE BEEN STORED STARTING AT 'BUFR1'.

                                        ;THE PRINTOUT LOOKS LIKE:
                                        ;SEC#    HDR RECVD
                                        ;AA      BBBBBB          AA=BAD SEC #  BBBBBB=BAD HEADER
                                        ;
                                        ;EXPCTD HDR=XXXXXX      TRY#=  Y

015364                                  ESR15:
015364  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
015366  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
015370  012701  001266                          MOV     #BUFR,R1        ;SEC #'S STORED HERE PREVIOUSLY
015374  012702  001320                          MOV     #BUFR1,R2       ;BAD HDRS STORED HERE PRVSLY
015400  012146                          1$:     MOV     (R1)+,-(SP)
015402  104403                                  TYPOS                   ;GO TYPE OUT BAD SEC # (OCTAL)
015404     002                                  .BYTE   2               ;ONLY 2 DIGITS
015405     000                                  .BYTE   0               ;SUPRES LDG 0'S
015406  104401                                  TYPE                    ;TYPE 3 BLNKS
015410  002107                                  BLNKS3
015412  012246                                  MOV     (R2)+,-(SP)     ;GO TYPE OUT BAD HEADER
015414  104402                                  TYPOC
015416  104401                                  TYPE
015420  002106                                  BLNKS4
015422  104401                                  TYPE
015424  001213                                  $CRLF
015426  022711  177777                         CMP     #177777,(R1)    ;ALL BAD SEC #'S TYPD OUT?
015432  001362                                  BNE     1$              ;IF NOT GO BAK

015434  104401                                  TYPE
015436  001716                                  MSG6
015440  010546                                  MOV     R5,-(SP)        ;TYPE OUT EXPCTD HEADER FOR
015442  042716  160037                         BIC     #160037,(SP)
015446  104402                                  TYPOC                   ;THAT CYLINDER

015450  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
015452  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
015454  000207                                  RTS     PC


                                                .SBTTL  ESR13
                                        ;ESR13
                                        ;THIS ROUTINE IS USED WITH 'ERROR 13"' TO TYPEOUT OUT ERROR
                                        ;DATA.  THE SECTOR #'S WHICH GAVE BAD HEADERS HAVE BEEN STORED
                                        ;STARTING AT 'BUFR'.  THE CORRESPONDING BAD HEADERS HAVE
                                        ;BEEN STORED STARTING AT 'BUFR1'.  R5 CONTAINS THE EXPECTED
```

```
                                        ;HEADER FOR THAT CYLINDER.  THE TYPEOUT LOOKS LIKE

                                        ;SEC#  HDR RCVD
                                        ;AA     BBBBBB          AA=BAD SEC #
                                        ;                       BBBBBB=BAD HEADER
                                        ;
                                        ;EXPCTD HDR=XXXXXX      TRY#:  Y        SUR=Z

015456  004737  015364        ESR13:  JSR     PC,ESR15                        '
015462  104401  015470                TYPE    ,65$                    ;;TYPE ASCIZ STRING
015466  000404                        BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /  SUR=/
                                64$:
015500  005046                        CLR     -(SP)
015502  032705  000020                BIT     #20,R5                  ;SUR 0 OR 11?
015506  001401                        BEQ     1$
015510  005216                        INC     (SP)
015512  104402                1$:     TYPOC

015514  104401  002053                TYPE    ,MSG13
015520  013746  001254                MOV     RETRY2,-(SP)
015524  005216                        INC     (SP)
015526  104402                        TYPOC
015530  000207                        RTS     PC


                                        .SBTTL  ESR20
                                ;ESR20
                                ;SUBROUTINE TO TYPE OUT ERROR DATA FOR 'ERROR 20'.  AT THE TIME
                                ;OF ENTRY, TABLE STARTING AT 'BUFR' CONTAINS SECTOR #'S THAT GAVE BAD
                                ;HEADERS.  TABLE AT 'BUFR1' CONTAINS BAD HEADERS, R5 CONTAINS EXPECTED
                                ;HEADER FOR THE CYLINDER.  'INADR' AND 'OUTADR' CONTAIN THE CYLINDER
                                ;ADDRESSES BETWEEN WHICH THE IMPLIED SEEK WAS TRIED.

015532  004737  015456        ESR20:  JSR     PC,ESR13                ;GO TYPE OUT SEC #'S, BAD HDRS
015536  004737  016320                JSR     PC,ERR2                 ;GET CYL #'S BETWN WHICH SEEK
                                                                        ;WAS TRIED
015542  104401  015550                TYPE    ,65$                    ;;TYPE ASCIZ STRING
015546  000404                        BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /  CYLA=/
                                64$:
015560
015560  013746  001162                MOV     $REG0,-(SP)             ;GO TYPE CYL # FROM WHERE
015564  104403                        TYPOS                           ;SEEK BEGAN
015566     003                      .BYTE   3                       ;TYPE 3 DIGITS
015567     000                      .BYTE   0                       ;SUPRES LDG 0'S
015570  104401  015576                TYPE    ,67$                    ;;TYPE ASCIZ STRING
015574  000404                        BR      66$                     ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /  CYLB=/
                                66$:
015606
015606  013746  001164                MOV     $REG1,-(SP)             ;TYPE CYL # TO WHICH SEEK
015612  104403                        TYPOS                           ;WAS DONE
015614     003                      .BYTE   3                       ;TYPE 3 DIGITS
015615     000                      .BYTE   0                       ;SUPRES LDG 0'S
015616  000207                        RTS     PC                      ;RETURN


                                        .SBTTL  ESR25
```

```
4242  015620  010205                    ESR2S:  MOV     R2,R5           ;SAVE ADRES OF TERMINATOR
4243
4244  015622  012702  001266                    MOV     #BUFR,R2        ;INITLZE PTR TO TABLE STORING
4245                                                                      ;ADRES OF BAD DATA
4246  015626  012703  001320                    MOV     #BUFR1,R3       ;INITLZE PTR TO 'EXPCTD' DATA
4247  015632  012704  001352                    MOV     #BUFR2,R4       ;INITLZE PTR TO 'RECVD' DATA
4248
4249  015636  032777  020000  163274  1$:       BIT     #SW13,@SWR      ;INHIBIT TYPE OUT?
4250  015644  001076                            BNE     4$              ;YES, EXIT
4251  015646  104401                            TYPE                    ;TYPE CR,LF
4252  015650  001213                            $CRLF
4253
4254  015652  163712  001404                    SUB     PBUFO,(R2)      ;GET WORD # IN BUFR (0,1,2...
4255  015656  006212                            ASR     (R2)
4256  015660  011246                            MOV     (R2),-(SP)      ;WHICH WAS BAD.  NOTE YOU
4257                                                                      ;CAN HAVE THE ACTUAL MEMORY
4258                                                                      ;ADRES BY ADDING 'IOBUFO'
4259                                                                      ;TO THIS
4260  015662  104403                            TYPOS                   ;GO TYPE WORD # THAT WAS BAD
4261  015664     004                            .BYTE   4
4262  015665     000                            .BYTE   0
4263  015666  104401                            TYPE
4264  015670  002107                            BLNKS3                  ;2 BLANKS
4265
4266  015672  012346                            MOV     (R3)+,-(SP)     ;GET EXPCTD DATA
4267  015674  104402                            TYPOC                   ;GO TYPE IT
4268  015676  104401                            TYPE
4269  015700  002110                            BLNKS2
4270  015702  012446                            MOV     (R4)+,-(SP)     ;GET RECVD DATA (BAD)
4271  015704  104402                            TYPOC                   ;GO TYPE IT
4272  015706  104401                            TYPE
4273  015710  002110                            BLNKS2
4274
4275  015712  012700  000400                    MOV     #400,R0         ;GET THE DISK ADRES FROM
4276  015716  021200                    2$:     CMP     (R2),R0         ;WHICH THIS (BAD) DATA WAS
4277  015720  002405                            BLT     3$              ;READ
4278  015722  062700  000400                    ADD     #400,R0
4279  015726  022700  002400                    CMP     #2400,R0
4280  015732  001371                            BNE     2$
4281
4282  015734  000300                    3$:     SWAB    R0
4283  015736  005300                            DEC     R0
4284  015740  063700  001450                    ADD     ADRES,R0        ;R0 CONTAINS THE DISK
4285                                                                      ;ADRES FROM WHICH THE (BAD,
4286  015744  010037  001170                    MOV     R0,$REG3        ;DATA WAS READ
4287
4288  015750  004737  016220                    JSR     PC,BRKDA        ;GO BREAK ABOVE DISK ADRES
4289                                                                      ;INTO CYL#, SUR#, SEC#
4290
4291  015754  013746  001174                    MOV     $REG5,-(SP)     ;GET THE CYL#
4292  015760  104403                            TYPOS                   ;TYPE IT
4293  015762     003                            .BYTE   3               ;ONLY 3 DIGITS
4294  015763     000                            .BYTE   0               ;NO LEADING 0'S
4295  015764  104401                            TYPE
4296  015766  002107                            BLNKS3
4297
```

```
 4298  015770  013746  001176                MOV     $REG6,-(SP)     ;GET SUR #
 4299  015774  104403                        TYPOS                   ;TYPE
 4300  015776     001                        .BYTE   1               ;1 DIGIT ONLY
 4301  015777     000                        .BYTE   0
 4302
 4303  016000  104401                        TYPE
 4304  016002  002106                        BLNKS4
 4305
 4306  016004  013746  001200                MOV     $REG7,-(SP)     ;GET SEC#
 4307  016010  104403                        TYPOS                   ;TYPE
 4308  016012     002                        .BYTE   2               ;2 DIGITS
 4309  016013     000                        .BYTE   0
 4310
 4311  016014  005722                        TST     (R2)+           ;INCREMNT PTR
 4312  016016  020205                        CMP     R2,R5           ;TYPED OUT ALL BAD DATA
 4313                                                                ;INFO?
 4314  016020  001306                        BNE     1$              ;IF NOT LUP BAK
 4315  016022  104401                        TYPE
 4316  016024  002053                        MSG13                   ;'  TRY #:'
 4317  016026  013746  001254                MOV     RETRY2,-(SP)    ;GET RETRY COUNT
 4318  016032  062716  000003                ADD     #3,(SP)         ;FORM THE RETRY NO.
 4319  016036  104403                        TYPOS                   ;TYPE IT OUT
 4320  016040     001                        .BYTE   1
 4321  016041     000                        .BYTE   0
 4322
 4323  016042  000207                4$:     RTS     PC              ;IF YES, RETURN
 4324                                ;MESSAGE HANDLER
 4325                                ;THE MESSAGE HANDLER IS USED FOR TYPING OUT MESSAGES & DATA
 4326                                ;RELATED TO THE MESSAGE. IF SW13 IS SET, THE TYPEOUT IS
 4327                                ;INHIBITED. THE CALL IS:
 4328                                ;       MESAGE  XX
 4329                                ;XXX IS THE MESSAGE NUMBER & PROVIDES AN INDEX TO THE
 4330                                ;'ERROR ITEMS TABLE' WHERE THAT MESAGE ITEM
 4331                                ;IS LOCATED.
 4332                                ;THE MESAGE ITEM CONTAINS:
 4333                                ;       MS:     POINTER TO THE ASCII MESAGE
 4334                                ;       DH:     POINTER TO THE DATA HEADER
 4335                                ;       DT:     POINTER TO THE DATA
 4336                                ;       0       TERMINATOR
 4337                                ;IF 'DT' IS 0 THE DATA IS TO BE PRINTED USING THE SUBROUTINE
 4338                                ;INDICATED IN PLACE OF THE TERMINATOR
 4339
 4340  016044  032777  020000  163066 MSGE:   BIT     #SW13,@SWR      ;INHIBIT TYPEOUT?
 4341  016052  001012                        BNE     1$              ;IF YES, EXIT
 4342  016054  011637  001116                MOV     (SP),$ERRPC     ;GET ADRES OF 'MESAGE' CALL
 4343  016060  162737  000002  001116        SUB     #2,$ERRPC       ;STORE IT
 4344  016066  117637  000000  001114        MOVB    @(SP),$ITEMB    ;GET MESAGE # (INDEX TO ITEM TABLE,
 4345  016074  004737  017446                JSR     PC,@#ERRTYP     ;GO TO 'ERRTYP' & TYPE OUT
 4346                                                                ;INFO
 4347  016100  062716  000002          1$:   ADD     #2,(SP)         ;ADJUST RETURN ADDRES
 4348  016104  000002                        RTI                     ;EXIT
 4349
 4350                                ;THIS ROUTINE IS USED FOR TYPING OUT ASCII MESSAGES. BEFORE
 4351                                ;THE MESSAGE IS TYPED SW13 IS CHECKED & IF SET THE
 4352                                ;TYPEOUT IS INHIBITED & AN EXIT IS MADE.
 4353                                ;THE CALL FOR THIS ROUTINE IS "TYPMSG", AN ENCODED
```

```
                                        ;TRAP INSTRUCTION.
                                        ;THE POINTER TO THE ASCII MESSAGE TO BE TYPED IS LOCATED IN  THE
                                        ;WORD FOLLOWING THE "TYPMSG" CALL.

016106  032777  020000  163024  TY.MSG: BIT     #SW13,@SWR      ;INHIBIT TYPEOUT?
016114  001005                  BNE     2$              ;YES, EXIT
016116  017537  000000  016126          MOV     @(SP),1$        ;GET POINTER TO ASCII MESSAGE
016124  104401                  TYPE                    ;GO TYPE ASCII STRING
016126  000000          1$:     0
016130  062716  000002          2$:     ADD     #2,(SP)         ;ADJUST RETURN ADRES, SKIP OVER
                                                                ;POINTER ON RETURN
016134  000002                  RTI                     ;EXIT


                                        ;GTSRG
                                        ;THIS ROUTINE EXTRACTS THE CYLINDER # FROM RKDA AND STORES IT
                                        ;IN $REG4. THEN TRANSFERS RKCS, ER, DS, DA TO $REG0, $REG1, $REG2, $REG3
016136  017746  163322          GTSRG:  MOV     @RKDA,-(SP)     ;PUSH RKDA ONTO STACK
016142  042716  160037                  BIC     #160037,(SP)    ;MASK OUT NON-CYLINDER BITS
016146  006316                  ASL     (SP)            ;SHIFT 8 BITS OF CYL ADRES TO LO BYTE
016150  006316                  ASL     (SP)
016152  006316                  ASL     (SP)
016154  000316                  SWAB    (SP)
016156  112637  001172                  MOVB    (SP)+,$REG4     ;UP STACK


                                        ;GT4RG
                                        ;THIS ROUTINE TRANSFERS THE CONTENTS OF RKCS, RKER, RKDS
                                        ;RKDA TO $REG0, $REG1, $REG2, $REG3 RESPECTIVELY. $REG'S
                                        ;ARE USED FOR TYPING OUT THERE CONTENTS AT THE TIME OF ERROR

016162  017737  163270  001162  GT4RG:  MOV     @RKCS,$REG0     ;GET RKCS
016170  017737  163260  001164          MOV     @RKER,$REG1     ;    RKER
016176  017737  163250  001166          MOV     @RKDS,$REG2     ;    RKDS
016204  017737  163254  001170          MOV     @RKDA,$REG3     ;    RKDA
016212  000207                  RTS     PC              ;EXIT FROM THIS ROUTINE

                                        ;GETINF
                                        ;THIS ROUTINE SAVES THE CONTENTS OF RKCS IN $REG0
                                        ;RKER IN $REG1, RKDS IN $REG2.  THEN IT BREAKS RKDA
                                        ;INTO DRIVE NO, CYLINDER #, SURFACE AND SECTOR #.
                                        ;AND SAVES THEM IN $REG4, $REG5, $REG6, $REG7.

016214  004737  016162          GETINF: JSR     PC,GT4RG
016220  010046          BRKDA:  MOV     R0,-(SP)
016222  010146                  MOV     R1,-(SP)
016224  010246                  MOV     R2,-(SP)
016226  012700  001202                  MOV     #$REG7+2,R0
016232  013701  001170                  MOV     $REG3,R1
016236  010102                  MOV     R1,R2
016240  042702  177760                  BIC     #177760,R2
016244  010240                  MOV     R2,-(R0)
016246  006201                  ASR     R1
```

```
 4410  016250  006201                          ASR     R1
 4411  016252  006201                          ASR     R1
 4412  016254  006201                          ASR     R1
 4413  016256  010102                          MOV     R1,R2
 4414  016260  042702  177776                  BIC     #177776,R2
 4415  016264  010240                          MOV     R2,-(R0)
 4416  016266  006201                          ASR     R1
 4417  016270  010102                          MOV     R1,R2
 4418  016272  042702  177400                  BIC     #177400,R2
 4419  016276  010240                          MOV     R2,-(R0)
 4420  016300  000301                          SWAB    R1
 4421  016302  042701  177770                  BIC     #177770,R1
 4422  016306  010140                          MOV     R1,-(R0)
 4423  016310  012602                          MOV     (SP)+,R2
 4424  016312  012601                          MOV     (SP)+,R1
 4425  016314  012600                          MOV     (SP)+,R0
 4426  016316  000207                          RTS     PC
 4427
 4428
 4429
 4430
 4431
 4432
 4433                                  .SBTTL  ERR2
 4434                                ;ERR2
 4435                                ;THIS ROUTINE GETS THE CYLINDER NUMBERS BETWEEN WHICH (IMPLIED) SEEK
 4436                                ;WAS DONE.  (R4)=0 INDICATES SEEK FROM 'OUTADR' TO 'INADR'
 4437                                ;(R4)=1 INDICATES SEEK TO 'OUTADR'.  ON EXIT $REG0 CONTAINS CYL #
 4438                                ;FROM WHICH SEEK WAS INITIATED, $REG1 CONTAINS CYL # TO WHICH SEEK WAS DONE
 4439  016320  013737  001260  001162  ERR2:   MOV     INADR,$REG0     ;GET CYL ADRES
 4440  016326  004737  016434                  JSR     PC,GCYL         ;GO GET CYL# FROM IT
 4441  016332  013737  001162  001164          MOV     $REG0,$REG1     ;SAVE
 4442  016340  013737  001262  001162          MOV     OUTADR,$REG0    ;GET CYL ADRES
 4443  016346  004737  016434                  JSR     PC,GCYL         ;GO GET CYL # FROM IT
 4444  016352  005704                          TST     R4              ;GOING WHICH WAY?
 4445  016354  001407                          BEQ     1$              ;'OUTADR' TO 'INADR', BRANCH
 4446  016356  013746  001162                  MOV     $REG0,-(SP)     ;EXCHANG CYL" TO GET
 4447  016362  013737  001164  001162          MOV     $REG1,$REG0     ;CORRECT 'TO' & 'FROM' CYLS
 4448  016370  012637  001164                  MOV     (SP)+,$REG1
 4449  016374  000207                  1$:     RTS     PC              ;RETURN
 4450
 4451                                  .SBTTL  ERR1
 4452                                ;ERR1
 4453                                ;THIS SUBROUTINE FINDS OUT THE CYLINDER NOS. BETWEEN WHICH THE SEEK
 4454                                ;IS DONE.THE CYLINDER # WHERE THE  HEADS WHERE PRIOR TO MOVING, IS
 4455                                ;DEPOSITED IN $REG0. THE CYLINDER # WHERE THE HEADS SHOULD BE AFTER
 4456                                ;MOVEMENT, IS DEPOSITED IN $REG1.  R4 INDICATES WHICH DIRECTION THE
 4457                                ;HEADS WERE MOVING, IN OR OUT. R5 CONTAINS THE
 4458                                ;DISK ADDRESS (IN OR OUT AS THE CASE MAY BE).
 4459
 4460  016376  010537  001162          ERR1:   MOV     R5,$REG0
 4461  016402  004737  016434                  JSR     PC,GCYL ;GO GET CYL #
 4462  016406  005704                          TST     R4              ;WAS GOING IN OR OUT
 4463  016410  001006                          BNE     1$              ;OUT
 4464  016412  013737  001162  001164          MOV     $REG0,$REG1
 4465  016420  005037  001162                  CLR     $REG0
```

```
 4466  016424  000207                          RTS     PC
 4467  016426  005037  001164          1$:     CLR     $REG1
 4468  016432  000207                          RTS     PC
 4469
 4470
 4471                                          .SBTTL  GCYL
 4472                                  ;GCYL
 4473                                  ;THIS ROUTINE EXTRACTS THE CYLINDER NO. FROM THE DISK ADDRESS
 4474                                  ;CONTAINED IN '$REGO', AND THEN STORES IT BACK IN '$REGO'
 4475  016434  010046                  GCYL:   MOV     R0,-(SP)        ;PUSH R0 ONTO STACK
 4476  016436  013700  001162                  MOV     $REGO,R0
 4477  016442  042700  160037                  BIC     #160037,R0      ;MASK OUT DRV # BITS &
 4478                                                                  ;SUR, SEC BITS IF PRESENT
 4479  016446  006200                          ASR     R0
 4480  016450  006200                          ASR     R0              ;SHIFT CYL BITS RIGHT
 4481  016452  006200                          ASR     R0              ;BY 5
 4482  016454  006200                          ASR     R0
 4483  016456  006200                          ASR     R0
 4484  016460  010037  001162                  MOV     R0,$REGO        ;STORE CYL # IN $REGO
 4485  016464  012600                          MOV     (SP)+,R0        ;POP R0 FROM STACK
 4486  016466  000207                          RTS     PC              ;EXIT
 4487
 4488
 4489                                          .SBTTL  DRV.RESET - DRIVE RESET ROUTINE
 4490                                          .SBTTL  RESDON - WAIT FOR DRIVE RESET TO BE DONE
 4491                                  ;DR.RST
 4492                                  ;THIS ROUTINE DOES A DRIVE RESET ON THE DRIVE WHOOSE ADDRESS IS IN
 4493                                  ;RKDA. MULTIPLE RETURN ADDRESSES FOR THIS ROUTINE ARE PROVIDED.
 4494                                  ;IF THERE IS NO ERROR (R/W/S RDY SETS WITHIN CERTAIN TIME)  THEN
 4495                                  ;A NORMAL EXIT IS MADE. IF R/W/S RDY DOES NOT SET ERROR IS REPORTED.
 4496
 4497
 4498  016470  005037  001174  DR.RST: CLR     $REG5           ;INITIALIZE THE COUNT
 4499  016474  013777  001230  162762          MOV     DRIVAD,@RKDA
 4500  016502  012777  000015  162746          MOV     #15,@RKCS       ;DRIVE RESET, GO
 4501  016510  104421                          CON.RDY
 4502  016512  000402                          BR      RES.DO+4
 4503  016514  005037  001174  RES.DO: CLR     $REG5
 4504  016520  032777  000100  162724  1$:     BIT     #100,@RKDS              ;DID R/W/S RDY SET?
 4505  016526  001024                          BNE     2$
 4506  016530  012746  177770                  MOV     #-10,-(SP)      ;PUSH COUNT ON SP
 4507  016534  005216                          INC     (SP)            ;COUNT IT DOW
 4508  016536  001376                          BNE     .-2
 4509  016540  005726                          TST     (SP)+           ;POP UP SP
 4510  016542  005237  001174                  INC     $REG5                   ;IF NOT WAIT
 4511  016546  001364                          BNE     1$                      ;WAITED LONG?
 4512  016550  032777  020000  162362          BIT     #SW13,@SWR
 4513  016556  001010                          BNE     2$
 4514  016560  104420                          TYPMSG
 4515  016562  002027                          MSG12
 4516  016564  104420  001733                  TYPMSG  ,MSG7
 4517  016570  011646                          MOV     (SP),-(SP)
 4518  016572  162716  000002                  SUB     #2,(SP)
 4519  016576  104402                          TYPOC
 4520  016600  000002                  2$:     RTI
 4521
```

```
4522
4523
4524                                            .SBTTL  CON.RESET - CONTROL RESET ROUTINE
4525                                            .SBTTL  CON.RDY - WAIT FOR CONTROL READY
4526                                    ;CON.RESET
4527                                    ;CON.RDY
4528                                    ;THIS ROUTINE IS CALLED BY USING 'CNT.RESET' WHICH IS ACTUALL'
4529                                    ;'TRAP' INSTRUCTION WITH THE LOWER BYTE ENCODED TO PROVIDE
4530                                    ;AN INDEX TO THE CONTROL-RESET ROUTINE BELOW.
4531                                    ;THE ROUTINE ISSUES A CONTROL RESET AND WAITS FOR
4532                                    ;THE 'CNTRL RDY' FLAG TO SET.  WHEN THE FLAG SETS
4533                                    ;AN EXIT IS MADE OUT OF THE ROUTINE.  IF 'CNTRL-RDY'
4534                                    ;DOES NOT SET WITHIN A CERTAIN TIME AN ERROR MESSAGE
4535                                    ;       CNT RDY DIDN'T SET
4536                                    ;       PC=XXXXXX RKCS=XXXXXX
4537                                    ;IS GIVEN.
4538                                    ;THIS ROUTINE IS CALLED THROUGH THE 'TRAP' INSTRUCTION
4539                                    ;USING THE LOWER BYTE AS AN INDEX TO THIS ROUTINE.
4540                                    ;THE TRAP DECODER LOCATED AT '$TRAP'.
4541
4542
4543
4544                                    ;CN.RDY
4545                                    ;THE CN.RDY ROUTINE IS CALLED BY USING CNT.RDY WHICH IS A TRAP
4546                                    ;INSTRUCTION WITH ITS LOWER BYTE ENCODED.
4547                                    ;THIS ROUTINE WAITS FOR THE CONTROL READY BIT TO SET AND WHEN IT
4548                                    ;SETS EXITS OUT. IF WITHIN A CERTAIN TIME CNTRL RDY DOES
4549                                    ;NOT SET AN ERROR IS REPORTED. WAITING TIME IS 883 MS FOR 11/20
4550                                    ;175 MS FOR 11/45 WITH BIPOLAR MEMORY.
4551  016602  012777  000001  162646  CN.RST: MOV     #1,@RKCS        ;ISSUE A CONTROL RESET
4552  016610  012737  177500  001170          MOV     #-300,$REG3     ;SET UP COUNT
4553  016616  000402                          BR      CN.RDY+4                ;SKIP OVER CN.RDY
4554  016620  005037  001170          CN.RDY: CLR     $REG3
4555  016624  105777  162626          1$:     TSTB    @RKCS           ;DID CNTRL-RDY SET?
4556  016630  100431                          BMI     2$              ;YES, EXIT
4557  016632  005237  001170                  INC     $REG3           ;WAITED LONG?
4558  016636  001372                          BNE     1$              ;IF NOT, GO BAK & WAIT
4559  016640  104420                          TYPMSG
4560  016642  001742                          MSG10
4561  016644  104401  016652                  TYPE    ,65$            ;;TYPE ASCIZ STRING
4562  016650  000403                          BR      64$             ;;GET OVER THE ASCIZ
4563                                  ;;65$:  .ASCIZ  <15><12>/PC=/
4564  016660                          64$:
4565  016660  011646                          MOV     (SP),-(SP)
4566  016662  162716  000002                  SUB     #2,(SP)
4567  016666  104402                          TYPOC                   ;GO TYPE PC IN THE MAIN PROGRAM.
4568                                                                  ; WHERE ERROR OCCURRED
4569  016670  104401  016676                  TYPE    ,67$            ;;TYPE ASCIZ STRING
4570  016674  000404                          BR      66$             ;;GET OVER THE ASCIZ
4571                                  ;;67$:  .ASCIZ  / RKCS=/
4572  016706                          66$:
4573  016706  017746  162544                  MOV     @RKCS,-(SP)     ;GET RKCS
4574  016712  104402                          TYPOC                   ;GO TYPE IT
4575
4576  016714  000002                  2$:     RTI                     ;RETURN FROM THIS
4577                                                                  ;ROUTINE TO THE MAIN
```

```
                                                                ;PROGRAM

                                        .SBTTL  TST.RWS - WAIT FOR R/W/S RDY
                                ;TST.RWS
                                ;THIS ROUTINE WAITS FOR THE R/W/S READY TO  ET AND RETURNS
                                ;TO THE MAIN PROGRAM WHEN IT SETS.  IF IT DOES NOT SET
                                ;WITHIN A CERTAIN TIME  AN ERROR IS REPORTED.
                                ;WAITING TIME APPROX. 1040 MS FOR 11/20. 208 MS FOR 11/45

4588  016716  005037  001264          TSTRWS: CLR     TIMER
4589  016722  032777  000100  162522  1$:     BIT     #100,@RKDS
4590  016730  001017                          BNE     2$
4591  016732  005237  001264                  INC     TIMER
4592  016736  001371                          BNE     1$
4593  016740  032777  020000  162172          BIT     #BIT13,@SWR
4594  016746  001010                          BNE     2$
4595  016750  104420  002027                  TYPMSG  ,MSG12
4596  016754  104420  001733                  TYPMSG  ,MSG7
4597  016760  011646                          MOV     (SP),-(SP)
4598  016762  162716  000002                  SUB     #2,(SP)
4599  016766  104402                          TYPOC
4600  016770  000002                  2$:     RTI
4601
4602                                          .SBTTL  TEST ABORT ROUTINE
4603                                  ;ABRT
4604
4605  016772  104401  001616          ABRT:   TYPE    ,MSG3
4606  016776  113746  001102                  MOVB    $TSTNM,-(SP)
4607  017002  104402                          TYPOC
4608  017004  000207                          RTS     PC
4609
4610
4611                                  ;COMMON SUBROUTINES & HANDLERS
4612
4613                                  .SBTTL  SCOPE HANDLER ROUTINE
4614
4615                                  ;;*****************************************************************
4616                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
4617                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
4618                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
4619                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
4620                                  ;*SW14=1         LOOP ON TEST
4621                                  ;*SW09=1         LOOP ON ERROR
4622                                  ;*CALL
4623                                  ;*      SCOPE           ;;SCOPE=IOT
4624
4625  017006                          $SCOPE:
4626  017006  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
4627  017010  032777  000400  162122          BIT     #SWB,@SWR               ;WAS SWB USED TO SELECT
4628  017016  001053                          BNE     $OVER           ;A TEST? IF YES, SKIP OVER
4629                                                                  ;THE REST, U ARE LOOPING ON
4630  017020  032777  040000  162112  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
4631  017026  001047                          BNE     $OVER           ;;YES IF SW14=1
4632                                  ;*****START OF CODE FOR THE XOR TESTER*****
4633  017030  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
```

```
4634
4635  017032  013746  000004                MOV     @#ERRVEC,-(SP)  ;;THIS INSTRUCTION TO A "NOP" (NOP=240
4636  017036  012737  017056  000004        MOV     #5$,@#ERRVEC    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
4637  017044  005737  177060                TST     @#177060        ;;SET FOR TIMEOUT
4638  017050  012637  000004                MOV     (SP)+,@#ERRVEC  ;;TIME OUT ON XOR?
4639  017054  000421                        BR      $SVLAD          ;;RESTORE THE ERROR VECTOR
4640  017056  022626                5$:     CMP     (SP)+,(SP)+     ;;GO TO THE NEXT TEST
4641  017060  012637  000004                MOV     (SP)+,@#ERRVEC  ;;CLEAR THE STACK AFTER A TIME OUT
4642  017064  000407                        BR      7$              ;;RESTORE THE ERROR VECTOR
4643  017066                        6$:;#####END OF CODE FOR THE XOR TESTER#####  ;;LOOP ON THE PRESENT TEST
4644  017066  105737  001103        2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
4645  017072  001412                        BEQ     $SVLAD          ;;BR IF NO
4646  017074  032777  001000  162036        BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
4647  017102  001404                        BEQ     4$              ;;BR IF NO
4648  017104  013737  001110  001106  7$:   MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
4649  017112  000415                        BR      $OVER
4650  017114  105037  001103        4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
4651  017120  105237  001102        $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
4652  017124  011637  001106                MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
4653  017130  011637  001110                MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
4654  017134  005037  001204                CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
4655  017140  112737  000001  001115        MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
4656  017146  013777  001102  161766  $OVER: MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
4657  017154  013716  001106                MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
4658  017160  000002                        RTI                     ;;FIXES PS
4659
4660                                ;;*************************************************************
4661
4662
4663                                .SBTTL  ERROR HANDLER ROUTINE
4664
4665                                ;*SW15=1        HALT ON ERROR
4666                                ;*SW13=1        INHIBIT ERROR TYPEOUTS
4667                                ;*SW10=1        BELL ON ERROR
4668                                ;*SW09=1        LOOP ON ERROR
4669                                ;*SW12=1        CYCLE ON ERROR TO PREVIOS 'SCOPE' STATEMENT
4670                                ;*GO TO ERRTYP ON ERROR
4671                                ;*NOT FROM SYSMAC
4672
4673  017162  104407                $ERROR: CKSWR                           ;CHECK FOR SOFTWARE SWITCH REGISTER REQUEST
4674  017164  105237  001103        7$:     INCB    $ERFLG          ;SET THE ERROR FLAG
4675  017170  001775                        BEQ     7$              ;DON'T LET THE FLAG GO TO ZERO
4676  017172  013777  001102  161742        MOV     $TSTNM,@DISPLAY
4677  017200  032777  002000  161732        BIT     #SW10,@SWR
4678  017206  001402                        BEQ     1$
4679  017210  104401  001206                TYPE    ,$BELL
4680  017214  005237  001112        1$:     INC     $ERTTL
4681  017220  011637  001116                MOV     (SP),$ERRPC
4682
4683  017224  032777  000004  161706        BIT     #SW2,@SWR       ;DROP THE DRIVE?
4684  017232  001404                        BEQ     5$              ;SW NOT SET, SKIP
4685  017234  023727  001112  000006        CMP     $ERTTL,#6       ;MORE THAN 6 ERRORS ON THIS DRIVE?
4686  017242  101044                        BHI     6$              ;YES, DROP THE DRIVE
4687
4688  017244  162737  000002  001116  5$:   SUB     #2,$ERRPC
4689  017252  117737  161640  001114        MOVB    @$ERRPC,$ITEMB
```

```
4690  017260  032777  020000  161652          BIT     #SW13,@SWR
4691  017266  001004                          BNE     2$
4692  017270  004737  017446                  JSR     PC,@#ERRTYP
4693  017274  104401  001213                  TYPE    ,$CRLF
4694  017300  023737  000042  000046  2$:     CMP     @#42,@#46               ;ARE WE IN ACT11 AUTO MODE? ;
4695  017306  001403                          BEQ     .+10                    ;YES, HALT ON ERROR
4696  017310  005777  161624                  TST     @SWR                    ;SWR15 (HALT ON ERROR) SET?
4697  017314  100002                          BPL     3$                      ;BRANCH IF NOT
4698  017316  000000                          HALT                            ;HALT ON ERROR
4699  017320  104407                          CKSWR                           ;CHECK FOR SOFTWARE SWITCH REGISTER REQUEST
4700  017322  032777  010000  161610  3$:     BIT     #SW12,@SWR
4701  017330  001402                          BEQ     .+6
4702  017332  013716  001106                  MOV     $LPADR,(SP)
4703  017336  032777  001000  161574          BIT     #SW09,@SWR
4704  017344  001402                          BEQ     4$
4705  017346  013716  001110                  MOV     $LPERR,(SP)
4706  017352  000002                  4$:     RTI
4707
4708  017354  013746  001226          6$:     MOV     DRVPTR,-(SP)    ;GET POINTER TO DRIVE #
4709  017360  162716  000002                  SUB     #2,(SP)
4710  017364  042736  000377                  BIC     #377,@(SP)+     ;CLEAR THE DRIVE PRESENT FLAG
4711  017370  104401  002064                  TYPE    ,MSG14
4712  017374  013746  001230                  MOV     DRIVAD,-(SP)
4713  017400  000241                          CLC                     ;GET THE DRIVE #
4714  017402  006116                          ROL     (SP)
4715  017404  006116                          ROL     (SP)
4716  017406  006116                          ROL     (SP)
4717  017410  006116                          ROL     (SP)
4718  017412  104402                          TYPOC                   ;TYPE IT OUT
4719  017414  104401  017422                  TYPE    ,65$            ;;TYPE ASCIZ STRING
4720  017420  000405                          BR      64$             ;;GET OVER THE ASCIZ
4721                                  ;;65$:  .ASCIZ  / DROPPED/
4722  017434                          64$:
4723  017434  105337  001224                  DECB    DRIVS           ;DECRMNT # OF DRIVS PRESENT
4724  017440  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
4725  017442  000137  015232                  JMP     @TEOP           ;EXIT
4726
4727  017446                          ERRTYP:
4728  017446  104401  001213                  TYPE    ,$CRLF          ;"CARRIAGE RETURN" & LINE FEED"
4729  017452  010046                          MOV     R0,-(SP)        ;SAVE R0
4730  017454  005000                          CLR     R0              ;PICKUP THE ITEM INDEX
4731  017456  153700  001114                  BISB    @#$ITEMB,R0
4732  017462  001011                          BNE     1$              ;IF ITEM NUMBER IS ZERO, JUST
4733
4734                                                                  ;TYPE THE PC OF THE ERROR
4735  017464  013746  001116                  MOV     $ERRPC,-(SP)    ;SAVE $ERRPC FOR TYPEOUT
4736                                                                  ;ERROR ADDRESS
4737  017470  104402                          TYPOC                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
4738  017472  104401                          TYPE
4739  017474  001733                          MSG7
4740  017476  013746  001116                  MOV     $ERRPC,-(SP)
4741  017502  104402                          TYPOC
4742  017504  000440                          BR      6$              ;GET OUT
4743  017506  005300                  1$:     DEC     R0              ;ADJUST THE INDEX SO THAT IT WILL
4744  017510  006300                          ASL     R0              ;       WORK FOR THE ERROR TABLE
4745  017512  006300                          ASL     R0
```

```
4746  017514  006300                        ASL     R0
4747  017516  062700  002122                ADD     #$ERRTB,R0      ;FORM TABLE POINTER
4748  017522  012037  017532                MOV     (R0)+,2$        ;PICKUP "ERROR MESSAGE" POINTER
4749  017526  001404                        BEQ     3$              ;SKIP TYPEOUT IF NOT POINTER
4750  017530  104401                        TYPE                    ;TYPE THE "ERROR MESSAGE"
4751  017532  000000            2$:         .WORD   0               ;"CARRIAGE RETURN" & LINE FEED"
4752  017534  104401  001213                TYPE    ,$CRLF          ;PICKUP "DATA HEADER" POINTER
4753  017540  032777  004000  161372  3$:   BIT     #SW11,@SWR      ;DUMP OUT ALL RK REGISTERS
4754  017546  001042                        BNE     10$             ;YES, BRANCH
4755  017550  012037  017560                MOV     (R0)+,4$        ;PICKUP "DATA HEADER" POINTER
4756  017554  001412                        BEQ     5$              ;SKIP TYPEOUT IF 0
4757  017556  104401                        TYPE                    ;TYPE THE "DATA HEADER"
4758  017560  000000            4$:         .WORD   0               ;"DATA HEADER" POINTER GOES HERE
4759  017562  104401  001213                TYPE    ,$CRLF          ;"CARRIAGE RETURN" & LINE FEED"
4760  017566  062700  000002                ADD     #2,R0           ;FORM POINTER TO TERMINATOR
4761  017572  005710                        TST     (R0)            ;IS THE TERMINATOR 0?
4762  017574  001017                        BNE     9$              ;IF NOT, BRANCH
4763  017576  162700  000002                SUB     #2,R0           ;YES, IT IS 0.  REPOINT TO "DATA"
4764                                                                ;GO TYPE OUT DATA AS USUAL
4765  017602  011000            5$:         MOV     (R0),R0         ;PICKUP "DATA TABLE" POINTER
4766  017604  001004                        BNE     7$              ;GO TYPE THE DATA
4767  017606  012600            6$:         MOV     (SP)+,R0        ;RESTORE R0
4768  017610  104401  001213                TYPE    ,$CRLF          ;"CARRIAGE RETURN" & LINE FEED"
4769  017614  000207                        RTS     PC              ;RETURN
4770  017616                    7$:
4771  017616  013046                        MOV     @(R0)+,-(SP)    ;SAVE @(R0)+ FOR TYPEOUT
4772  017620  104402                        TYPOC                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
4773  017622  005710                        TST     (R0)            ;IS THERE ANOTHER NUMBER?
4774  017624  001770                        BEQ     6$              ;BR IF NO
4775  017626  104401  002110                TYPE    ,BLNKS2
4776  017632  000771                        BR      7$
4777  017634  004770  000000            9$:         JSR     PC,@(R0)        ;GO TO THE SPECIAL ERROR
4778                                                                ;DATA HANDLING SUBROUTINE
4779                                                                ;NOTE THAT THIS ROUTINE IS
4780                                                                ;THE ONE INDICATED IN THE
4781                                                                ;LAST WORD OF AN ERROR
4782                                                                ;ITEM IN THE ERROR TABLE
4783                                                                ;(STARTING AT $ERRTB)
4784  017640  104401                        TYPE
4785  017642  001733                        MSG7
4786  017644  013746  001116                MOV     $ERRPC,-(SP)
4787  017650  104402                        TYPOC
4788  017652  000755                        BR      6$              ;GO BACK, TO THE EXIT POINT
4789                                                                ;FOR 'ERRTYP'
4790
4791  017654  004737  017662            10$:        JSR     PC,DMPREG
4792  017660  000752                        BR      6$
4793
4794
4795
4796                                    ;DMPREG
4797                                    ;DUMPS OUT ALL RK REGISTERS WHEN SW 11 IS SET
4798
4799  017662                            DMPREG:
4800  017662  104401  017670                    TYPE    ,65$            ;;TYPE ASCIZ STRING
4801  017666  000441                            BR      64$             ;;GET OVER THE ASCIZ
```

```
                                        ;;65$:  .ASCIZ  <15><12>/ PC     RKDS    RKER    RKCS    RKWC    RKBA    RKDA    RKDB//
                                        64$:
017772                                          MOV     $ERRPC,-(SP)
017772  013746  001116                          TYPOC
017776  104402                                  TYPE    ,BLNKS2
020000  104401  002110                          MOV     R0,-(SP)
020004  010046                                  MOV     #RKDS,R0
020006  012700  001452                  1$:     MOV     @(R0)+,-(SP)
020012  013046                                  TYPOC
020014  104402                                  TYPE    ,BLNKS2
020016  104401  002110                          CMP     R0,#RKDB
020022  020027  001466                          BLE     1$
020026  003771                                  MOV     (SP)+,R0
020030  012600                                  RTS     PC
020032  000207

                                        .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                        ;;**************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                        ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                        ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                        ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                        ;*REPLACED WITH SPACES.
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                                        ;*      TYPDS                   ;;GO TO THE ROUTINE

020034                                  $TYPDS:
020034  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
020036  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
020040  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
020042  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
020044  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
020046  012746  020200                          MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
020052  016605  000020                          MOV     20(SP),R5       ;;GET THE INPUT NUMBER
020056  100004                                  BPL     1$              ;;BR IF INPUT IS POS.
020060  005405                                  NEG     R5              ;;MAKE THE BINARY NUMBER POS.
020062  112766  000055  000001                  MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
020070  005000                          1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
020072  012703  020250                          MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
020076  112723  000040                          MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
020102  005002                          2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
020104  016001  020240                          MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
020110  160105                          3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
020112  002402                                  BLT     4$              ;;BR IF DONE
020114  005202                                  INC     R2              ;;INCREASE THE BCD DIGIT BY 1
020116  000774                                  BR      3$
020120  060105                          4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
020122  005702                                  TST     R2              ;;CHECK IF BCD DIGIT=0
020124  001002                                  BNE     5$              ;;FALL THROUGH IF 0
020126  105716                                  TSTB    (SP)            ;;STILL DOING LEADING 0'S?
020130  100407                                  BMI     7$              ;;BR IF YES
```

```
 4858  020132  106316                  5$:     ASLB    (SP)            ;;MSD?
 4859  020134  103003                          BCC     6$              ;;BR IF NO
 4860  020136  116663  000000'  177777         MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 4861  020144  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 4862  020150  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 4863  020154  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 4864  020156  005720                          TST     (R0)+           ;;JUST INCREMENTING
 4865  020160  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
 4866  020164  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
 4867  020166  003002                          BGT     8$              ;;GO TO EXIT
 4868  020170  010502                          MOV     R5,R2           ;;GET THE LSD
 4869  020172  000764                          BR      6$              ;;GO CHANGE TO ASCII
 4870  020174  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 4871  020176  100003                          BPL     9$              ;;BR IF NO
 4872  020200  116663  177777  177776         MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 4873  020206  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 4874  020210  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
 4875  020212  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
 4876  020214  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 4877  020216  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 4878  020220  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 4879  020222  104401  020250                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 4880  020226  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 4881  020234  012616                          MOV     (SP)+,(SP)
 4882  020236  000002                          RTI                     ;;RETURN TO USER
 4883  020240  023420                  $DTBL:  10000.
 4884  020242  001750                          1000.
 4885  020244  000144                          100.
 4886  020246  000012                          10.
 4887  020250  000004                  $DBLK:  .BLKW   4
 4888
 4889                                  .SBTTL  TYPE ROUTINE
 4890
 4891                                  ;;*****************************************************************
 4892                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 4893                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 4894                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 4895                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 4896                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 4897                                  ;*
 4898                                  ;*CALL:
 4899                                  ;*1) USING A TRAP INSTRUCTION
 4900                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 4901                                  ;*OR
 4902                                  ;*      TYPE
 4903                                  ;*      MESADR
 4904                                  ;*
 4905
 4906  020260  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 4907  020264  100002                          BPL     1$              ;;BR IF YES
 4908  020266  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
 4909  020270  000407                          BR      3$              ;;LEAVE
 4910  020272  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
 4911  020274  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 4912  020300  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 4913  020302  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
```

```
 4914  020304  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 4915  020306  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
 4916  020310  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 4917  020314  000002                          RTI                     ;;RETURN
 4918  020316  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 4919  020322  001430                          BEQ     8$
 4920  020324  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 4921  020330  001006                          BNE     5$
 4922  020332  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
 4923  020334  104401                          TYPE                    ;;TYPE A CR AND LF
 4924  020336  001213                          $CRLF
 4925  020340  105037  020474                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 4926  020344  000755                          BR      2$              ;;GET NEXT CHARACTER
 4927  020346  004737  020430          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 4928  020352  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 4929  020356  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
 4930  020360  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
 4931                                                                  ;;AND THE NULL CHAR.
 4932  020364  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 4933  020370  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 4934  020372  004737  020430                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
 4935  020376  105337  020474                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 4936  020402  000770                          BR      7$              ;;LOOP
 4937
 4938                                  ;HORIZONTAL TAB PROCESSOR
 4939
 4940  020404  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 4941  020410  004737  020430          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 4942  020414  132737  000007  020474          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 4943  020422  001372                          BNE     9$              ;;TAB STOP
 4944  020424  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
 4945  020426  000724                          BR      2$              ;;GET NEXT CHARACTER
 4946  020430  105777  160514          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 4947  020434  100375                          BPL     $TYPEC
 4948  020436  116677  000002  160506          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 4949  020444  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 4950  020452  001003                          BNE     1$              ;;BRANCH IF NO
 4951  020454  105037  020474                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 4952  020460  000406                          BR      $TYPEX          ;;EXIT
 4953  020462  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 4954  020470  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 4955  020472  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 4956  020474  000000          $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
 4957  020476  000207          $TYPEX: RTS     PC
 4958
 4959
 4960
 4961                                  .SBTTL  INTEGER MULTIPLY ROUTINE
 4962
 4963                                  ;;*****************************************************************
 4964                                  ;*CALL
 4965                                  ;*      MOV     MULTIPLER,-(SP)
 4966                                  ;*      MOV     MULTIPLICAND,-(SP)
 4967                                  ;*      JSR     PC,@#$MULT
 4968                                  ;*      RETURN  ;;PRODUCT IS ON THE STACK
 4969                                  ;*
```

```
                                ;*      STACK   PRODUCT
                                ;*      -----   -------
                                ;*      TOP     LSB'S
                                ;*      +2      MSB'S

020500                          $MULT:
020500  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
020502  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
020504  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
020506  005046                          CLR     -(SP)           ;;CLEAR THE SIGN KEY
020510  016601  000012                  MOV     12(SP),R1       ;;GET THE MULTIPLICAND
020514  100002                          BPL     1$              ;;BR IF PLUS
020516  005216                          INC     (SP)            ;;SET THE SIGN KEY
020520  005401                          NEG     R1              ;;MAKE THE MULTIPLICAND POSTIVE
020522  016602  000014          1$:     MOV     14(SP),R2       ;;GET THE MULTIPLIER
020526  100002                          BPL     2$              ;;BR IF PLUS
020530  005316                          DEC     (SP)            ;;UPDATE THE SIGN KEY
020532  005402                          NEG     R2              ;;MAKE THE MULTIPLIER POSTIVE
020534  012746  000021          2$:     MOV     #17.,-(SP)      ;;SET THE LOOP COUNT
020540  005000                          CLR     R0              ;;SETUP FOR THE MULTIPLY LOOP
020542  103001                  3$:     BCC     4$              ;;DON'T ADD IF MULTIPLICAND = 0
020544  060200                          ADD     R2,R0
020546  006000                  4$:     ROR     R0              ;;POSITION THE PARITIAL PRODUCT AND
020550  006001                          ROR     R1              ;;THE MULTIPLICAND
020552  005316                          DEC     (SP)            ;;HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
020554  001372                          BNE     3$              ;;BR IF NO
020556  022616                          CMP     (SP)+,(SP)      ;;SHOULD PRODUCT BE NEGATIVE?
020560  001403                          BEQ     5$              ;;GO TO EXIT IF NO
020562  005400                          NEG     R0              ;;YES--SO MAKE IT SO
020564  005401                          NEG     R1
020566  005600                          SBC     R0
020570  005726                  5$:     TST     (SP)+           ;;CLEAR SIGN INFO. OFF OF STACK
020572  010066  000012                  MOV     R0,12(SP)       ;;PUT THE PRODUCT ON THE STACK (MSB'S.
020576  010166  000010                  MOV     R1,10(SP)       ;;LSB'S
020602  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
020604  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
020606  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
020610  000207                          RTS     PC

                                .SBTTL  TTY INPUT ROUTINE

                                ;;**************************************************************
                                .ENABL  LSB
020612  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
020614  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
020616  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
020620  000001                  $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
        020621                  $TKQEND=.
        020622                  .EVEN

                                ;*TK INITIALIZE ROUTINE
                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                ;
                                ;*CALL:
                                ;*      JSR     PC,$TKINT
```

```
5026                                    ;*      RETURN
5027                                    ;
5028  020622  005037  020612            $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
5029  020626  012737  020620  020614            MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
5030  020634  013737  020614  020616            MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
5031  020642  012737  020672  000060            MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
5032  020650  012737  000200  000062            MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
5033  020656  005777  160264                    TST     @$TKB           ;;CLEAR DONE FLAG
5034  020662  012777  000100  160254            MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
5035  020670  000207                            RTS     PC              ;;RETURN TO CALLER
5036
5037                                    ;*TK SERVICE ROUTINE
5038                                    ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
5039                                    ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
5040                                    ;*IT IN THE QUEUE.
5041                                    ;
5042  020672  117746  160250            $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
5043  020676  042716  177600                    BIC     #^C177,(SP)     ;;STRIP THE JUNK
5044  020702  021627  000007            1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
5045  020706  001004                            BNE     2$              ;;BRANCH IF NO
5046  020710  022737  000176  001140            CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
5047  020716  001500                            BEQ     6$              ;;GO TO SWR CHANGE
5048
5049  020720                            2$:
5050  020720  022737  000001  020612            CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
5051  020726  001004                            BNE     3$              ;;BRANCH IF NO
5052  020730  104401  001206                    TYPE    ,$BELL          ;;RING THE TTY BELL
5053  020734  005726                            TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
5054  020736  000451                            BR      5$              ;;EXIT
5055  020740  021627  000023            3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
5056  020744  001021                            BNE     32$             ;;BRANCH IF NO
5057  020746  005077  160172                    CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
5058  020752  005726                            TST     (SP)+           ;;CLEAN CHAR OFF STACK
5059  020754  105777  160164            31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
5060  020760  100375                            BPL     31$             ;;LOOP UNTIL ITS THERE
5061  020762  117746  160160                    MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
5062  020766  042716  177600                    BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
5063  020772  022627  000021                    CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
5064  020776  001366                            BNE     31$             ;;BRANCH IF NO
5065  021000  012777  000100  160136            MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
5066  021006  000002                            RTI                     ;;RETURN
5067  021010  005237  020612            32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
5068  021014  021627  000140                    CMP     (SP),#140       ;;IS IT UPPER CASE?
5069  021020  002405                            BLT     4$              ;;BRANCH IF YES
5070  021022  021627  000175                    CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
5071  021026  003002                            BGT     4$              ;;BRANCH IF YES
5072  021030  042716  000040                    BIC     #40,(SP)        ;;MAKE IT UPPER CASE
5073  021034  112677  177554            4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
5074  021040  005237  020614                    INC     $TKQIN          ;;UPDATE THE POINTER
5075  021044  023727  020614  020621            CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
5076  021052  001003                            BNE     5$              ;;BRANCH IF NO
5077  021054  012737  020620  020614            MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
5078  021062  000002                    5$:     RTI                     ;;RETURN
5079
5080                                    ;;*****************************************************************
5081                                    ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
```

```
5082                                    ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
5083                                    ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
5084                                    ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
5085  021064  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
5086  021072  001104                          BNE     15$             ;;EXIT IF NOT
5087  021074  105777  160044                  TSTB    @#TKS           ;;IS A CHAR WAITING?
5088  021100  100101                          BPL     15$             ;;IF NOT, EXIT
5089  021102  117746  160040                  MOVB    @#TKB,-(SP)     ;;YES
5090  021106  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
5091  021112  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
5092  021116  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
5093                                                                  ;;AND EXIT
5094
5095                                    ;;*****************************************************************
5096                                    ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
5097                                    ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
5098                                    ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
5099  021120  123727  001134  000001  6$:      CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
5100  021126  001674                          BEQ     2$              ;;BRANCH IF YES
5101  021130  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
5102  021132  004737  020622                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
5103  021136  005077  160002                  CLR     @#TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
5104  021142  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
5105
5106  021150  104401  021727                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
5107  021154  104401  021734  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
5108  021160  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
5109  021164  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS.
5110  021166  104401  021745                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
5111  021172  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
5112  021174  005046                          CLR     -(SP)           ;;THE NEW SWR
5113  021176  105777  157742          7$:     TSTB    @#TKS           ;;CHAR THERE?
5114  021202  100375                          BPL     7$              ;;IF NOT TRY AGAIN
5115
5116  021204  117746  157736                  MOVB    @#TKB,-(SP)     ;;PICK UP CHAR
5117  021210  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
5118
5119
5120
5121  021214  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
5122  021220  001005                          BNE     10$             ;;BRANCH IF NOT
5123  021222  104401  021722                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
5124  021226  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
5125  021232  000757                          BR      19$             ;;LET'S TRY IT AGAIN
5126
5127
5128  021234  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
5129  021240  001022                          BNE     16$             ;;BRANCH IF NO
5130  021242  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
5131  021246  001403                          BEQ     11$             ;;BRANCH IF YES
5132  021250  016677  000002  157662          MOV     2(SP),@SWR      ;;SAVE NEW SWR
5133  021256  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
5134  021262  104401  001213          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
5135  021266  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS
5136  021274  001003                          BNE     15$             ;;BRANCH IF NOT
5137  021276  012777  000100  157640          MOV     #100,@#TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
```

```
5138  021304  000002                  15$:    RTI                     ;;RETURN
5139  021306  004737  020430          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
5140  021312  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
5141  021316  002420                  BLT     18$                     ;;BRANCH IF YES
5142  021320  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
5143  021324  003015                  BGT     18$                     ;;BRANCH IF YES
5144  021326  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
5145  021332  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
5146  021336  001403                  BEQ     17$                     ;;BRANCH IF YES
5147  021340  006316                  ASL     (SP)                    ;;NO, SHIFT PRESENT
5148  021342  006316                  ASL     (SP)                    ;;    CHAR OVER TO MAKE
5149  021344  006316                  ASL     (SP)                    ;;    ROOM FOR NEW ONE.
5150  021346  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
5151  021352  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
5152  021356  000707                  BR      7$                      ;;GET THE NEXT ONE
5153  021360  104401  001212          18$:    TYPE    ,$QUES          ;;TYPE ?(CR)(LF)
5154  021364  000720                  BR      20$                     ;;SIMULATE CONTROL-U
5155                                  .DSABL  LSB
5156
5157
5158                                  ;;*****************************************************************
5159                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
5160                                  ;*CALL:
5161                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
5162                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
5163                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
5164                                  ;
5165
5166  021366  011646          $RDCHR: MOV     (SP),-(SP)              ;;PUSH DOWN THE PC AND
5167  021370  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
5168  021376  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
5169  021402  005046                  CLR     -(SP)                   ;;PUT NEW PS ON STACK
5170  021404  012746  021412                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
5171  021410  000002                  RTI                             ;;POP NEW PC AND PS
5172  021412                          64$:
5173  021412  005737  020612          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
5174  021416  001775                  BEQ     1$
5175  021420  005337  020612                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
5176  021424  117766  177166  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
5177  021432  005237  020616                  INC     $TKQOUT         ;;UPDATE THE POINTER
5178  021436  023727  020616  020621          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
5179  021444  001003                  BNE     2$                      ;;BRANCH IF NO
5180  021446  012737  020620  020616          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
5181  021454  000002                  2$:     RTI                     ;;RETURN
5182                                  ;;*****************************************************************
5183                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
5184                                  ;*CALL:
5185                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
5186                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
5187                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
5188
5189  021456  010346          $RDLIN: MOV     R3,-(SP)                ;;SAVE R3
5190  021460  005046                  CLR     -(SP)                   ;;CLEAR THE RUBOUT KEY
5191  021462  012703  021712          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
5192  021466  022703  021722          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
5193  021472  101456                  BLOS    4$                      ;;BR IF YES
```

```
5194  021474  104410                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
5195  021476  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
5196  021500  122713  000177                  10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
5197  021504  001022                          BNE     5$              ;;BR IF NO
5198  021506  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
5199  021510  001007                          BNE     6$              ;;BR IF NO
5200  021512  112737  000134  021710          MOVB    #'\,9$          ;;TYPE A BACK SLASH
5201  021520  104401  021710                  TYPE    ,9$
5202  021524  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
5203  021530  005303                          6$:     DEC     R3              ;;BACKUP BY ONE
5204  021532  020327  021712                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
5205  021536  103434                          BLO     4$              ;;BR IF YES
5206  021540  111337  021710                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
5207  021544  104401  021710                  TYPE    ,9$             ;;GO TYPE
5208  021550  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
5209  021552  005716                          5$:     TST     (SP)            ;;RUBOUT KEY SET?
5210  021554  001406                          BEQ     7$              ;;BR IF NO
5211  021556  112737  000134  021710          MOVB    #'\,9$          ;;TYPE A BACK SLASH
5212  021564  104401  021710                  TYPE    ,9$
5213  021570  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
5214  021572  122713  000025                  7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
5215  021576  001003                          BNE     8$              ;;BR IF NO
5216  021600  104401  021722                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
5217  021604  000726                          BR      1$              ;;GO START OVER
5218  021606  122713  000022                  8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
5219  021612  001011                          BNE     3$              ;;BRANCH IF NO
5220  021614  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
5221  021616  104401  001213                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
5222  021622  104401  021712                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
5223  021626  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
5224  021630  104401  001212                  4$:     TYPE    ,$QUES          ;;TYPE A '?'
5225  021634  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
5226  021636  111337  021710                  3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
5227  021642  104401  021710                  TYPE    ,9$
5228  021646  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
5229  021652  001305                          BNE     2$              ;;LOOP IF NOT RETURN
5230  021654  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
5231  021660  104401  001214                  TYPE    ,$LF            ;;TYPE A LINE FEED
5232  021664  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
5233  021666  012603                          MOV     (SP)+,R3        ;;RESTORE R3
5234  021670  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
5235  021672  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
5236  021700  012766  021712  000004          MOV     #$TTYIN,4(SP)
5237  021706  000002                          RTI                     ;;RETURN
5238  021710     000                          9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
5239  021711     000                          .BYTE   0               ;;TERMINATOR
5240  021712  000010                          $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
5241  021722  052536  005015     000         $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
5242  021727     136  006507  000012          $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
5243  021734  005015  053523  020122          $MSWR:  .ASCIZ  <15><12>/SWR = /
5244  021742  020075     000
5245  021745     040  047040  053505          $MNEW:  .ASCIZ  /  NEW = /
5246  021752  036440  000040
5247
5248                                          .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
5249
```

```
5250                                    ;;*****************************************************************
5251                                    ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
5252                                    ;*CHANGE IT TO BINARY.
5253                                    ;*CALL:
5254                                    ;*      RDOCT                   ;;READ AN OCTAL NUMBER
5255                                    ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
5256                                    ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
5257
5258  021756  011646                    $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
5259  021760  016666  000004  000002            MOV     4(SP),2(SP)     ;;INPUT NUMBER
5260  021766  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
5261  021770  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
5262  021772  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
5263  021774  104411                    1$:     RDLIN                   ;;READ AN ASCIZ LINE
5264  021776  012600                            MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
5265  022000  005001                            CLR     R1              ;;CLEAR DATA WORD
5266  022002  005002                            CLR     R2
5267  022004  112046                    2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
5268  022006  001412                            BEQ     3$              ;;IF ZERO GET OUT
5269  022010  006301                            ASL     R1              ;;*2
5270  022012  006102                            ROL     R2
5271  022014  006301                            ASL     R1              ;;*4
5272  022016  006102                            ROL     R2
5273  022020  006301                            ASL     R1              ;;*8
5274  022022  006102                            ROL     R2
5275  022024  042716  177770                    BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
5276  022030  062601                            ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
5277  022032  000764                            BR      2$              ;;LOOP
5278  022034  005726                    3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
5279  022036  010166  000012                    MOV     R1,12(SP)       ;;SAVE THE RESULT
5280  022042  010237  022056                    MOV     R2,$HIOCT
5281  022046  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
5282  022050  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
5283  022052  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
5284  022054  000002                            RTI                     ;;RETURN
5285  022056  000000                    $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
5286
5287                                    .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
5288
5289                                    ;;*****************************************************************
5290                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
5291                                    ;*OCTAL (ASCII) NUMBER AND TYPE IT.
5292                                    ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
5293                                    ;*CALL:
5294                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
5295                                    ;*      TYPOS                   ;;CALL FOR TYPEOUT
5296                                    ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
5297                                    ;*      .BYTE   M               ;;M=1 OR 0
5298                                    ;*                                      ;;1=TYPE LEADING ZEROS
5299                                    ;*                                      ;;0=SUPPRESS LEADING ZEROS
5300                                    ;*
5301                                    ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
5302                                    ;*$TYPOS OR $TYPOC
5303                                    ;*CALL:
5304                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
5305                                    ;*      TYPON                   ;;CALL FOR TYPEOUT
```

```
5306                                    ;*
5307                                    ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
5308                                    ;*CALL:
5309                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
5310                                    ;*      TYPOC                   ;;CALL FOR TYPEOUT
5311
5312  022060  017646  000000           $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
5313  022064  116637  000001  022303           MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
5314  022072  112637  022305                   MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
5315  022076  062716  000002                   ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
5316  022102  000406                           BR      $TYPON
5317  022104  112737  000001  022303   $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
5318  022112  112737  000006  022305           MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
5319  022120  112737  000005  022302   $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
5320  022126  010346                           MOV     R3,-(SP)        ;;SAVE R3
5321  022130  010446                           MOV     R4,-(SP)        ;;SAVE R4
5322  022132  010546                           MOV     R5,-(SP)        ;;SAVE R5
5323  022134  113704  022305                   MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
5324  022140  005404                           NEG     R4
5325  022142  062704  000006                   ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
5326  022146  110437  022304                   MOVB    R4,$OMODE       ;;SAVE IT FOR USE
5327  022152  113704  022303                   MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
5328  022156  016605  000012                   MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
5329  022162  005003                           CLR     R3              ;;CLEAR THE OUTPUT WORD
5330  022164  006105                   1$:     ROL     R5              ;;ROTATE MSB INTO "C"
5331  022166  000404                           BR      3$              ;;GO DO MSB
5332  022170  006105                   2$:     ROL     R5              ;;FORM THIS DIGIT
5333  022172  006105                           ROL     R5
5334  022174  006105                           ROL     R5
5335  022176  010503                           MOV     R5,R3
5336  022200  006103                   3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
5337  022202  105337  022304                   DECB    $OMODE          ;;TYPE THIS DIGIT?
5338  022206  100016                           BPL     7$              ;;BR IF NO
5339  022210  042703  177770                   BIC     #177770,R3      ;;GET RID OF JUNK
5340  022214  001002                           BNE     4$              ;;TEST FOR 0
5341  022216  005704                           TST     R4              ;;SUPPRESS THIS 0?
5342  022220  001403                           BEQ     5$              ;;BR IF YES
5343  022222  005204                   4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
5344  022224  052703  000060                   BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
5345  022230  052703  000040           5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
5346  022234  110337  022300                   MOVB    R3,8$           ;;SAVE FOR TYPING
5347  022240  104401  022300                   TYPE    ,8$             ;;GO TYPE THIS DIGIT
5348  022244  105337  022302           7$:     DECB    $OCNT           ;;COUNT BY 1
5349  022250  003347                           BGT     2$              ;;BR IF MORE TO DO
5350  022252  002402                           BLT     6$              ;;BR IF DONE
5351  022254  005204                           INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
5352  022256  000744                           BR      2$              ;;GO DO THE LAST DIGIT
5353  022260  012605                   6$:     MOV     (SP)+,R5        ;;RESTORE R5
5354  022262  012604                           MOV     (SP)+,R4        ;;RESTORE R4
5355  022264  012603                           MOV     (SP)+,R3        ;;RESTORE R3
5356  022266  016666  000002  000004           MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
5357  022274  012616                           MOV     (SP)+,(SP)
5358  022276  000002                           RTI                     ;;RETURN
5359  022300     000                   8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
5360  022301     000                           .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
5361  022302     000                   $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
```

```
5362  022303  000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
5363  022304  000000               $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
5364
5365
5366                                       .SBTTL  TYPDSS - TYPE DECIMAL, LEADING ZEROES SUPPRESSED
5367                               ;TYPDSS
5368                               ;ROUTINE FOR TYPING OUT DECIMAL NUMBERS, LEADING 0'S ARE SUPPRESSED
5369                               ;THE NUMBER IS LEFT JUSTIFIED. NOTE THE 16 BIT BINARY NUMBER SHOULD
5370                               ;BE POSITIVE (BIT 15= 0).
5371                               ;CALL:  MOV     NUMBER,-(SP)    ;PUT BINARY NUMBER ON STACK
5372                               ;       TYPDSS                  ;GO TYPE DECIMAL
5373
5374  022306  016637  000004  022346  TYPDES: MOV     4(SP),1$                ;GET THE NUMBER
5375  022314  012746  022346                MOV     #1$,-(SP)               ;PUT PTR ON THE STACK
5376  022320  004737  022506                JSR     PC,@#$DB2D              ;GO CONVERT BINARY NO. TO
5377                                                                          ;ASCII STRING
5378  022324  004737  022352                JSR     PC,@#$SUPRS             ;GO TYPE OUT DECIMAL STRING
5379                                                                          ;SUPRESING LEADING 0'S
5380  022330  016666  000002  000004        MOV     2(SP),4(SP)             ;ADJUST RETURN
5381  022336  011666  000002               MOV     (SP),2(SP)              ;ADJUST RETURN ADRES
5382  022342  005726                       TST     (SP)+                   ;POP STACK
5383  022344  000002                       RTI                             ;RETURN
5384
5385  022346  000000  000000          1$:     .WORD   0,0
5386
5387
5388                                       .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
5389
5390                                       ;;*****************************************************************
5391                                       ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
5392                                       ;*LEADING NUMBERS.
5393                                       ;*CALL
5394                                       ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
5395                                       ;*      JSR     PC,@#$SUPRS
5396
5397
5398  022352  010046                   $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
5399  022354  016600  000004                   MOV     4(SP),R0        ;;PICKUP THE POINTER
5400  022360  105710                   1$:     TSTB    (R0)            ;;TERMINATEOR?
5401  022362  001403                           BEQ     2$              ;;BR IF YES
5402  022364  122720  000060                   CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
5403  022370  001773                           BEQ     1$              ;;BR IF YES
5404  022372  005300                   2$:     DEC     R0              ;;BACKUP BY "1"
5405  022374  010037  022402                   MOV     R0,3$           ;;SAVE FOR TYPING
5406  022400  104401                           TYPE                    ;;GO TYPE
5407  022402  000000                   3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
5408  022404  012600                           MOV     (SP)+,R0        ;;RESTORE R0
5409  022406  012616                           MOV     (SP)+,(SP)      ;;RESTORE THE STACK
5410  022410  000207                           RTS     PC              ;;RETURN
5411
5412                                       .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
5413
5414                                       ;;*****************************************************************
5415                                       ;*SAVE R0-R5
5416                                       ;*CALL:
5417                                       ;*      SAVREG
```

```
                                ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                ;*
                                ;*TOP---(+16)
                                ;* +2---(+18)
                                ;* +4---R5
                                ;* +6---R4
                                ;* +8---R3
                                ;*+10---R2
                                ;*+12---R1
                                ;*+14---R0
                                
022412                          $SAVREG:
022412  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
022414  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
022416  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
022420  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
022422  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
022424  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
022426  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
022432  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
022436  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
022442  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
022446  000002                          RTI
                                
                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
022450                          $RESREG:
022450  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
022454  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
022460  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
022464  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
022470  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
022472  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
022474  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
022476  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
022500  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
022502  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
022504  000002                          RTI
                                
                                .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
                                
                                ;;*****************************************************************
                                ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                                ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                ;*POSITIVE.
                                ;*CALL
                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                ;*      JSR     PC,@#$DB2D
                                ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                ;;IS ON THE STACK
                                
                                
                                $DB2D:  SAVREG                  ;;SAVE REGISTERS
022506  104413
022510  016602  000002                  MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
022514  012700  022666                  MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
```

```
5474  022520  010066  000002          MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
5475  022524  012201                  MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
5476  022526  012202                  MOV     (R2)+,R2
5477  022530  012737  000012  022604  MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
5478  022536  012704  022616          MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
5479  022542  012705  022620          MOV     #$TNPWR+2,R5
5480  022546  005003          1$:     CLR     R3              ;;CLEAR PARTIAL
5481  022550  161401          2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
5482  022552  005602                  SBC     R2
5483  022554  161502                  SUB     (R5),R2
5484  022556  002402                  BLT     3$              ;;BR IF TEN POWER TO LARGE
5485  022560  005203                  INC     R3              ;;ADD 1 TO PARTIAL
5486  022562  000772                  BR      2$              ;;LOOP
5487  022564  062401          3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
5488  022566  005502                  ADC     R2
5489  022570  062402                  ADD     (R4)+,R2
5490  022572  022525                  CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
5491  022574  052703  000060          BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
5492  022600  110320                  MOVB    R3,(R0)+        ;;SAVE IT
5493  022602  005327                  DEC     (PC)+           ;;DONE?
5494  022604  000000          4$:     .WORD   0
5495  022606  001357                  BNE     1$              ;;BR IF NO
5496  022610  105020                  CLRB    (R0)+           ;;TERMINATOR
5497  022612  104414                  RESREG                  ;;RESTORE REGISTERS
5498  022614  000207                  RTS     PC              ;;RETURN
5499  022616  145000          $TNPWR: 145000                  ;;1.0E09
5500  022620  035632                  35632
5501  022622  160400                  160400                  ;;1.0E08
5502  022624  002765                  2765
5503  022626  113200                  113200                  ;;1.0E07
5504  022630  000230                  230
5505  022632  041100                  041100                  ;;1.0E06
5506  022634  000017                  17
5507  022636  103240                  103240                  ;;1.0E05
5508  022640  000001                  1
5509  022642  023420                  23420                   ;;1.0E04
5510  022644  000000                  0
5511  022646  001750                  1750                    ;;1.0E03
5512  022650  000000                  0
5513  022652  000144                  144                     ;;1.0E02
5514  022654  000000                  0
5515  022656  000012                  12                      ;;1.0E01
5516  022660  000000                  0
5517  022662  000001                  1                       ;;1.0E00
5518  022664  000000                  0
5519  022666  000014          $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING
5520
5521
5522
5523                          .SBTTL  TRAP DECODER
5524
5525                          ;;**************************************************************
5526                          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
5527                          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
5528                          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
5529                          ;*GO TO THAT ROUTINE.
```

```
5530
5531  022702  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
5532  022704  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
5533  022710  005740                          TST     -(R0)           ;;BACKUP BY 2
5534  022712  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
5535  022714  006300                          ASL     R0              ;;POSITION FOR INDEXING
5536  022716  016000  022736                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
5537  022722  000200                          RTS     R0              ;;GO TO ROUTINE
5538
5539
5540                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
5541
5542  022724  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
5543  022726  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
5544  022734  000002                          RTI                     ;;RESTORE THE PSW
5545
5546                                  .SBTTL  TRAP TABLE
5547
5548                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
5549                                  ;*BY THE "TRAP" INSTRUCTION.
5550
5551                                  ;       ROUTINE
5552                                  ;       -------
5553  022736  022724                  $TRPAD: .WORD   $TRAP2
5554  022740  020260                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
5555  022742  022104                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
5556  022744  022060                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
5557  022746  022120                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
5558  022750  020034                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
5559
5560  022752  021154                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
5561
5562  022754  021064                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
5563  022756  021366                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
5564  022760  021456                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
5565  022762  021756                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
5566  022764  022412                          $SAVREG ;;CALL=SAVREG   TRAP+13(104413) SAVE R0-R5 ROUTINE
5567  022766  022450                          $RESREG ;;CALL=RESREG   TRAP+14(104414) RESTORE R0-R5 ROUTINE
5568
5569  022770  016602                          CN.RST  ;;CALL=CON.RESET        TRAP+15(104415) CONTROL RESET ROUTINE
5570
5571  022772  016470                          DR.RST  ;;CALL=DRV.RESET        TRAP+16(104416) DRIVE RESET ROUTINE
5572
5573  022774  016044                          MSGE    ;;CALL=MESAGE   TRAP+17(104417) MESSAGE HANDLER
5574
5575  022776  016106                          TY.MSG  ;;CALL=TYPMSG   TRAP+20(104420) MESSAGE TYPEOUT ROUTINE
5576
5577  023000  016620                          CN.RDY  ;;CALL=CON.RDY  TRAP+21(104421) WAIT FOR CONTROL READY
5578
5579  023002  016716                          TSTRWS  ;;CALL=TST.RWS  TRAP+22(104422) TEST R/W/S RDY SET
5580
5581  023004  016514                          RES.DO  ;;CALL=RESDON   TRAP+23(104423) DRIVE RESET DONE?
5582
5583  023006  022306                          TYPDES  ;;CALL=TYPDSS   TRAP+24(104424) TYPE DECIMAL, SUPRES LDG 0'S
5584
5585
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES
                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
023010  012737  023154  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
023016  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
023024  010046                          MOV     R0,-(SP)         ;;PUSH R0 ON STACK
023026  010146                          MOV     R1,-(SP)         ;;PUSH R1 ON STACK
023030  010246                          MOV     R2,-(SP)         ;;PUSH R2 ON STACK
023032  010346                          MOV     R3,-(SP)         ;;PUSH R3 ON STACK
023034  010446                          MOV     R4,-(SP)         ;;PUSH R4 ON STACK
023036  010546                          MOV     R5,-(SP)         ;;PUSH R5 ON STACK
023040  017746  156074                  MOV     @SWR,-(SP)       ;;PUSH @SWR ON STACK
023044  010637  023160                  MOV     SP,$SAVR6        ;;SAVE SP
023050  012737  023062  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
023056  000000                          HALT
023060  000776                          BR      .-2              ;;HANG UP
                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
023062  012737  023154  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
023070  013706  023160                  MOV     $SAVR6,SP        ;;GET SP
023074  005037  023160                  CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
023100  005237  023160          1$:     INC     $SAVR6           ;;WAIT FOR THE INC
023104  001375                          BNE     1$               ;;OF  WORD
023106  012677  156026                  MOV     (SP)+,@SWR       ;;POP STACK INTO @SWR
023112  012605                          MOV     (SP)+,R5         ;;POP STACK INTO R5
023114  012604                          MOV     (SP)+,R4         ;;POP STACK INTO R4
023116  012603                          MOV     (SP)+,R3         ;;POP STACK INTO R3
023120  012602                          MOV     (SP)+,R2         ;;POP STACK INTO R2
023122  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
023124  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
023126  012737  023010  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
023134  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
023142  104401                          TYPE                     ;REPORT THE POWER FAILURE
023144  023162          $PWRMG: .WORD   $POWER           ;;POWER FAIL MESSAGE POINTER
023146  012716                  MOV     (PC)+,(SP)       ;;RESTART AT PWRFL
023150  004244          $PWRAD: .WORD   PWRFL            ;;RESTART ADDRESS
023152  000002                  RTI
023154  000000          $ILLUP: HALT                     ;;THE POWER UP SEQUENCE WAS STARTED
023156  000776                  BR      .-2              ;; BEFORE THE POWER DOWN WAS COMPLE"E
023160  000000          $SAVR6: 0                        ;;PUT THE SP HERE
023162  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
023170  000122
                                        .EVEN
```

```
                        .SBTTL  FUNCTION SELECTION PROGRAM

                ;THIS IS THE FUNCTION SELECTION PROGRAM.
                ;ON ENTERING THIS SUB-PROGRAM THE FIRST QUESTION ASKED IS
                ;       FUNCTION?       IN REPLY TYPE IN ONE OF THE FOLLOWING COMMANDS
                ;COMMANDS:      CR - CONTROL RESET
                ;               DR - DRIVE RESET
                ;               SK - SEEK
                ;               WR - WRITE
                ;               RD - READ
                ;               WC - WRITE CHECK
                ;               RC - READ CHECK
                ;TERMINATE EVERY COMMAND WITH <CR>. FURTHER QUESTIONS (RKBA?   RKDA?
                ;#WORDS? ) WILL BE ASKED. TYPE IN THE BUS ADDRESS (OCTAL), DISK ADDRESS
                ;(OCTAL), AND NUMBER OF WORDS TO BE TRANSFERRED (OCTAL). IF A NON-EXISTENT
                ;CYLINDER OR A NON-EXISTENT SECTOR IS TYPED IN, THE QUESTION (RKDA?) WILL
                ;BE ASKED AGAIN.

                ;IN CASE OF SEEK FUNCTION, SEEK IS DONE BETWEEN A SET OF CYLINDERS GIVEN
                ;BY THE USER  (CYL1? , CYL2?). IN REPLY TO (CYL1?, CYL2?) TYPE IN THE OCTAL
                ;CYLINDER NUMBERS BETWEEN WHICH THE SEEK IS TO BE DONE. SET SWITCH
                ;REGISTER BITS <15-13> TO THE DRIVE # ON WHICH SEEK IS TO BE DONE.

                ;IN CASE OF A WRITE FUNCTION IF SW 0 IS SET TO 1 THE PROGRAM WILL ASK
                ;THE USER FOR THE PATTERN TO BE WRITTEN:
                ;       PATRN?125252<CR>
                ;THE USER SHOULD TYPE IN THE (OCTAL) PATTERN HE WANTS TO WRITE.
                ;NOTE THAT THE NUMBER OF WORDS TRANSFERRED SHOULD BE WITHIN THE
                ;BOUNDS OF THE SYSTEM.

                ;IN CASE OF A WRITE CHECK FUNCTION IF SW0 IS SET TO 1 THE PROGRAM
                ;WILL ASK THE USER FOR THE PATTERN TO BE WRITE CHECKED:
                ;       PATRN?125252<CR>
                ;THE USER SHOULD TYPE IN THE (OCTAL) PATTERN TO BE WRITE CHECKED.

                ;LOCATIONS "IOBUF0" ONWARDS  ARE RESERVED FOR DATA BUFFERS. YOU CAN USE
                ;THESE LOCATIONS FOR DOING DATA TRANSFERS IN THIS PROGRAM (OR YOU CAN
                ;USE ANY OTHER BUFFER FOR DATA TRANSFER AS LONG AS IT DOES NOT OVERLAY
                ;THE PROGRAM).

                ;THE SAME FUNCTION IS PERFORMED AGAIN AND AGAIN, THUS PROVIDING
                ;A VERY GOOD SCOPE LOOP. IF YOU WANT TO GIVE A NEW FUNCTION PUT SW 3 UP.
                ;THE QUESTION (FUNCTION?) WILL BE ASKED AND YOU CAN START ALL OVER AGAIN.
                ;IF ON EXECUTING A FUNCTION AN ERROR OCCURS, IT IS REPOTRED , WITH
                ; RELEVANT REGISTER CONTENTS GIVEN.

                                                ;R2 CONTAINS RKCS CONTENTS
                                                ;R3 CONTAINS RKDA CONTENTS
                                                ;R4,R5 CONTAIN THE CYLINDER #S BETWEEN
                                                ;WHICH SEEK IS  TO BE DONE.
023172          FUNBEG:
023172  104401  023200          TYPE    ,65$            ;;TYPE ASCIZ STRING
023176  000407                  BR      64$             ;;GET OVER THE ASCIZ
                ;;65$:  .ASCIZ  <15><12>/FUNCTION? /
023216          64$:
023216  104411                  RDLIN
```

```
5689  023220  012600                  MOV     (SP)+,R0
5690  023222  112001                  MOVB    (R0)+,R1
5691  023224  120127  000127          CMPB    R1,#'W
5692  023230  001026                  BNE     2$
5693  023232  121027  000122          CMPB    (R0),#'R
5694  023236  001010                  BNE     1$
5695  023240  004737  024064          JSR     PC,CHKSW0       ;CHECK SW 0 SET?
5696  023244  012702  004003          MOV     #4003,R2
5697  023250  000536                  BR      NXTDA
5698
5699  023252  012702  000003  9$:     MOV     #3,R2
5700  023256  000521                  BR      NXTBA
5701  023260  121027  000103  1$:     CMPB    (R0),#'C
5702  023264  001342                  BNE     FUNBEG
5703  023266  004737  024064          JSR     PC,CHKSW0       ;CHECK SW 0 SET?
5704  023272  012702  004007          MOV     #4007,R2
5705  023276  000523                  BR      NXTDA
5706
5707  023300  012702  000007          MOV     #7,R2
5708  023304  000506                  BR      NXTBA
5709
5710  023306  120127  000122  2$:     CMPB    R1,#'R
5711  023312  001014                  BNE     3$
5712  023314  121027  000104          CMPB    (R0),#'D
5713  023320  001003                  BNE     8$
5714  023322  012702  000005          MOV     #5,R2
5715  023326  000475                  BR      NXTBA
5716  023330  121027  000103  8$:     CMPB    (R0),#'C
5717  023334  001316                  BNE     FUNBEG
5718  023336  012702  000013          MOV     #13,R2
5719  023342  000501                  BR      NXTDA
5720
5721  023344  120127  000104  3$:     CMPB    R1,#'D
5722  023350  001006                  BNE     4$
5723  023352  121027  000122          CMPB    (R0),#'R
5724  023356  001305                  BNE     FUNBEG
5725  023360  012702  000015          MOV     #15,R2
5726  023364  000470                  BR      NXTDA
5727
5728  023366  120127  000103  4$:     CMPB    R1,#'C
5729  023372  001006                  BNE     5$
5730  023374  121027  000122          CMPB    (R0),#'R
5731  023400  001274                  BNE     FUNBEG
5732  023402  012702  000001          MOV     #1,R2
5733  023406  000533                  BR      EXEC
5734
5735  023410  120127  000123  5$:     CMPB    R1,#'S
5736  023414  001256                  BNE     FUNBEG
5737  023416  121027  000113          CMPB    (R0),#'K
5738  023422  001263                  BNE     FUNBEG
5739  023424  012702  000011          MOV     #11,R2
5740
5741  023430                  6$:
5742  023430  104401  023436          TYPE    ,67$            ;;TYPE ASCIZ STRING
5743  023434  000404                  BR      66$             ;;GET OVER THE ASCIZ
5744                          ;;67$:  .ASCIZ  /CYL1? /
```

```
5745  023446                          66$:
5746  023446  004737  024032                  JSR     PC,INPT
5747  023452  000766                          BR      6$
5748  023454  010004                          MOV     R0,R4
5749
5750  023456                          7$:
5751  023456  104401  023464                  TYPE    ,69$            ;;TYPE ASCIZ STRING
5752  023462  000404                          BR      68$             ;;GET OVER THE ASCIZ
5753                                  ;;69$:  .ASCIZ  /CYL2? /
5754  023474                          68$:
5755  023474  004737  024032                  JSR     PC,INPT
5756  023500  000766                          BR      7$
5757  023502  010005                          MOV     R0,R5
5758  023504  017700  155430                  MOV     @#SWR,R0        ;GET DRIVE # FROM SW REG<15-13>
5759  023510  042700  017777                  BIC     #17777,R0               ;CLR UNWANTED BITS
5760  023514  050004                          BIS     R0,R4           ;SET DRIVE # IN DSK ADRES
5761  023516  050005                          BIS     R0,R5
5762  023520  000466                          BR      EXEC
5763
5764
5765  023522                          NXTBA:
5766  023522  104401  023530                  TYPE    ,65$            ;;TYPE ASCIZ STRING
5767  023526  000404                          BR      64$             ;;GET OVER THE ASCIZ
5768                                  ;;65$:  .ASCIZ  /RKBA? /
5769  023540                          64$:
5770  023540  104412                          RDOCT
5771  023542  012637  001476                  MOV     (SP)+,INDX2
5772
5773  023546                          NXTDA:
5774  023546  104401  023554                  TYPE    ,65$            ;;TYPE ASCIZ STRING
5775  023552  000404                          BR      64$             ;;GET OVER THE ASCIZ
5776                                  ;;65$:  .ASCIZ  /RKDA? /
5777  023564                          64$:
5778  023564  104412                          RDOCT
5779  023566  012600                          MOV     (SP)+,R0
5780  023570  010001                          MOV     R0,R1
5781  023572  006201                          ASR     R1
5782  023574  006201                          ASR     R1
5783  023576  006201                          ASR     R1
5784  023600  006201                          ASR     R1
5785  023602  006201                          ASR     R1
5786  023604  042701  177400                  BIC     #177400,R1
5787  023610  020127  000312                  CMP     R1,#312
5788  023614  003354                          BGT     NXTDA
5789  023616  010001                          MOV     R0,R1
5790  023620  042701  177760                  BIC     #177760,R1
5791  023624  020127  000013                  CMP     R1,#13
5792  023630  003346                          BGT     NXTDA
5793  023632  010003                          MOV     R0,R3
5794  023634  022702  000015                  CMP     #15,R2
5795  023640  001416                          BEQ     EXEC
5796
5797
5798  023642                          NXTWC:
5799  023642  104401  023650                  TYPE    ,65$            ;;TYPE ASCIZ STRING
5900  023646  000405                          BR      64$             ;;GET OVER THE ASCIZ
```

```
5801                                    ;;65$:  .ASCIZ  /#WORDS? /
5802  023662                            64$:
5803  023662  104412                            RDOCT
5804  023664  005416                            NEG     (SP)
5805  023666  012637  001502                    MOV     (SP)+,INDX4
5806  023672  000401                            BR      EXEC
5807
5808  023674  104415                    EXEC1:  CON.RESET                       ;CLR EROR, CONTROL RESET
5809  023676  022702  000011            EXEC:   CMP     #11,R2                  ;SEEK FUNCTION?
5810  023702  001005                            BNE     2$                      ;NO
5811  023704  020403                            CMP     R4,R3                   ;IF SEEK, INSERT THE RIGHT
5812  023706  001402                            BEQ     3$                      ;CYLINDER ADDRESS
5813  023710  010403                            MOV     R4,R3
5814  023712  000401                            BR      2$
5815  023714  010503                    3$:     MOV     R5,R3
5816  023716  013777  001476  155536  2$:     MOV     INDX2,@RKBA
5817  023724  010377  155534                    MOV     R3,@RKDA
5818  023730  013777  001502  155522          MOV     INDX4,@RKWC
5819  023736  010277  155514                    MOV     R2,@RKCS
5820  023742  105777  155510            1$:     TSTB    @RKCS                   ;WAIT FOR CNTROL RDY
5821  023746  100375                            BPL     1$
5822  023750  022702  000001                    CMP     #1,R2                   ;IF IT'S CON RESET FUNCTION
5823  023754  001401                            BEQ     4$                      ;DONT WAIT FOR R/W/S RDY
5824  023756  104423                            RESDON                          ;R/W/S RDY?
5825
5826  023760  032777  140000  155470  4$:     BIT     #140000,@RKCS           ;ERROR?
5827  023766  001006                            BNE     FUNERR                  ;YES
5828  023770  032777  000010  155142  CHSW:   BIT     #SW3,@SWR               ;TERMINATE THIS FUNCTION OR REPEAT?
5829  023776  001737                            BEQ     EXEC                    ;REPEAT
5830  024000  000137  023172                    JMP     FUNBEG                  ;TERMINATE
5831
5832
5833  024004  012737  023674  001110  FUNERR: MOV     #EXEC1,$LPERR           ;SET UP FOR LUPING
5834  024012  012737  023674  001106          MOV     #EXEC1,$LPADR
5835  024020  004737  016162                    JSR     PC,GT4RG
5836  024024  104030                            ERROR   30                      ;REPORT ERROR
5837  024026  104415                            CON.RESET                       ;CLR ERROR
5838  024030  000757                            BR      CHSW
5839
5840  024032  104412                    INP":   RDOCT
5841  024034  012600                            MOV     (SP)+,R0
5842  024036  020027  000312                    CMP     R0,#312
5843  024042  003007                            BGT     1$
5844  024044  006300                            ASL     R0
5845  024046  006300                            ASL     R0
5846  024050  006300                            ASL     R0
5847  024052  006300                            ASL     R0
5848  024054  006300                            ASL     R0
5849  024056  062716  000002                    ADD     #2,(SP)
5850  024062  000207                    1$:     RTS     PC
5851
5852  024064  032777  000001  155046  CHKSWD: BIT     #SWD,@SWR               ;WRITE A PATTERN GIVEN BY USER?
5853  024072  001416                            BEQ     1$                      ;NO
5854                                                                            ;YES, ASK FOR PATTERN
5855  024074  104401  024102                    TYPE    ,65$                    ;;TYPE ASCIZ STRING
5856  024100  000404                            BR      64$                     ;;GET OVER THE ASCIZ
```

```
5857                                        ;;65$:  .ASCIZ  /PATRN?/
5858  024112                                64$:
5859  024112  104412                              RDOCT
5860  024114  012637  031446                      MOV     (SP)+,IOBUF1    ;SAVE THE PATTERN
5861  024120  012737  031446  001476              MOV     #IOBUF1,INDX2
5862  024126  000207                              RTS     PC
5863  024130  062716  000006            1$:       ADD     #6,(SP)         ;SKIP NXT 2 INST ON RETURN
5864  024134  000207                              RTS     PC
5865
5866  024136  104416                    FCHECK:   DRV.RESET
5867  024140  104415                              CON.RESET
5868  024142  013737  001230  002120              MOV     DRIVAD,DRHOLD           ;SAVE DRIVE ADRR
5869  024150  032737  020000  001230              BIT     #20000,DRIVAD           ;SEE IF ODD
5870  024156  001404                              BEQ     1$
5871  024160  042737  020000  001230              BIC     #20000,DRIVAD           ;MAKE EVEN
5872  024166  000403                              BR      2$
5873  024170  052737  020000  001230  1$:       BIS     #20000,DRIVAD           ;MAKE ODD
5874  024176  013777  001230  155260  2$:       MOV     DRIVAD,@RKDA    ;DRIVE ADDR
5875  024204  012777  000011  155244              MOV     #11,@RKCS               ;DRIVE SEEK
5876  024212  104421                              CON.RDY
5877  024214  013777  002120  155242              MOV     DRHOLD,@RKDA            ;OTHER DRIVE
5878  024222  104421                              CON.RDY
5879  024224  032777  000100  155220              BIT     #100,@RKDS              ;HEAEDS IN MOTIONN?
5880  024232  001001                              BNE     3$                      ;NO SO RK-05J
5881  024234  005725                              TST     (R5)+                   ;YES RK-05F
5882  024236  013737  002120  001230  3$:       MOV     DRHOLD,DRIVAD           ;RESTORE ADDR
5883  024244  104416                              DRV.RESET
5884  024246  000205                              RTS     R5
5885  024250  005037  001230            SIZEF:    CLR     DRIVAD          ;START AT DRO
5886  024254  012700  001232                      MOV     #DRIVO,R0               ;TABLE OF AVAIL DRIVES
5887  024260  105710                    4$:       TSTB    (R0)                    ;THIS DRIVE HERE?
5888  024262  001413                              BEQ     2$                      ;NO
5889  024264  105760  000002                      TSTB    2(R0)                   ;COMPLEMENT HERE?
5890  024270  001410                              BEQ     2$                      ;NO
5891  024272  004537  024136                      JSR     R5,FCHECK               ;SEE IF F MODEL
5892  024276  000405                              BR      2$                      ;J MODEL
5893  024300  052710  000002                      BIS     #2,(R0)         ;SET SIGN FOR F
5894  024304  052760  000002  000002              BIS     #2,2(R0)                ;BOTH DRIVES
5895  024312  005720                    2$:       TST     (R0)+
5896  024314  005720                              TST     (R0)+                   ;NEXT PAIR OF DRIVES
5897  024316  062737  040000  001230              ADD     #40000,DRIVAD           ;NEXT ACTUL ADDR
5898  024324  022700  001251                      CMP     #DRIV7+1,R0             ;CHECKED ALL?
5899  024330  003353                              BGT     4$                      ;NOT YET
5900  024332  000207                              RTS     PC
5901
5902                                        ;ERROR MESSAGES
5903
5904
5905  024334  047103  051124  020114  EM1:      .ASCIZ  /CNTRL RDY DIDN'T SET AFTR SEEK/
5906  024342  042122  020131  044504
5907  024350  047104  052047  051440
5908  024356  052105  040440  052106
5909  024364  020122  042523  045505
5910  024372     000
5911
5912  024373     123  047111  047440  EM2:      .ASCIZ  /SIN ON SEEK/
```

```
5913  024400  020116  042523  045505
5914  024406     000
5915
5916  024407     104  042522  047440  EM3:    .ASCIZ  /DRE ON SEEK/
5917  024414  020116  042523  045505
5918  024422     000
5919
5920  024423     047  051105  023522  EM4:    .ASCIZ  /'ERR' ON SEEK/
5921  024430  047440  020116  042523
5922  024436  045505     000
5923
5924  024441     104  052522  047440  EM5:    .ASCIZ  /DRU ON SEEK, PUT DRV ON 'LOAD' & 'RUN'/
5925  024446  020116  042523  045505
5926  024454  020054  052520  020124
5927  024462  051104  020126  047117
5928  024470  023440  047514  042101
5929  024476  020047  020046  051047
5930  024504  047125  000047
5931
5932  024510  027522  027527  020123  EM6:    .ASCIZ  "R/W/S RDY NOT SET AFTER SEEK"
5933  024516  042122  020131  047516
5934  024524  020124  042523  020124
5935  024532  043101  042524  020122
5936  024540  042523  045505     000
5937
5938  024545     123  047111  047440  EM7:    .ASCIZ  /SIN ON WRT FMT/
5939  024552  020116  051127  020124
5940  024560  046506  000124
5941
5942  024564  042447  051122  020047  EM10:   .ASCIZ  /'ERR' ON DOING WRITE FMT/
5943  024572  047117  042040  044517
5944  024600  043516  053440  044522
5945  024606  042524  043040  052115
5946  024614     000
5947  024615     123  047111  047440  EM11:   .ASCIZ  "SIN ON RD/FMT"
5948  024622  020116  042122  043057
5949  024630  052115     000
5950  024633     047  051105  023522  EM12:   .ASCIZ  /'ERR' ON READ FMT/
5951  024640  047440  020116  042522
5952  024646  042101  043040  052115
5953  024654     000
5954  024655     127  047522  043516  EM13:   .ASCIZ  /WRONG HDRS FROM 'SEC#'/
5955  024662  044040  051104  020123
5956  024670  051106  046517  023440
5957  024676  042523  021503  000047
5958
5959  024704  051105  051117  047440  EM14:   .ASCIZ  /EROR ON IMPLIED SEEK FROM 'CYLA' TO 'CYLB'
5960  024712  020116  046511  046120
5961  024720  042511  020104  042523
5962  024726  045505  043040  047522
5963  024734  020115  041447  046131
5964  024742  023501  052040  020117
5965  024750  041447  046131  023502
5966  024756     000
5967
5968  024757     122  040505  020104  MS15:   .ASCIZ  /READ WRONG HDRS FROM 'CYLB' ABOVE/
```

```
 5969  024764  051127  047117  020107
 5970  024772  042110  051522  043040
 5971  025000  047522  020115  041447
 5972  025006  046131  023502  040440
 5973  025014  047502  042526     000
 5974
 5975  025021     122  040505  020104  EM16:   .ASCIZ  /READ WRONG, 1ST WRD FROM SEC 0, 'CYLB' (ON IMPLIED SEEK FROM 'CYLA')/
 5976  025026  051127  047117  026107
 5977  025034  030440  052123  053440
 5978  025042  042122  043040  047522
 5979  025050  020115  042523  020103
 5980  025056  026060  023440  054503
 5981  025064  041114  020047  047450
 5982  025072  020116  046511  046120
 5983  025100  042511  020104  042523
 5984  025106  045505  043040  047522
 5985  025114  020115  041447  046131
 5986  025122  023501  000051
 5987
 5988  025126  042522  042101  030440  MS17:   .ASCIZ  /READ 1ST WRD FROM SEC 1, 'CYLB' ABOVE/
 5989  025134  052123  053440  042122
 5990  025142  043040  047522  020115
 5991  025150  042523  020103  026061
 5992  025156  023440  054503  041114
 5993  025164  020047  041101  053117
 5994  025172  000105
 5995
 5996  025174  042522  042101  053440  EM20:   .ASCIZ  /READ WRONG HDRS ON IMPLIED SEEK FROM 'CYLA' TO 'CYLB'/
 5997  025202  047522  043516  044040
 5998  025210  051104  020123  047117
 5999  025216  044440  050115  044514
 6000  025224  042105  051440  042505
 6001  025232  020113  051106  046517
 6002  025240  023440  054503  040514
 6003  025246  020047  047524  023440
 6004  025254  054503  041114  000047
 6005
 6006  025262  051105  051117  047440  EM21:   .ASCIZ  /EROR ON DOING WRITE ON DISK/
 6007  025270  020116  047504  047111
 6008  025276  020107  051127  052111
 6009  025304  020105  047117  042040
 6010  025312  051511  000113
 6011
 6012  025316  044523  020116  047117  EM22:   .ASCIZ  /SIN ON DOING WRITE/
 6013  025324  042040  044517  043516
 6014  025332  053440  044522  042524
 6015  025340     000
 6016
 6017  025341     110  020105  047117  EM23:   .ASCIZ  /HE ON DOING READ/
 6018  025346  042040  044517  043516
 6019  025354  051040  040505  000104
 6020
 6021  025362  051503  020105  047117  EM24:   .ASCIZ  /CSE ON READ/
 6022  025370  051040  040505  000104
 6023
 6024  025376  040504  040524  042440  EM25:   .ASCIZ  /DATA EROR ON READ FROM DISK ADRES/
```

```
6025  025404  047522  020122  047117
6026  025412  051040  040505  020104
6027  025420  051106  046517  042040
6028  025426  051511  020113  042101
6029  025434  042522  000123
6030
6031  025440  042510  047440  020116  EM26:   .ASCIZ  /HE ON WRT CHK/
6032  025446  051127  020124  044103
6033  025454  000113
6034
6035  025456  051127  020124  044103  EM27:   .ASCIZ  /WRT CHK EROR/
6036  025464  020113  051105  051117
6037  025472     000
6038
6039  025473     105  047522  000122  EM30:   .ASCIZ  /EROR/
6040
6041                                   ;DATA HEADERS
6042
6043
6044  025500  020040  041520  020040  DH1:    .ASCIZ  /  PC     RKCS    RKER    RKDS .  RKDA/
6045  025506  020040  051040  041513
6046  025514  020123  020040  051040
6047  025522  042513  020122  020040
6048  025530  051040  042113  020123
6049  025536  020040  051040  042113
6050  025544  000101
6051
6052  025546  042523  045505  021440  DH6:    .ASCIZ  /SEEK #  SEEKING  ERRORS/
6053  025554  020040  042523  045505
6054  025562  047111  020107  042440
6055  025570  051122  051117  000123
6056
6057  025576  020040  041520  020040  DH7:    .ASCIZ  /  PC     RKCS    RKER    RKDS    RKDA    CYL/
6058  025604  020040  051040  041513
6059  025612  020123  020040  051040
6060  025620  042513  020122  020040
6061  025626  051040  042113  020123
6062  025634  020040  051040  042113
6063  025642  020101  020040  041440
6064  025650  046131     000
6065
6066  025653     040  050040  020103  DH11:   .ASCIZI/  PC    RKCS    RKER    RKDS   RKDA: DR  CYL     SUR     SEC/
6067  025660  020040  051040  041513
6068  025666  020123  020040  051040
6069  025674  042513  020122  020040
6070  025702  051040  042113  020123
6071  025710  020040  045522  040504
6072  025716  020072  051104  020040
6073  025724  054503  020114  020040
6074  025732  020040  052523  020122
6075  025740  020040  020040  042523
6076  025746  000103
6077  025750  042523  021503  020040  DH13:   .ASCIZ  /SEC#  HDR RCVD/
6078  025756  042110  020122  041522
6079  025764  042126     000
6080
```

```
6081  025767  040     050040  020103  DH14:   .ASCIZ  / PC     CYLA    CYLB     RKER   RKDS     TR#/
6082  025774  020040  020040  054503
6083  026002  040514  020040  020040
6084  026010  054503  041114  020040
6085  026016  020040  051040  042513
6086  026024  020122  020040  045522
6087  026032  051504  020040  020040
6088  026040  052040  054522  000043
6089
6090  026046  020040  041520  020040  DH16:   .ASCIZ  / PC     CYLA    CYLB    EXPCT   RECVD   TRY#/
6091  026054  020040  041440  046131
6092  026062  020101  020040  054503
6093  026070  041114  020040  020040
6094  026076  054105  041520  020124
6095  026104  020040  042522  053103
6096  026112  020104  020040  051124
6097  026120  021531     000
6098
6099  026123     040  050040  020103  DH17:   .ASCIZ  / PC     CYLB    EXPCT   RECVD/
6100  026130  020040  020040  054503
6101  026136  041114  020040  042440
6102  026144  050130  052103  020040
6103  026152  051040  041505  042126
6104  026160     000
6105
6106  026161     040  050040  020103  DH24:   .ASCIZ/ PC     TRY#    RKCS    RKER    RKDS  RKDA: DR  CYL     SUR     SEC/
6107  026166  020040  020040  051124
6108  026174  021531  020040  051040
6109  026202  041513  020123  020040
6110  026210  051040  042513  020122
6111  026216  020040  051040  042113
6112  026224  020123  051040  042113
6113  026232  035101  042040  020122
6114  026240  041440  046131  020040
6115  026246  020040  051440  051125
6116  026254  020040  020040  020040
6117  026262  042523  000103
6118
6119  026266  047527  042122  020043  DH25:   .ASCIZ  /WORD#  EXPCT   RECVD   CYL  SUR  SEC/
6120  026274  042440  050130  052103
6121  026302  020040  051040  041505
6122  026310  042126  020040  041440
6123  026316  046131  020040  052523
6124  026324  020122  051440  041505
6125  026332     000
6126
6127
6128                                  ;ERROR DATA POINTERS
6129
6130
6131          026334                  .EVEN
6132
6133  026334  001116  001162  001164  DT1:    .WORD   $ERRPC,$REG0,$REG1,$REG2,$REG3,0
6134  026342  001166  001170  000000
6135
6136  026350  001116  001162  001164  DT7:    .WORD   $ERRPC,$REG0,$REG1,$REG2,$REG3,$REG4,0
```

```
 6137  026356  001166  001170  001172
 6138  026364  000000
 6139
 6140  026366  001116  001162  001164  DT11:   .WORD   $ERRPC,$REG0,$REG1,$REG2,$REG4,$REG5,$REG6,$REG7,0
 6141  026374  001166  001172  001174
 6142  026402  001176  001200  000000
 6143
 6144  026410  001116  001162  001164  DT17:   .WORD   $ERRPC,$REG0,$REG1,$REG2,0
 6145  026416  001166  000000
 6146
 6147  026422  001116  001202  001162  DT24:   .WORD   $ERRPC,$REG10,$REG0,$REG1,$REG2,$REG4,$REG5,$REG6,$REG7,0
 6148  026430  001164  001166  001172
 6149  026436  001174  001176  001200
 6150  026444  000000
 6151
 6152                                  ;DATA BUFFERS
 6153
 6154  026446                          IOBUF0:
 6155  026446  000030                  OUTBUF: .BLKW   24.      ;IOBUF0 AND IOBUF1 ARE
 6156  026526  000030                  BUFR4:  .BLKW   24.      ;TWO - 768 WORDS LONG BUFFERS
 6157  026606  000030                  BUFR5:  .BLKW   24.      ;NORMALLY USED FOR DATA TRANSFERS
 6158  026666  000030                  BUFR6:  .BLKW   24.      ;TO AND FROM DISK
 6159  026746  000030                  BUFR7:  .BLKW   24.
 6160  027026  000200                  BUFR10: .BLKW   128.
 6161  027426  000200                  BUFR11: .BLKW   128.
 6162  030026  000200                  BUFR12: .BLKW   128.
 6163  030426  000200                  BUFR13: .BLKW   128.
 6164  031026  000210                  BUFR14: .BLKW   136.
 6165
 6166  031446  001400                  IOBUF1: .BLKW   768.
 6167
 6168
 6169
 6170          000001                          .END
```

```
ABBI    016772
ADRES   001450
BADTMO  002422
BEGIN   010532
BEGSK   013250
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05 = 000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BLNKS1  002111
BLNKS2  002110
BLNKS3  002107
BLNKS4  002106
BLNKS5  002105
BLNKS6  002104
BLNKS7  002103
BLNKS8  002102
BLNKS9  002101
BLNK10  002100
BLNK13  002075
BPTVEC= 000014
BRKDA   016220
BTEOP   015232
BUFLG0  001410
BUFLG1  001412
BUFNO   001446
BUFR    001266
BUFR1   001320
BUFR10  027026
BUFR11  027426
BUFR12  030026
BUFR13  030426
BUFR14  031026
BUFR2   001352
BUFR4   026526
BUFR5   026606
BUFR6   026666
BUFR7   026746
BUSADR  001434
CHKHDR  007260
CHKSWD  024064
CHSW    023770
CKSWR = 104407
CLRERR  012162
CMPAGA  011244
CMPGRP  014622
CN.RDY  016620
CN.RST  016602
COMPAR  011220
CON.RD= 104421
CON.RE= 104415
CR    = 000015
CRLF  = 000200
CSEROR  011654
DDISP = 177570
DH1     025500
DH11    025653
DH13    025750
DH14    025767
DH16    026046
DH17    026123
DH24    026161
DH25    026266
DH6     025546
DH7     025576
DISPLA  001142
DISPRE  000174
DMPREG  017662
DONE    011636
DONTRK  011576
DOSUR1  011630
DOWCHK  012420
DOWRT   012224
DRHOLD  002120
DRIVAD  001230
DRIVS   001224
DRIV0   001232
DRIV1   001234
DRIV2   001236
DRIV3   001240
DRIV4   001242
DRIV5   001244
DRIV6   001246
DRIV7   001250
DRVDON  001223
DRVPTR  001226
DRV.RE= 104416
DR.RST  016470
DSKADR  001432
DSWR  = 177570
DT1     026334
DT11    026366
DT17    026410
DT24    026422
DT7     026350
EMTVEC= 000030
EM1     024334
EM10    024564
EM11    024615
EM12    024633
EM13    024655
EM14    024704
EM16    025021
EM2     024373
EM20    025174
EM21    025262
EM22    025316
EM23    025341
EM24    025362
EM25    025376
EM26    025440
EM27    025456
EM3     024407
EM30    025473
EM4     024423
EM5     024441
EM6     024510
EM7     024545
ERCNT1  001504
ERCNT2  001506
ERCNT3  001510
ERCNT4  001512
ERCNT5  001514
ERCNT6  001516
ERCNT7  001520
ERCNT8  001522
ERRTYP  017446
ERRVEC= 000004
ERR1    016376
ERR2    016320
ERWCHK  011404
ESR13   015456
ESR15   015364
ESR20   015532
ESR25   015620
EXEC    023676
EXEC1   023674
EXIIO   012732
EXIQ    006614
EXT5    007352
EXT6    010432
EXT7    012170
FCHECK  024136
FDRIVE  002114
FDRVE1  002116
FFUNC   001216
FINISH  011512
FUNBEG  023172
FUNERR  024004
GCYL    016434
GETBUF  010044
GETINF  016214
GOBAK   011650
GOTYPE  013600
GTSWR = 104406
GT4RG   016162
GT5RG   016136
HT    = 000011
INADR   001260
INDX1   001474
INDX2   001476
INDX3   001500
INDX4   001502
INPT    024032
IOBUF0  026446
IOBUF1  031446
IOTVEC= 000020
LF    = 000012
LUPHE   010640
LUPSIN  010632
LUPSW   001222
LUPWCE  011350
MESAGE= 104417
MISCMP  012016
MSGE    016044
MSG1    001602
MSG10   001742
MSG11   001772
MSG12   002027
MSG13   002053
MSG14   002064
MSG2    001610
MSG3    001616
MSG4    001640
MSG5    001657
MSG6    001716
MSG7    001733
MS15    024757
MS17    025126
NOHE    010760
NOSIN   011026
NXTBA   023522
NXTDA   023546
NXTDRV  003760
NXTPAT  001426
NXTWC   023642
OUTADR  001262
OUTBUF  026446
PAT0    001414
PAT1    001416
PAT2    001420
PAT3    001422
PBUF0   001404
PBUF1   001406
PGSUBR  001430
PIRQ  = 177772
PIRQVE= 000240
PLOT    014222
PLTGRP  014106
PLTPT   014736
PLT1    014436
PRSPAT  001424
PR0   = 000000
PR1   = 000040
PR2   = 000100
PR3   = 000140
PR4   = 000200
PR5   = 000240
PR6   = 000300
PR7   = 000340
PS    = 177776
PSW   = 177776
PTGEN0  010106
PTGEN1  010170
PTGEN2  010272
PTGEN3  010334
PTRN01  010164
PTRN02  010166
PT1     001560
PT10    001576
PT11    001600
PT2     001562
PT3     001564
PT4     001566
PT5     001570
PT6     001572
PT7     001574
PWRFL   004244
PWRVEC= 000024
RDAGAI  010642
RDCHR = 104410
RDLIN = 104411
RDOCT = 104412
```

```
READ    010606          STKLMT= 177774          TST7    010436          $ENULL  015344          $REG3   001170
REPEAT  012666          ST2     003512          TYPDES  022306          $EOP    015246          $REG4   001172
REPMSC  011274          ST3     003742          TYPDS = 104405          $EOPCT  015270          $REG5   001174
REPTIM  013200          SWR     001140          TYPDSS= 104424          $ERFLG  001103          $REG6   001176
RESDON= 104423          SWREG   000176          TYPE  = 104401          $ERMAX  001115          $REG7   001200
RESREG= 104414          SW0   = 000001          TYPMSG= 104420          $ERROR  017162          $RESRE  022450
RESVEC= 000010          SW00  = 000001          TYPOC = 104402          $ERRPC  001116          $RTNAD  015342
RES.DO  016514          SW01  = 000002          TYPON = 104404          $ERRTB  002122          $SAVRE  022412
RETRY1  001252          SW02  = 000004          TYPOS = 104403          $ERTTL  001112          $SAVR6  023160
RETRY2  001254          SW03  = 000010          TYPTIM  013736          $ESCAP  001204          $SCOPE  017006
RETRY3  001256          SW04  = 000020          TY.MSG  016106          $FILLC  001156          $SETUP= 000117
RKBA    001462          SW05  = 000040          WBUSAD  001442          $FILLS  001155          $STUP = 177777
RKCS    001456          SW06  = 000100          WCAGAI  011132          $GDADR  001120          $SUPRS  022352
RKDA    001464          SW07  = 000200          WCEROR  012062          $GDDAT  001124          $SVLAD  017120
RKDB    001466          SW08  = 000400          WCERR   012442          $GET42  015320          $SVPC = 000220
RKDS    001452          SW09  = 001000          WCHI    012446          $GTSWR  021154          $SWR  = 163000
RKER    001454          SW1   = 000002          WCHI1   012440          $HD   = 000000          $SWRMK= 000000
RKPRI   001470          SW10  = 002000          WCLO    012646          $HIOCT  022056          $TKB    001146
RKVEC   001472          SW11  = 004000          WCREPT  011334          $ICNT   001104          $TKCNT  020612
RKWC    001460          SW12  = 010000          WDSKAD  001440          $ILLUP  023154          $TKINT  020622
R6    =%000006          SW13  = 020000          WRDCNT  001436          $INTAG  001135          $TKQEN= 020621
R7    =%000007          SW14  = 040000          WRERR   012240          $ITEMB  001114          $TKQIN  020614
SAVREG= 104413          SW15  = 100000          WRHI    012400          $LF     001214          $TKQOU  020616
SBR1    006526          SW2   = 000004          WRLO    012242          $LPADR  001106          $TKQSR  020620
SBR2    006552          SW3   = 000010          WRLO1   012236          $LPERR  001110          $TKS    001144
SBR3    007224          SW4   = 000020          WRTCHK  011116          $MNEW   021745          $TKSRV  020672
SECPTR  010426          SW5   = 000040          WWRDCN  001444          $MSWR   021734          $TN   = 000013
SFTCMP  011166          SW6   = 000100          XHE     011442          $MUL  = 000003          $TNPWR  022616
SIAD    001542          SW7   = 000200          XXDPMD  001220          $MULT   020500          $TPB    001152
SIZEF   024250          SW8   = 000400          $AUTOB  001134          $NULL   001154          $TPFLG  001157
SKDON   015152          SW9   = 001000          $BDADR  001122          $NWTST= 000001          $TPS    001150
SMGRP   014472          TBITVE= 000014          $BDDAT  001126          $OCNT   022302          $TRAP   022702
SOAD    001524          TIMDON  014050          $BELL   001206          $OMODE  022304          $TRAP2  022724
SORT    013342          TIMER   001264          $CHARC  020474          $OVER   017146          $TRP  = 000025
SP1     012702          TIMSEK  014776          $CKSWR  021064          $PASS   001100          $TRPAD  022736
SP10    012724          TKVEC = 000060          $CMTAG  001100          $POWER  023162          $TSTNM  001102
SP11    012726          TPVEC = 000064          $CM1  = 000011          $PWRAD  023150          $TTYIN  021712
SP12    012730          TRAPVE= 000034          $CM2  = 000022          $PWRDN  023010          $TYPDS  020034
SP2     012704          TRTVEC= 000014          $CM3  = 000011          $PWRMG  023144          $TYPE   020260
SP3     012706          TSTRWS  016716          $CNTLG  021727          $PWRUP  023062          $TYPEC  020430
SP4     012710          TST.RW= 104422          $CNTLU  021722          $QUES   001212          $TYPEX  020476
SP5     012712          TST1    004260          $CRLF   001213          $RDCHR  021366          $TYPOC  022104
SP6     012714          TST10   012174          $DBLK   020250          $RDLIN  021456          $TYPON  022120
SP7     012716          TST11   012736          $DB2D   022506          $RDOCT  021756          $TYPOS  022060
SP8     012720          TST12   015224          $DECVL  022666          $RDSZ = 000010          $XTSTR  017030
SP9     012722          TST2    004624          $DOAGN  015340          $REGAD  001160          $$GET4= 000000
STACK = 001100          TST3    005122          $DTBL   020240          $REGO   001162          $OFILL  022303
START   002462          TST4    005612          $ENDAD  015330          $REG1   001164          .     = 034446
STARTA  002530          TST5    006620          $ENDCT  015276          $REG10  001202
START1  003074          TST6    007356          $ENDMG  015347          $REG2   001166

. ABS.  034446      000

ERRORS DETECTED:  0
```

```
DSKM:DZRKLE,DSKZ:DZRKLE/SOL=DSKZ:SYSMAC.SML,DSKZ:DZRKLE.P11
RUN-TIME:  15 21  .5 SECONDS
RUN-TIME RATIO: 124/37=3.3
CORE USED:  35K  (69 PAGES)
```