

.REM 8

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

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN

WRITING BY DIGITAL.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1975,1977 BY DIGITAL EQUIPMENT CORPORATION

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	EQUIPMENT
2.2	PRELIMINARY PROGRAMS
2.3	EXECUTION TIME
3.0	STARTING ADDRESSES
4.0	PROGRAM CONTROL MODES AND OPERATOR ACTION
4.1	PAPER TAPE
4.2	RKDP DUMP MODE
4.3	RKDP CHAIN MODE
4.4	ACT11
5.0	DRIVE SELECTION
6.0	SWITCH OPTIONS
7.0	PROGRAM DESCRIPTION
7.1	PERMISSIBLE USER PROGRAM MODIFICATIONS
8.0	SEEK TIMER AND GRAPHS
9.0	FUNCTION SELECTION PROGRAM
10.0	ERROR INFORMATION
11.0	UNEXPECTED TIMEOUTS
12.0	COMMONLY USED SUBROUTINES
13.0	SAMPLE GRAPH AND TIMER OUTPUTS

1.0 ABSTRACT

THE RK11/RK05 DYNAMIC TEST AIMS AT  
1. DEMONSTRATING THE ELECTROMECHANICAL INTEGRITY OF THE DRIVE.  
2. CHECKING THE LINEAR POSITIONER CONTROL AND SPEED CONTROL  
3. VERIFYING THE INTEGRITY 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 SWITCHES 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	300	LOOP ON TEST INDICATED BY USER (SEE SEC. 6.8)
SW<06>=1	100	TYPE SEEK TIMER
SW<05>=1	40	TYPE THE GRAPHS
SW<04>=1	20	PRINT THE COMPLETE GRAPH

SW<03>=1	10	TERMINATE FUNCTION SELECTED BY USER
SW<02>=1	4	DROP THE DRIVE AFTER MAXIMUM ALLOWABLE NUMBER OF ERRORS OCCUR
SW<00>=1	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 LOOPING 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 TIMEOUT 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 17777. 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 APPEARS:

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.



## 7.0 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 CONVICIENCE 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 PRECEDED 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 BEGINNING.

- 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 'PBUFO' 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 BEGINNING OF TEST 6. THIS CALLING ORDER CAN BE CHANGED BY MAKING CHANGES IN THE FOUR POINTERS A. 'PATO' 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 'PTRN01' AND 'PTRN02' 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 ADDRESS (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 PERTAINING 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 TIME 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 EXPLANATION 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

(STRPAD) FOR THE STARTING ADDRESS OF THE DESIRED ROUTINE.

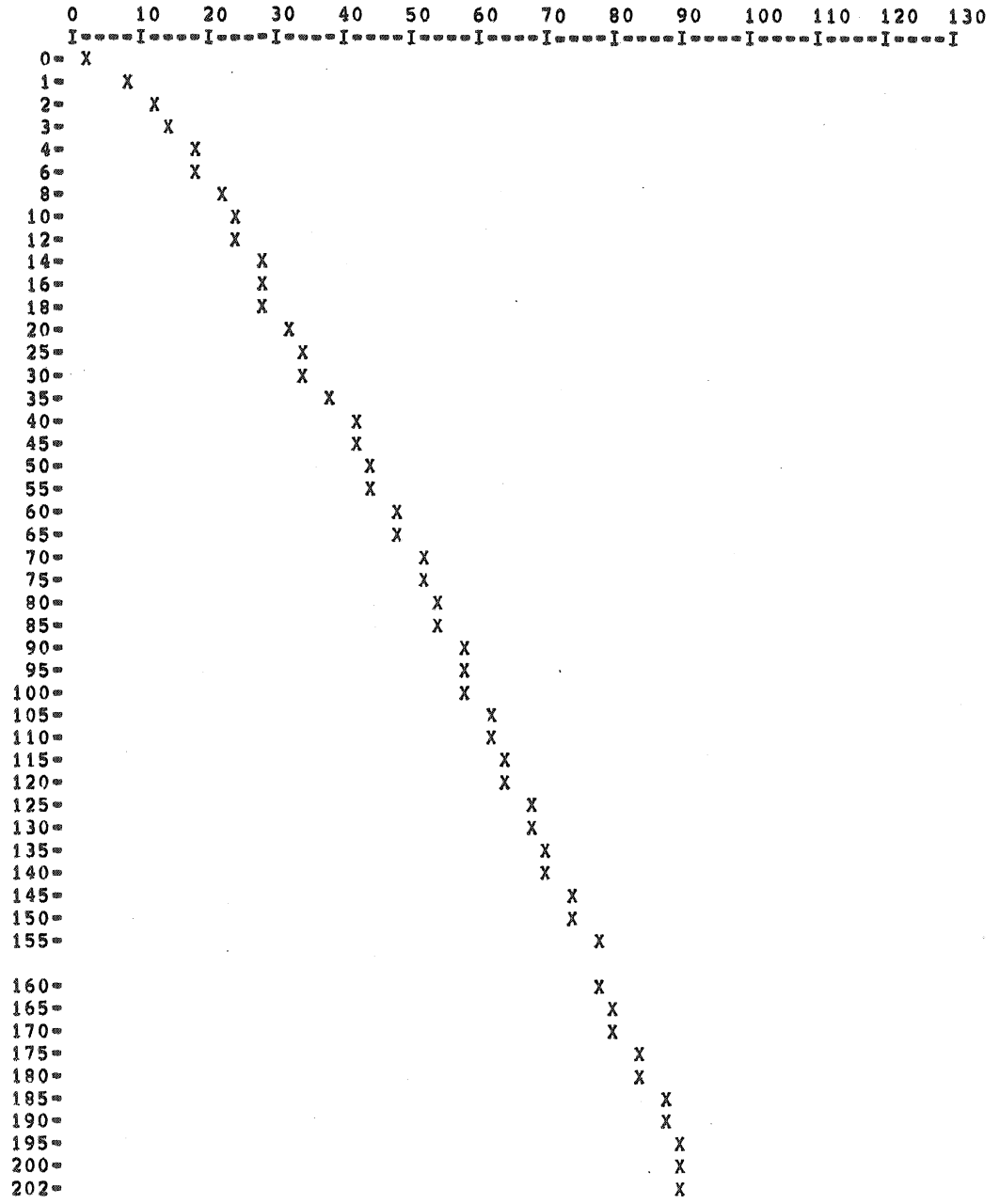
13.0 SAMPLE GRAPH AND SEEK TIMER OUTPUTS

'# 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                   \*\*SAMPLE OUTPUT\*\*  
 Y AXIS - CYLINDER SEEKED FROM 0



MD-11-DZRKL-E, RK11-RK05 DYNAMIC TEST MACY11 30(1046) 14-JUL-77 08:03 PAGE 17  
DZRKLE,P11 26-APR-77 12:27

8



```

825
826
827 .TITLE MD-11-DZRKL-E, RK11-RK05 DYNAMIC TEST
828 ;*COPYRIGHT (C) 1974,1977
829 ;*DIGITAL EQUIPMENT CORP.
830 ;*MAYNARD, MASS. 01754
831 ;*
832 ;*PROGRAM BY JIM KAPADIA
833 ;*
834 ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
835 ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
836 ;*
837 ;*JANUARY 1975
838 ;*
839 ;*REVISED MARCH 1976 BY TOM SAWYER
840 ;*REVISED BY CHUCK HESS, AUGUST, 1976
841
842 .SBTTL OPERATIONAL SWITCH SETTINGS
843 ;*
844 ;* SWITCH USE
845 ;* -----
846 ;* 15 HALT ON ERROR
847 ;* 14 LOOP ON TEST
848 ;* 13 INHIBIT ERROR TYPEOUTS
849 ;* 12 CYCLE ON ERROR TO PREVIOUS 'SCOPE'
850 ;* 10 BELL ON ERROR
851 ;* 9 LOOP ON ERROR
852 ;* 8 SELECT TEST TYPED IN BY USER
853 ;* 6 EXECUTE THE SEEK TIMER (TEST 11)
854 ;* 5 TYPE THE SEEK TIMER GRAPHS (TEST 11)
855 ;* 4 TYPE THE COMPLETE GRAPH (ALL SEEK TIMES)
856 ;* NOTE, OTHERWISE YOU GET SMALL GRAPH
857 ;* 3 TERMINATE FUNCTION SELECTED BY USER
858 ;* (FOR FUNCTION SELECTION PROGRAM SA=210)
859 ;* 2 DROP THE DRIVE AFTER MAXIMUM ALLOWABLE
860 ;* NUMBER OF ERRORS HAVE OCCURED
861 ;* 0 ASK FOR PATTERN TO BE WRITTEN (OR WRITE
862 ;* CHECKED), IN FUNCTION SELECTION PROGRAM
863 ;* 11 DUMP OUT ALL RK11 REGISTERS ON ERROR
864
865 ;* YOU ARE ADVISED TO READ THE DOCUMENT FOR THIS PROGRAM.
866 ;* FUNCTION SELECTION PROGRAM STARTS AT 210.
867
    
```

```

868 .SBTTL BASIC DEFINITIONS
869
870 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
871 STACK= 1100
872 ;*EQUIV EMT,ERROR ;*BASIC DEFINITION OF ERROR CALL
873 ;*EQUIV IOT,SCOPE ;*BASIC DEFINITION OF SCOPE CALL
874
875 ;*MISCELLANEOUS DEFINITIONS
876 HT= 11 ;*CODE FOR HORIZONTAL TAB
877 LF= 12 ;*CODE FOR LINE FEED
878 CR= 15 ;*CODE FOR CARRIAGE RETURN
879 CRLF= 200 ;*CODE FOR CARRIAGE RETURN=LINE FEED
880 PS= 177776 ;*PROCESSOR STATUS WORD
881 ;*EQUIV PS,PSW
882 STKLMT= 177774 ;*STACK LIMIT REGISTER
883 PIRO= 177772 ;*PROGRAM INTERRUPT REQUEST REGISTER
884 DSWR= 177570 ;*HARDWARE SWITCH REGISTER
885 DDISP= 177570 ;*HARDWARE DISPLAY REGISTER
886
887 ;*GENERAL PURPOSE REGISTER DEFINITIONS
888 RO= %0 ;*GENERAL REGISTER
889 R1= %1 ;*GENERAL REGISTER
890 R2= %2 ;*GENERAL REGISTER
891 R3= %3 ;*GENERAL REGISTER
892 R4= %4 ;*GENERAL REGISTER
893 R5= %5 ;*GENERAL REGISTER
894 R6= %6 ;*GENERAL REGISTER
895 R7= %7 ;*GENERAL REGISTER
896 SP= %6 ;*STACK POINTER
897 PC= %7 ;*PROGRAM COUNTER
898
899 ;*PRIORITY LEVEL DEFINITIONS
900 PR0= 0 ;*PRIORITY LEVEL 0
901 PR1= 40 ;*PRIORITY LEVEL 1
902 PR2= 100 ;*PRIORITY LEVEL 2
903 PR3= 140 ;*PRIORITY LEVEL 3
904 PR4= 200 ;*PRIORITY LEVEL 4
905 PR5= 240 ;*PRIORITY LEVEL 5
906 PR6= 300 ;*PRIORITY LEVEL 6
907 PR7= 340 ;*PRIORITY LEVEL 7
908
909 ;*"SWITCH REGISTER" SWITCH DEFINITIONS
910 SW15= 100000
911 SW14= 40000
912 SW13= 20000
913 SW12= 10000
914 SW11= 4000
915 SW10= 2000
916 SW09= 1000
917 SW08= 400
918 SW07= 200
919 SW06= 100
920 SW05= 40
921 SW04= 20
922 SW03= 10
923 SW02= 4
    
```

```

924      000002      SW01= 2
925      000001      SW00= 1
926      ,EQUIV SW09,SW9
927      ,EQUIV SW08,SW8
928      ,EQUIV SW07,SW7
929      ,EQUIV SW06,SW6
930      ,EQUIV SW05,SW5
931      ,EQUIV SW04,SW4
932      ,EQUIV SW03,SW3
933      ,EQUIV SW02,SW2
934      ,EQUIV SW01,SW1
935      ,EQUIV SW00,SW0
936
937      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
938      100000      BIT15= 100000
939      040000      BIT14= 40000
940      020000      BIT13= 20000
941      010000      BIT12= 10000
942      004000      BIT11= 4000
943      002000      BIT10= 2000
944      001000      BIT09= 1000
945      000400      BIT08= 400
946      000200      BIT07= 200
947      000100      BIT06= 100
948      000040      BIT05= 40
949      000020      BIT04= 20
950      000010      BIT03= 10
951      000004      BIT02= 4
952      000002      BIT01= 2
953      000001      BIT00= 1
954      ,EQUIV BIT09,BIT9
955      ,EQUIV BIT08,BIT8
956      ,EQUIV BIT07,BIT7
957      ,EQUIV BIT06,BIT6
958      ,EQUIV BIT05,BIT5
959      ,EQUIV BIT04,BIT4
960      ,EQUIV BIT03,BIT3
961      ,EQUIV BIT02,BIT2
962      ,EQUIV BIT01,BIT1
963      ,EQUIV BIT00,BIT0
964
965      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
966      000004      ERRVEC= 4          ;;TIME OUT AND OTHER ERRORS
967      000010      RESVEC= 10        ;;RESERVED AND ILLEGAL INSTRUCTIONS
968      000014      TBITVEC=14        ;;T" BIT
969      000014      TRTVEC= 14        ;;TRACE TRAP
970
971      000014      BPTVEC= 14         ;;BREAKPOINT TRAP (BPT)
972      000020      IOTVEC= 20         ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
973      000024      PWRVEC= 24         ;;POWER FAIL
974      000030      EMTVEC= 30         ;;EMULATOR TRAP (EMT) **ERROR**
975      000034      TRAPVEC=34         ;;"TRAP" TRAP
976      000060      TKVEC= 60          ;;TTY KEYBOARD VECTOR
977      000064      TPVEC= 64          ;;TTY PRINTER VECTOR
978      000240      PIRQVEC=240        ;;PROGRAM INTERRUPT REQUEST VECTOR
979
980      ,SBTTL TRAP CATCHER
  
```

```

980
981      000000      ,=0
982      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ",+2,HALT"
983      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
984      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
985      ,=174
986      000174      000000      DISPREG; ,WORD 0          ;;SOFTWARE DISPLAY REGISTER
987      000176      000000      SWREG; ,WORD 0          ;;SOFTWARE SWITCH REGISTER
988      ,SBTTL STARTING ADDRESS(ES)
989      000200      000137      002462      JMP @*START ;;JUMP TO STARTING ADDRESS OF PROGRAM
990
991      ,=210
992      000210      105237      001216      INCB FFUNC          ;;SET FLAG INDICATING SELECTION OF
993      000214      000137      002462      JMP @*START          ;;FUNCTION PROGRAM.
994      ,SBTTL ACT11 HOOKS
995
996      ;*****
997      ;HOOKS REQUIRED BY ACT11
998      ,=SVPC=,          ;;SAVE PC
999      ,=46
1000     000046      015330      SENDAD          ;;1)SET LOC,46 TO ADDRESS OF SENDAD IN ,SEOP
1001     ,=52
1002     000052      000000      ,WORD 0          ;;2)SET LOC,52 TO ZERO
1003     ,=SVPC          ;; RESTORE PC
1004
  
```

```
1005          ,SBTTL COMMON TAGS
1006
1007          ;*****
1008          ;THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1009          ;USED IN THE PROGRAM.
1010
1011          001100          ,=1100
1012 001100          $CMTAG:          ;START OF COMMON TAGS
1013 001100 000000          $PASS:   ,WORD 0          ;CONTAINS PASS COUNT
1014 001102 000          $STNM:    ,BYTE 0          ;CONTAINS THE TEST NUMBER
1015 001103 000          $ERFLG:   ,BYTE 0          ;CONTAINS ERROR FLAG
1016 001104 000000          $ICNT:    ,WORD 0          ;CONTAINS SUBTEST ITERATION COUNT
1017 001106 000000          $LPADR:   ,WORD 0          ;CONTAINS SCOPE LOOP ADDRESS
1018 001110 000000          $LPERR:   ,WORD 0          ;CONTAINS SCOPE RETURN FOR ERRORS
1019 001112 000000          $ERTTL:   ,WORD 0          ;CONTAINS TOTAL ERRORS DETECTED
1020 001114 000          $ITEMB:   ,BYTE 0          ;CONTAINS ITEM CONTROL BYTE
1021 001115 001          $ERMAX:   ,BYTE 1          ;CONTAINS MAX. ERRORS PER TEST
1022 001116 000000          $ERRPC:   ,WORD 0          ;CONTAINS PC OF LAST ERROR INSTRUCTION
1023 001120 000000          $GDADR:   ,WORD 0          ;CONTAINS ADDRESS OF 'GOOD' DATA
1024 001122 000000          $BDADR:   ,WORD 0          ;CONTAINS ADDRESS OF 'BAD' DATA
1025 001124 000000          $GDDAT:   ,WORD 0          ;CONTAINS 'GOOD' DATA
1026 001126 000000          $BDDAT:   ,WORD 0          ;CONTAINS 'BAD' DATA
1027 001130 000000          ,WORD 0          ;RESERVED--NOT TO BE USED
1028 001132 000000          ,WORD 0
1029 001134 000          $AUTOB:  ,BYTE 0          ;AUTOMATIC MODE INDICATOR
1030 001135 000          $INTAG:   ,BYTE 0          ;INTERRUPT MODE INDICATOR
1031 001136 000000          ,WORD 0
1032 001140 177570          SWR:      ,WORD DSWR          ;ADDRESS OF SWITCH REGISTER
1033 001142 177570          DISPLAY: ,WORD DDISP          ;ADDRESS OF DISPLAY REGISTER
1034 001144 177560          $TKS:    ,WORD 177560          ;TTY KBD STATUS
1035 001146 177562          $TKB:    ,WORD 177562          ;TTY KBD BUFFER
1036 001150 177564          $TPS:    ,WORD 177564          ;TTY PRINTER STATUS REG. ADDRESS
1037 001152 177566          $TPB:    ,WORD 177566          ;TTY PRINTER BUFFER REG. ADDRESS
1038 001154 000          $NULL:   ,BYTE 0          ;CONTAINS NULL CHARACTER FOR FILLS
1039 001155 002          $FILLS:  ,BYTE 2          ;CONTAINS # OF FILLER CHARACTERS REQUIRED
1040 001156 012          $FILLC:  ,BYTE 12          ;INSERT FILL CHARS. AFTER A "LINE FEED"
1041 001157 000          $TPFLG: ,BYTE 0          ;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1042 001160 000000          $REGAD: ,WORD 0          ;CONTAINS THE ADDRESS FROM
1043          ,WHICH ($REGO) WAS OBTAINED
1044 001162 000000          $REG0:   ,WORD 0          ;CONTAINS (($REGAD)+0)
1045 001164 000000          $REG1:   ,WORD 0          ;CONTAINS (($REGAD)+2)
1046 001166 000000          $REG2:   ,WORD 0          ;CONTAINS (($REGAD)+4)
1047 001170 000000          $REG3:   ,WORD 0          ;CONTAINS (($REGAD)+6)
1048 001172 000000          $REG4:   ,WORD 0          ;CONTAINS (($REGAD)+10)
1049 001174 000000          $REG5:   ,WORD 0          ;CONTAINS (($REGAD)+12)
1050 001176 000000          $REG6:   ,WORD 0          ;CONTAINS (($REGAD)+14)
1051 001200 000000          $REG7:   ,WORD 0          ;CONTAINS (($REGAD)+16)
1052 001202 000000          $REG10: ,WORD 0          ;CONTAINS (($REGAD)+20)
1053 001204 000000          $ESCAPE:0          ;ESCAPE ON ERROR ADDRESS
1054 001206 177607 000377          $BELL:   ,ASCIZ <207><377><377>          ;CODE FOR BELL
1055 001212 077          $QUES:   ,ASCII /?/          ;QUESTION MARK
1056 001213 015          $CRLF:   ,ASCII <15>          ;CARRIAGE RETURN
1057 001214 000012          $LF:     ,ASCIZ <12>          ;LINE FEED
1058          ;*****
1059
1060
```

```
1061          ;IN CASE YOU WANT THE OUTPUT TO COME OUT ON LINE PRINTER, (IF YOU HAVE
1062          ;ONE), MAKE THE FOLLOWING CHANGES ABOVE:
1063
1064          ;CHANGE CONTENTS OF '$TPS' TO 177514 (LPT VECTOR)
1065          ;CHANGE CONTENTS OF '$TPB' TO 177516 ( " ")
1066
1067          ;TAGS AND GENERAL DATA AREA
1068
1069 001216 000000          FFUNC:   ,WORD 0          ;FLAG SET, TO INDICATE ENTRY INTO FUNCTION PROGRAM
1070 001220 000000          XXDPM:   ,WORD 0          ;IF PROGRAM LOADED BY XXDP, THE
1071          ,LOWER BYTE HAS THE DRIVE NUMBER
1072          ;AND THE UPPER BYTE CONTAINS THE RK05 'XXDP' CODE
1073 001222 000          LUPSW:   ,BYTE 0          ;FLAG, SET TO INDICATE THAT A
1074          ;PARTICULAR TEST WAS SELECTED BY USER (SW 8)
1075
1076
1077
1078 001223 000          DRVDON: ,BYTE 0          ;CONTAINS NUMBER OF DRIVES THAT HAVE
1079          ;BEEN ALREADY CHECKED
1080 001224 000          DRVS:    ,BYTE 0          ;CONTAINS TOTAL # OF DRIVES PRESENT
1081
1082 001226          ,EVEN
1083
1084
1085 001226 000000          DRVPT: 0          ;CONTAINS POINTER TO INDICATOR STARTING
1086          ;WHICH CHECKING SHOULD BE DONE FOR NEXT
1087          ;AVAILABLE DRIVE
1088 001230 000000          DRIVAD:0          ;CONTAINS THE ADDRESS OF THE DRIVE
1089          ;BEING TESTED
1090
1091 001232 000000          DRIV0: 000000          ;THESE ARE FLAGS TO INDICATE
1092 001234 020000          DRIV1: 020000          ;THAT A PARTICULAR DRIVE IS
1093 001236 040000          DRIV2: 040000          ;PRESENT. BIT 0 IS SET TO
1094 001240 060000          DRIV3: 060000          ;INDICATE THAT. BITS 13, 14, 15
1095 001242 100000          DRIV4: 100000          ;CONTAIN THE LOGICAL DRIVE
1096 001244 120000          DRIV5: 120000          ;ADDRESS
1097 001246 140000          DRIV6: 140000
1098 001250 160000          DRIV7: 160000
1099
1100
1101
1102 001252 000000          RETRY1: 0          ;GENERAL REGISTERS
1103 001254 000000          RETRY2: 0
1104 001256 000000          RETRY3: 0
1105
1106
1107 001260 000000          INADR: 0          ;CONTAINS INNER ADDRESS
1108 001262 000000          OUTADR:0          ;CONTAINS OUTER ADDRESS
1109 001264 000000          TIMER: 0
1110
1111
1112
1113 001266 000015          BUFR:   ,BLKW 13,          ;GENERAL BUFFERS
1114 001320 000015          BUFR1: ,BLKW 13,
1115 001352 000015          BUFR2: ,BLKW 13,
1116
```

```

1117
1118 ;IN CASE, YOU WANT TO USE BUFFERS STARTING AT SOME OTHER MEMORY
1119 ;ADDRESS YOU CAN DO SO BY CHANGING THE FOLLOWING POINTERS.
1120 ;BOTH THE BUFFERS SHOULD BE 768 (DECIMAL) WORDS LONG.
1121
1122 001404 026446 PBUF0: IOBUFO ;POINTER TO THE STARTING ADDRESS OF THE
1123 ;BUFFER USED TO READ INTO FROM DIBK.
1124 001406 031446 PBUF1: IOBUF1 ;POINTER TO STARTING ADDRESS OF BUFFER
1125 ;IN WHICH PATTERNS ARE GENERATED. (WRITING
1126 ;IS DONE FROM THIS BUFFER)
1127 001410 000000 BUFLG0: ,WORD 0 ;FLAG FOR 'IOBUFO'
1128 001412 000000 BUFLG1: ,WORD 0 ;FLAG FOR 'IOBUF1'
1129
1130
1131 001414 010106 PAT0: PTGEN0 ;ADRES OF 'PATRN GENERATOR 0'
1132 ;ROUTINE
1133 001416 010170 PAT1: PTGEN1 ;ADRES OF 'PATRN GENERATOR 1'
1134
1135 001420 010272 PAT2: PTGEN2 ;ADRES OF 'PATRN GENRATOR 2'
1136
1137 001422 010334 PAT3: PTGEN3 ;ADRES OF 'PATRN GENRATOR 3'
1138
1139 001424 000000 PRSPAT: ,WORD 0 ;CONTAINS THE POINTER TO THE
1140 ;ADRES OF 1 OF THE 3 'PATRN
1141 ;GENRATOR' ROUTINES
1142 001426 000000 NXTPAT: ,WORD 0 ;SAME AS ABOVE
1143
1144 001430 000000 PGSUBR: ,WORD 0
1145
1146 001432 000000 DSKADR: ,WORD 0 ;CONTAINS DISK ADRES (DA)
1147
1148 001434 000000 BUSADR: ,WORD 0 ;CONTAINS BUS ADRES (BA)
1149
1150 001436 000000 WRDCNT: ,WORD 0 ;CONTAINS WORD COUNT
1151
1152 001440 000000 WDSKAD: ,WORD 0 ;CONTAINS DISK ADRES
1153
1154 001442 000000 WBUSAD: ,WORD 0 ;CONTAINS BUS ADRES
1155
1156 001444 000000 WWRDCN: ,WORD 0 ;CONTAINS WORD COUNT
1157
1158 001446 000000 BUFNO: ,WORD 0 ;CONTAINS STARTING ADRES
1159
1160 001450 000000 ADRES: ,WORD 0 ;OF A BUFFER
1161
1162 ;RK11 REGISTERS
1163
1164 ;IF FOR ANY REASON THE REGISTER ADDRESSES ARE DIFFERENT FROM
1165 ;THESE (BELOW), THE CONTENTS OF THE APPROPRIATE POINTERS SHOULD
1166 ;BE MODIFIED SO THAT THE CORRECT REGISTER ADDRESS IS USED.
1167
1168 001452 177400 RKDS: 177400
1169 001454 177402 RKER: 177402
1170 001456 177404 RKCS: 177404
1171 001460 177406 RKWC: 177406
1172 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 ;CYLINDER 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  
1234 ;FOLLOWING POINTERS ARE USED TO TRANSFER CONTROL TO THE  
1235 ;TEST SELECTED BY USING SW 8. IF ANY MORE TESTS ARE  
1236 001560 004262 ;ADDED TO THIS PROGRAM ADDITIONAL POINTERS SHOULD BE INSERTED.  
1237 001562 004626 PT1: TST1+2  
1238 001564 005124 PT2: TST2+2  
1239 001566 005614 PT3: TST3+2  
1240 001570 006622 PT4: TST4+2  
1241 001572 007360 PT5: TST5+2  
1242 001574 010440 PT6: TST6+2  
1243 001576 012176 PT7: TST7+2  
1244 001600 012740 PT10: TST10+2  
1245 PT11: TST11+2  
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 #1/  
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 BLNK59: .ASCII / /  
1301 002102 040 BLNK58: .ASCII / /  
1302 002103 040 BLNK57: .ASCII / /  
1303 002104 040 BLNK56: .ASCII / /  
1304 002105 040 BLNK55: .ASCII / /  
1305 002106 040 BLNK54: .ASCII / /  
1306 002107 040 BLNK53: .ASCII / /  
1307 002110 040 BLNK52: .ASCII / /  
1308 002111 040 000 BLNK51: .ASCIZ / /  
1309  
1310 002114 .EVEN  
1311 002114 000000 FDRIVE: 0  
1312 002116 000000 FDRVE1: 0  
1313 002120 000000 DRHOLD: 0

```

1314 ,SBTTL ERROR POINTER TABLE
1315
1316 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR,
1317 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
1318 ;*LOCATION $ITEMB, THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT,
1319 ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRRPC),
1320 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
1321
1322 ;* EM ;POINTS TO THE ERROR MESSAGE
1323 ;* DH ;POINTS TO THE DATA HEADER
1324 ;* DT ;POINTS TO THE DATA
1325 ;* DF ;POINTS TO THE DATA FORMAT
1326
1327
1328 002122 $ERRTB:
1329
1330 ;ERROR ITEMS TABLE
1331 ;
1332 ;
1333 ;
1334 ;ITEM 1
1335
1336 EM1 ;CNTRL RDY DIDN'T SET AFTER SEEK
1337 002122 024334 ;PC RKCS RKER RKDS RKDA
1338 002124 025500 ;ERRRPC $REG0 $REG1 $REG2 $REG3
1339 002126 026334
1340 002130 000000
1341
1342 ;ITEM 2
1343
1344 EM2 ;SIN ON SEEK
1345 002132 024373 ;PC RKCS RKER RKDS RKDA
1346 002134 025500 ;ERRRPC $REG0 $REG1 $REG2 $REG3
1347 002136 026334
1348 002140 000000
1349
1350 ;ITEM 3
1351 EM3 ;DRE ON SEEK
1352 002142 024407 ;PC RKCS RKER RKDS RKDA
1353 002144 025500 ;ERRRPC $REG0 $REG1 $REG2 $REG3
1354 002146 026334
1355 002150 000000
1356
1357 ;ITEM 4
1358 EM4 ;'ERR' ON SEEK
1359 002152 024423 ;PC RKCS RKER RKDS RKDA
1360 002154 025500
1361 DT1 ;ERRRPC $REG0 $REG1 $REG2 $REG3
1362 0
1363
1364 ;ITEM 5
1365 EM5 ;'DRU' ON SEEK; PUT DRIVE ON 'LOAD' BACK TO 'RUN'
1366 002162 024441 ;PC RKCS RKER RKDS RKDA
1367 002164 025500 ;ERRRPC $REG0 $REG1 $REG2 $REG3
1368 002166 026334
1369 002170 000000
1370 0

```

```

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

```



```

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),RO ;SAVE PC WHERE TIME OUT OCCURED
1525 002424 005740 TST =(RO)
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$: .ASCIZ <15><12>/TIMOUT:PC=/
64$:
1530 002454 MOV RO,=(SP) ;SET UP FOR TYPING OUT PC
1531 002454 010046 TYPOC ;GO TYPE OUT OCTAL PC
1532 002456 104402 HALT
1533 002460 000000
1534
1535 002462 000005 START: RESET ;CLEAR THE BUS
;GIVE DRIVES TIME TO RELOAD HEADS IN CASE OF AN APT START
1536
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,RO ;WAIT FOR..
1541 002504 032777 000200 176740 20$: BIT @#200,@RKDS ;DRIVE READY..
1542 002512 001006 BNE STARTA ;IN CASE..
1543 002514 005001 CLR R1 ;OF APT..
1544 002516 005301 DEC R1 ;START, BUT..
1545 002520 001376 BNE @-2 ;DON'T WAIT..
1546 002522 005300 DEC RO ;FOREVER.
1547 002524 001367 BNE 20$
1548 002526 000000 HALT ;RKDS BIT 7 (DIRVE READY) NEVER SET
1549 002530
1550
1551 ;SBTTL INITIALIZE THE COMMON TAGS
;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
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 MOV $1,$ERMAX ;ALLOW ONE ERROR PER TEST
1568 002642 012737 002642 001106 MOV $,$SLPADR ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1569 002650 012737 002650 001110 MOV $,$SLPERR ;SETUP THE ERROR LOOP ADDRESS
1570
;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
;EQUAL TO A "1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1571
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 SWICH 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
;BRANCH IF NO TIMEOUT
1579 002714 000403 BR 65$
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 ;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
;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
1590
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$: MOV $1,$AUTOB ;SET AUTO-MODE INDICATOR
1602 003022 70$:
1603 003022 000424 BR 67$ ;GET OVER THE ASCIZ
1604
;68$: .ASCIZ <CRLF>/RK11 DYNAMIC TEST/<15><12>/MAINDEC-11-DZRKL-E/<CRLF>
67$:
1605 003074
1606 003074 105737 001216 START: TSTB FFUNC ;FUNCTION PROGRAM SELECTED?
1607 003100 001404 BEQ 78 ;NO
1608 003102 105037 001216 CLRFB FFUNC ;YES, CLEAR THE FLAG
1609 003106 000137 023172 JMP @#FUNBEG ;GO TO "FUNCTION SELECTION PROGRAM"
1610 003112 012700 001230 78: MOV $XXDPMD,RO ;CLEAR FLAGS FROM
1611 003116 105020 58: CLRFB (RO)+ ;"XXDPMD" TO "DRIVAD"
1612 003120 020027 001232 CMP RO,$DRIVAD+2
1613 003124 001374 BNE 58
1614 003126 012701 177770 MOV @-10,R1
1615 003132 042720 000003 68: BIC $3,(RO)+ ;CLEAR BIT 0'S IN "DRIVE
1616 003136 005201 INC R1 ;PRESENT" FLAGS.
1617 003140 001374 BNE 68
1618
;THE FOLLOWING CODE FINDS OUT THE PROGRAM CONTROL MODE:
;PAPER TAPE (MANUAL), ACT11, RKDP CHAIN OR DUMP
1619
1620
1621
1622 003142 122737 000002 000041 CMPB $2,$41 ;LOADED FROM AN RK05 ?
1623 003150 001160 BNE $T2 ;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 CLRFB $XXDPMD ;DRIVE ZERO LOADED THE PROGRAM
1629 003174 005737 000042 28: TST $42 ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
1630 003200 001424 BEQ $3 ;BR IF NEITHER

```



```

1631 003202 104401 003210      TYPE      ,65$      ;;TYPE ASCIZ STRING
1632 003206 000413      BR        64$      ;;GET OVER THE ASCIZ
1633      ;165$:      ,ASCIZ <15><12>/NOT TESTING DRIVE /
1634      64$;
1635 003236 005046      CLR      -(SP)      ;CLEAR WORD ON STACK
1636 003240 113716 001220      MOV     XDXPMD,(SP) ;GET DRIVE ADDRESS
1637 003244 104403      TYPOS   ;TYPE THE ADDRESS
1638 003246 001      ,BYTE 1 ;ONLY 1 CHARACTER
1639 003247 000      ,BYTE 0 ;SUPPRESS LEADING ZEROS
1640 003250 000520      BR      ST2        ;GET NUMBER OF DRIVES
1641 003252 005227 177777      INC     #-1        ;FIRST TIME THROUGH HERE ?
1642 003256 001115      BNE     ST2        ;BR IF NOT FIRST TIME
1643 003260 104401 003266      TYPE   ,67$      ;;TYPE ASCIZ STRING
1644 003264 000411      BR      66$      ;;GET OVER THE ASCIZ
1645      ;167$:      ,ASCIZ <15><12>/TO TEST DRIVE /
1646      66$;
1647 003310      CLR      -(SP)      ;CLEAR WORD ON THE STACK
1648 003312 113716 001220      MOV     XDXPMD,(SP) ;GET DRIVE ADDRESS
1649 003316 104403      TYPOS   ;TYPE THE DRIVE ADDRESS
1650 003320 001      ,BYTE 1 ;ONLY 1 CHARACTER
1651 003321 000      ,BYTE 0 ;SUPPRESS LEADING ZEROS
1652 003322 104401 003330      TYPE   ,69$      ;;TYPE ASCIZ STRING
1653 003326 000431      BR      68$      ;;GET OVER THE ASCIZ
1654      ;169$:      ,ASCIZ / HALT PROGRAM, REMOVE RKDP PACK AND REPLACE IT/<15><12>
1655      68$;
1656 003412 104401 003420      TYPE   ,71$      ;;TYPE ASCIZ STRING
1657 003416 000435      BR      70$      ;;GET OVER THE ASCIZ
1658      ;171$:      ,ASCIZ /WITH A WORK PACK, CLEAR LOCATION 40, AND RESTART PROGRAM/
1659      70$;
1660
1661
1662 003512 012737 002422 000004 ST2:  MOV     #BADTMO,ERRVEC ;SET TIMEOUT VECTOR FOR
1663      ;UNEXPECTED TIME OUT
1664
;THIS CODE FINDS WHICH DRIVES ARE PRESENT & PRINTS OUT THE DRIVE
;DRIVE NUMBERS THAT WERE FOUND ON LINE,
1665
1666
1667
1668 003520 104401 003526      TYPE   ,65$      ;;TYPE ASCIZ STRING
1669 003524 000411      BR      64$      ;;GET OVER THE ASCIZ
1670      ;165$:      ,ASCIZ <15><12>/DRIVES PRESENT/
1671      64$;
1672
1673 003550      CLR     DRVS      ;INITIALIZE NO. OF DRVS PRESENT
1674 003554 005001      CLR     R1
1675 003556 012702 001232      MOV     #DRIVO,R2 ;INITIALIZE COUNT TO 0
1676 003562 005003      CLR     R3        ;LOADED FROM AN RK05 ?
1677 003564 005737 001220      TST    XDXPMD
1678
1679      BEQ   66        ;BR IF NOT
1680      CMPB R3,XDXPMD ;CHECKING THE LOAD DRIVE ?
1681      BEQ   28        ;BR IF YES
1682
1683
1684 003600 010177 175660      68:  MOV     R1,#RKDA ;ADRES A DRIVE
1685 003604 105777 175642      TSTB   @RKDS      ;IS IT PRESENT?
1686 003610 100004      BPL     28        ;NO, BRANCH
1687 003612 105237 001224      INCB   DRVS      ;INCREMENT TOTAL # OF DRVS
1688 003616 052712 000001      BIS   #1,(R2)    ;SET FLAG INDICATING THIS DRV PRSNT
1689 003622 005722      TST   (R2)+
1690 003624 095203      INC    R3        ;INCREMENT COUNT

```

```

1687 003626 062701 020000      ADD     #20000,R1 ;ADRES THE NXT DRV
1688      ;CHKD ALL 8 DRIVES?
1689 003632 001354      BNE     18        ;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   DRVS      ;WERE ANY DRIVES FOUND?
1693 003644 001010      BNE     38        ;YES, BRANCH
1694 003646 104401 003654      TYPE   ,67$      ;;TYPE ASCIZ STRING
1695 003652 000403      BR      66$      ;;GET OVER THE ASCIZ
1696
;167$:      ,ASCIZ / NONE/
1697      66$;
1698 003662 000137 015246      JMP     $EOP      ;IF NONE WERE FOUND, GO
1699      ;TO THE END OF PROGRAM
1700
38:  CLR     R2        ;DRIVE NUMBER
1701 003670 012700 001232      MOV     #DRIVO,R0 ;TABLE OF AVALI DRIVES
1702 003674 105710      TSTB   (R0)      ;DRIVE HERE?
1703 003676 001414      BEQ     48        ;NO
1704 003700 104401      TYPE   $CRDIF
1705 003702 001213      MOV     R2,-(SP) ;PUSH NO ON THE STACK
1706 003704 010246      TYPOS   ;TO TYPE OCTAL NO.
1707 003706 104403      ,BYTE 1 ;TYPE 1 DIGIT, SUPPRESS LDG 0'S
1708 003710 001      ,BYTE 0
1709 003711 000      BIT     #2,(R0)   ;IS IT RK05F?
1710 003712 032710 000002      BEQ     48        ;NO
1711 003716 001404      TYPE   ,69$      ;;TYPE ASCIZ STRING
1712 003720 104401 003726      BR      68$      ;;GET OVER THE ASCIZ
1713 003724 000401      ;169$:      ,ASCIZ /F/
1714      68$;
1715
48:  INC     R2        ;POINT TO NEXT DRIVE #
1716 003730      TST   (R0)+      ;NEXT DRIVE IN TABLE
1717 003732 005720      CMP     R0,#DRIV7+1 ;ALL DONE?
1718 003734 020027 001251      BLT    58        ;NO, CHECK REST
1719 003740 002755
1720
;FIND OUT THE FIRST (FROM 0-7) DRIVE THAT IS PRESENT. PUT THE ADDRESS
;OF THAT DRIVE IN "DRIVAD". INDICATE THAT DRIVE # WILL
;BE TESTED.
1721
1722
1723
1724
1725 003742 012737 001232 001226 ST3:  MOV     #DRIVO,DRVPT ;DRIVE,DRVPT
1726 003750 105037 001223      CLRB   DRVAD     ;RESET TEST NUMBER TO 1
1727 003754 005037 001230      CLR     DRIVAD   ;CLEAR ERROR COUNT FOR THIS DRIVE
1728 003760 105037 001102      CLR     $TINM
1729 003764 005037 001112      CLR     $ERTL
1730 003770 013701 001226      MOV     DRVPT,R1 ;IS THIS DRIVE PRESENT?
1731 003774 032721 000001      BIT   #1,(R1)+  ;YES, BRANCH
1732 004000 001005      BNE     28        ;CHECKED THE WHOLE LIST?
1733 004002 020127 001252      CMP     R1,#DRIV7+2 ;YES, EXIT
1734 004006 001372      BNE     18        ;NO, GO AHEAD
1735 004010 000137 015246      JMP     $EOP     ;GET DRIVE NO. TO BE TESTED
1736 004014 010137 001226      MOV     R1,DRVPT ;SHOWS F IF -1
1737 004020 014104      MOV     -(R1),R4 ;RK-05F?
1738 004022 005037 002114      CLR     FDRVE1   ;NO
1739 004026 005037 002114      CLR     FDRIVE   ;SHOWS F
1740 004032 032704 000002      BIT   #2,R4
1741 004036 001410      BEQ     78        ;NO
1742 004040 005237 002114      INC     FDRVE1   ;SHOWS F

```

```
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 64# :
1765 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 CLR B 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
1773 004146 104401 004154 5# TYPE ,67# ;;TYPE ASCIZ STRING
1774 004152 000410 BR 66# ;;GET OVER THE ASCIZ
1775 ;;67# : ,ASCIZ <12>/OCTAL TEST#?/
1776 66# :
1777 004174 RDOCT
1778 004176 104412 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 MOV B,R0,#STSTNM
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 ERRORS
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 ERROR 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 R4 ;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#
1854 004410 000403 BR 6#
```

```

1855 004412 105777 175040 58: TSTB @RKCS ;DID CNTRL RDY SET?
1856 004416 100364 BPL 48 ;IF NOT WAIT FOR IT
1857
1858 004420 005000 68: CLR R0 ;INITIALIZE COUNT
1859 004422 032777 000100 175022 BIT #100,@RKDS ;R/W/S RDY SET?
1860 004430 001010 BNE 78 ;YES
1861 004432 005200 INC R0 ;WAIT FOR R/W/S RDY
1862 004434 001372 BNE 68+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
;WAS DONE TO CYLINDER INDICATED IN RKDA
1865
1866 004444 005203 INC R3 ;SET ERROR FLAG
1867 004446 005204 INC R4 ;IF MAXM EROR COUNT, ABORT
1868 004450 001461 BEQ 188
1869 004452 032777 001000 174772 78: BIT #1000,@RKDS ;SIN ERROR?
1870 004460 001406 BEQ 88 ;NO, BRANCH
1871 004462 004737 016162 JSF PC,GT4RG ;GO, GET RKCS, ER, DS, DA
1872 004466 104002 ERROR 2 ;SIN ERROR, ON DOING SEEK TO
;CYL AS SHOWN IN RKDA
1873
1874 004470 005203 INC R3 ;SET ERROR FLAG
1875 004472 005204 INC R4 ;IF MAXM EROR COUNT REACHED,
1876 004474 001447 BEQ 188 ;ABORT THE TEST
1877 004476 005777 174752 88: TST @RKER ;DRE ERROR?
1878 004502 100006 BPL 108 ;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
;AS SHOWN IN RKDA
1882
1883 004512 005203 INC R3 ;SET ERROR FLAG
1884 004514 005205 INC R5 ;IF MAXM EROR COUNT REACHED,
1885 004516 001767 BEQ 88 ;ABORT THE TEST
1886
1887 004520 005777 174732 108: TST @RKCS ;'ERR' BIT IN RKCS SET?
1888 004524 100006 BPL 128 ;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
;TO CYL AS SHOWN IN RKDA, NOTE
;WHICH BIT IN RKER SET?
1891
1892
1893 004534 005203 INC R3 ;SET ERROR FLAG
1894 004536 005205 INC R5 ;IF MAXM EROR COUNT REACHED,
1895 004540 001425 BEQ 188 ;ABORT THE TEST
1896
1897 004542 032777 002000 174702 128: BIT #2000,@RKDS ;DRU SET?
1898 004550 001406 BEQ 158 ;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 TRANSDVCR 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 188 ;IF MORE THAN 5
    
```

```

1911 ;GO TO THE END OF THE PROGRAM
1912
1913 004566 005702 158: TST R2 ;WAS SEEKING TO 0 OR 312?
1914 004570 001402 BEQ 168 ;TO 0, BRANCH
1915 ;TO 312,
1916 004572 005002 CLR R2 ;SEEK NXT TIME TO 0
1917 004574 000647 BR 18 ;GO BAK & SK TO 0
1918
1919 004576 012702 014500 168: MOV #14500,R2 ;SEEK NXT TIME TO 312
1920
1921 004602 005201 INC R1 ;DONE SEEKS 200 TIMES?
1922 004604 022701 000200 CMP #200,R1
1923
1924 004610 001241 BNE 18 ;IF NOT, GO BAK
1925 004612 000404 BR TST2 ;EXIT
1926
1927
1928 004614 104401 001640 188: TYPE ,MSG4
1929 004620 000137 015224 JMP TST12
1930
1931 ;*****
1932 ;*TEST 2 FORMAT THE DISK
1933 ;*THIS PROGRAM ASSUMES AN UNFORMATTED DISK AND ITS
1934 ;*FORMATTING IS DONE IN THIS TEST, A SECTOR IS FORMATTED
1935 ;*AT A TIME, THE FIRST OWRD OF EVERY SECTOR IS WRITTEN
1936 ;*TO BE A PSEUDO-HEADER CONTAINING THE DRIVE #, CYLINDER
1937 ;*, SURFACE AND SECTOR #. THE FOLLOWING IS CHECKED
1938 ;*1. 'SIN' IF 'SIN' OCCURS, A DRIVE RESET IS DONE
1939 ;*AND THE SAME SECTOR IS FORMATTED AGAIN, THREE
1940 ;*RETRIES ARE DONE BEFORE AN ERROR MESSAGE IS PRINTED.
1941 ;*2. 'ERR' ON FINDING THAT THE 'ERR' BIT SET, RKER
1942 ;*SCANNED TO FIND OUT WHAT CAUSED IT AND THE
1943 ;*ERROR IS REPORTED.
1944 ;*****
1945 004624 000004 TST2: SCOPE
1946 004626 013737 001230 002120 MOV DRIVAD,DRHOLD ;SAVE DRIVE NUMBER
1947 004634 005737 002114 TST FDRIVE ;SEE IF EVEN RK-05F DRIVE
1948 004640 001003 BNE 18 ;YES
1949 004642 005737 002116 TST FDRVE1 ;ODD RK-05F?
1950 004646 001125 BNE TST3 ;DO NOT FORMAT IF ODD RK-05F
1951 004650
1952 004650 012702 177152 118: MOV #-626,R2 ;203 CYLINDERS, (406 TRAKS)
1953 004654 012703 177764 MOV #-14,R3 ;12 SECTORS
1954 004660 012701 177773 MOV #-5,R1 ;ALLOW ONLY 5 'SIN' ERRORS
1955 004664 012705 177773 MOV #-5,R5 ;ALLOW ONLY 5 'ERR'S
1956 004670 013704 001230 MOV DRIVAD,R4 ;STORE ADRES OF DRIVE.
1957 004674 104415 48: CON,RESET
1958 004676 104416 DRV,RESET ;GO TO 'DR-RST' & DO DRIVE RESET
1959 004700 005000 18: CLR R0 ;KEEP COUNT OF 'SIN' ERORS
1960 ;ALLOW 3 RETRIES ON SIN
1961 004702 005777 174550 TST @RKCS ;ERR?
1962 004706 100001 BPL 38 ;NO
1963
1964 004710 104415 CON,RESET ;GO TO 'CN-RST' & DO CONTROL RESET
1965
1966 004712 005046 38: CLR -(SP)
    
```



```

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 005270 005237 001510  INC      ERCNT3   ;WAS TRIED TWICE, BUT "SIN".
2085 005274 001002          BNE      6#      ;ALLOW 5 ERRORS AT MOST
2086 005276 000137 005606  JMP      16#
2087 005302 005777 174150 68:  TST      @RKCS   ;"ERR" IN RKCS?
2088 005306 100010          BPL      7#      ;NO, BRANCH
2089 005310 004737 016136  JSR      PC,GTSRG ;GO, GET RKCS, ER,DS,DA,CYLNR
2090 005314 104012          ERROR    12     ;"ERR" SET WHILE DOING RD FMT
2091 005316 104415          CON,RESET        ;GO DO CNTRL RESET
2092 005320 005237 001506  INC      ERCNT2   ;ALLOW ONLY 12 ERRORS OF THIS
2093 005324 001532          BEQ      TST4    ;KIND, IF MORE THAN FIVE ERRORS
2094 005326 000520          BR       14#     ;SKIP THIS TEST
2095 005330 004737 007260 7#:  JSR      PC,CHKHDRS ;GO CHECK IF CORRECT HEADERS WERE READ
2096 005334 005737 001500  TST      INDX3   ;WAS THERE A MISCOMPARISON?
2097 005340 001513          BEQ      14#     ;IF NOT, GO SET UP TO RD FMT
2098 005342 012737 005170 001110 MOV      #2#,$LPERR ;NXT CYL IN LINE
2099 005350 104013          ERROR    13     ;CORRECT HDRS WERE NOT RECVD
2100 005352 005237 001254  INC      RETRY2   ;FROM SECTRS AS TYPED OUT,
2101 005356 022737 000002 001254 CMP      #2,RETRY2 ;THE SAME CYLINDER WAS READ TWICE
2102 005364 001301          BNE      2#      ;RETRY RD FMT ON SAME CYL AGAIN
2103 005366 005237 001504  INC      ERCNT1   ;TRIED RDING SAME CYL TWICE
2104 005372 001505          BEQ      16#     ;IF NOT, GO RD AGAIN
2105 005374 001505          BEQ      16#     ;YES, REPORT ERROR
2106 005376 001505          BEQ      16#     ;ALLOW ONLY 5 ERRORS OF THE
2107 005378 001505          BEQ      16#     ;ABOVE TYPE, IF MORE THAN 12
2108 005380 001505          BEQ      16#     ;EXIT THIS TEST
2109 005382 001505          BEQ      16#     ;THE PSUEDO-HEADERS (FIRST WORD OF EVERY
2110 005384 001505          BEQ      16#     ;SECTOR) FROM THIS CYLINDER (ABOVE,
2111 005386 001505          BEQ      16#     ;THE CYLINDER THAT GAVE WRONG HEADERS)
2112 005388 001505          BEQ      16#     ;WILL BE READ, NOW, FOLLOWING WILL B TYPD OUT;
2113 005390 001505          BEQ      16#     ;SEC#, EXPCD PSUEDO-HDR, RECVD PHDR,
2114 005392 001505          BEQ      16#     ;IF "INHIBIT TYPEOUT" SW IS SET THIS ENTIRE
2115 005394 001505          BEQ      16#     ;READING & TYPING WILL BE SKIPPED
2116 005396 001505          BEQ      16#     ;INHIBIT TYPEOUT?
2117 005398 001505          BEQ      16#     ;YES, SKIP THE FOLLOWING & GO
2118 005400 001505          BEQ      16#     ;SET UP TO RD FMT NXT CYL IN LINE
2119 005402 001505          BEQ      16#     ;READ FROM 12 SECTRS
2120 005404 001505          BEQ      16#
2121 005406 001505          BEQ      16#
2122 005408 001505          BEQ      16#
2123 005410 001505          BEQ      16#
2124 005412 001505          BEQ      16#
2125 005414 001505          BEQ      16#
2126 005416 001505          BEQ      16#
2127 005418 001505          BEQ      16#
2128 005420 001505          BEQ      16#
2129 005422 001505          BEQ      16#
2130 005424 001505          BEQ      16#
2131 005426 001505          BEQ      16#
2132 005428 001505          BEQ      16#
2133 005430 001505          BEQ      16#
2134 005432 001505          BEQ      16#

```

```

2135 005410 010577 174050          MOV      R5,@RKDA ;FROM THIS DSK=ADRES
2136 005414 012777 026446 174040 MOV      #OUTBUF,@RKBA ;INTO THIS BUS-ADRES
2137 005422 012777 177777 174030 10#: MOV      #-1,@RKWC  ;RD 1 WRD
2138 005430 012777 000005 174020 MOV      #5,@RKCS   ;GO, RD
2139 005436 104421          CON,RDY        ;WAIT FOR CNTRL RDY
2140 005440 005777 174012  TST      @RKCS   ;ANY EROR?
2141 005444 100002          BPL      15#     ;NO, PROCEED
2142 005446 104415          CON,RESET        ;CLEAR THE EROR
2143 005450 000447          BR       14#     ;ERROR, SO COULDN'T READ PSUEDO-HDRS
2144 005452 005201          INC      R1      ;READ FROM ALL 12 SECS
2145 005454 001362          BNE      10#     ;IF NOT GO RD THE NXT ONE
2146 005456 001362          BNE      10#
2147 005458 001362          BNE      10#
2148 005460 001362          BNE      10#
2149 005462 001362          BNE      10#
2150 005464 001362          BNE      10#
2151 005466 001362          BNE      10#
2152 005468 001362          BNE      10#
2153 005470 001362          BNE      10#
2154 005472 001362          BNE      10#
2155 005474 001362          BNE      10#
2156 005476 001362          BNE      10#
2157 005478 001362          BNE      10#
2158 005480 001362          BNE      10#
2159 005482 001362          BNE      10#
2160 005484 001362          BNE      10#
2161 005486 001362          BNE      10#
2162 005488 001362          BNE      10#
2163 005490 001362          BNE      10#
2164 005500 005702          TST      R2      ;IS THIS SEC # CORRESPONDING TO THE
2165 005502 001403          BEQ      13#     ;ONE IN ERROR
2166 005504 005302          DEC      R2      ;R2 CONTAINS THE SEC #
2167 005506 005723          TST      (R3)+   ;
2168 005510 000773          BR       12#     ;
2169 005512 001146          MOV      (R1),-(SP) ;GO TYPEOUT SEC # GIVING
2170 005514 104403          TYPOS         ;MISCOMPARISON OF HEADERS
2171 005516 002         ,BYTE 2
2172 005517 000         ,BYTE 0
2173 005518 000         ,BYTE 0
2174 005520 104401          TYPE          ;TYPE 2 BLANKS
2175 005522 002105          BLNKS5
2176 005524 010546          MOV      R5,-(SP) ;GO TYPE EXPCD PSUEDO HEADER
2177 005526 051116          BIS      (R1),(SP)
2178 005530 104402          TYPOC
2179 005532 104401          TYPE          ;TYPE 2 BLNKS
2180 005534 002103          BLNKS7
2181 005536 011346          MOV      (R3),-(SP) ;GO TYPE PSUEDO-HEADER RECVD
2182 005540 104402          TYPOC
2183 005542 005721          TST      (R1)+   ;TYPED OUT ALL SEC #'S IN ERROR,
2184 005544 021127 177777          CMP      (R1),#177777
2185 005550 001346          BNE      11#     ;IF NOT GO BAK & TYPE NXT

```

```
2191
2192 005552 104401 001733 TYPE ,MSG7 ;TYPE OUT PC
2193 005556 012746 005374 MOV #208,-(SP)
2194 005562 104402 TPOC
2195 005564 104401 001213 TYPE ,%CRLF ;TYPE ROUTINE ENDS HERE
2196
2197 ;FIND OUT NXT TRAK TO B READ
2198 ;FORMATTED
2199
2200
2201 005570 062705 000020 14$: ADD #20,R5 ;SET ADRES FOR SUR 0, NXT CYL IN LINE
2202 005574 005237 001476 INC INDX2 ;READ ALL 313 CYLINDERS (626 TRAKS)?
2203 005600 001404 BEQ TST4 ;EXIT
2204 005602 000137 005164 JMP 18 ;IF NOT, GO BAK & READ NXT
2205
2206 005606 004737 016772 16$: JSR PC,ABRT
2207
;*****
;TEST 4 SEEK PATTERNS: 0-312-0-311-...,USING IMPLIED SEEK
2208
;****TEST 2 (WRITING PSEUDO-HEADERS) SHOULD HAVE BEEN DONE BEFORE****
2209 ;**** DOING THIS TEST****
2210 ;THIS TEST PERFORMS SEEKS (IMPLIED SEEKS USING 'READS') IN THE
2211 ;FOLLOWING PATTERN.
2212 ;*0-312-0-311-0-310-...,0-1-0-0
2213
;*THE FIRST WORD OF EVERY SECTOR IS A PSEUDO-HEADER (WRITTEN IN
;*A PREVIOUS TEST) CONSISTING OF DRIVE NO, CYLINDER NO,, SURFACE
;*AND SECTOR NO. AN IMPLIED SEEK IS DONE BY ISSUING A 'READ' FOR
;*THE PSEUDO-HEADER OF SECTOR 0, SURFACE 0.
2214
;*IF A 'SIN' OCCURS TWO TRIES ARE DONE BEFORE ABORTING. IF A 'SKE'
;*OCCURS IT COULD MEAN THAT 1) EITHER THE HEADERS WAS READ WRONG
;*OR 2) THE HEADS GOT POSITIONED ON THE WRONG CYLINDER. IN
;*ORDER TO PROVIDE A FURTHER INSIGHT INTO THE PROBLEM, THE FOLLOWING
;*IS DONE:
;*THE HEADERS ARE READ FROM THE CYLINDER THAT GAVE 'SKE'. IF THE HEADERS
;*ARE CORRECT IT IS SO REPORTED. IF THE HEADERS ARE INCORRECT, THEN THE
;*EXPECTED HEADERS AND THE RECEIVED ONES ARE REPORTED. ONE MORE TRY IS
;*DONE (THE IMPLIED SEEK IS TRIED AGAIN BETWEEN THE CYLINDERS THAT GAVE RISE
;*TO 'SKE')
2215
;*THE FOLLOWING ACTION IS TAKEN WHEN THERE IS NO 'SKE' OR 'SIN' BUT STILL THE
;*PSEUDO-HEADER IS READ WRONG:
;*FIRST THE HEADERS ARE READ FROM THAT CYLINDER AND CHECKED. IF THEY ARE
;**CORRECT, IT IS SO REPORTED. IF THEY ARE WRONG THEN THE EXPECTED AND RECEIVED
2216
;*HEADERS ARE REPORTED. THEN THE PSEUDO-HEADER FROM SECTOR 1 IS READ AND REPORT
;*ONE MORE TRY IS DONE BY REPEATING THE WHOLE PROCESS. (IMPLIED SEEK
;*BETWEEN THE TWO CYLINDERS AND READING PSEUDO-HEADER FROM SECTOR 0 OF THE DESTI
;*CYLINDER).
2217
;*UP TO 12 ERRORS OF EACH KIND (SIN, SKE, BAD PSEUDO-HEADER) ARE ALLOWED.
;*IF ANY ERROR OCCURS MORE THAN 12 TIMES THE TEST IS ABORTED.
;*****
TST4: SCOPE
2218 CON,RESET ;GO DO CONTROL RESET
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245 005612 000004
2246 005614 104415
```

```
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 INNER
2260 ;CYL TO WHICH IMPLIED SEEK WILL
2261 ;BE DONE
2262
2263 005656 005704 18: TST R4 ;GOING IN OR OUT?
2264 005660 001005 BNE 28 ;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 38 ;CYLINDER
2268 005674 013705 001230 28: MOV DRIVAD,R5 ;FORM DISK ADRES FOR OUTER
2269 ;CYLINDER = 0
2270 ;ALLOW 2 TRIES WHEN
2271 ;ERRORS OCCUR
2272 005700 012737 177776 001254 38: MOV #-2,RETRY2
2273 005706 012737 177776 001252 138: MOV #-2,RETRY1
2274 005714 012737 177777 001256 48: MOV #-1,RETRY3
2275 005722 000404 BR 58
2276 005724 104415 68: CON,RESET
2277 005726 104416 DRV,RESET
2278 005730 004737 006526 JSR PC,SBR1 ;REPOSITION HEADS TO PRE-ERROR CYL
2279
2280 005734 012777 177777 173516 58: MOV #-1,@RKWC ;READ 1 WORD
2281 005742 010577 173516 MOV R5,@RKDA ;FROM THIS CYLINDER, SEC 0
2282 005746 012777 026446 173506 MOV @OUTBUF,@RKBA ;INTO THIS BUS ADRES
2283 005754 012737 005724 001110 MOV #68,@LPERR ;SET RETURN ADRES FOR LUPING
2284 ;ON 'ERROR'
2285
2286 005762 012777 000005 173466 MOV #5,@RKCS ;GO, READ
2287
2288 CON,RDY ;WAIT FOR CNTRL RDY
2289
2290 005772 032777 001000 173452 BIT #1000,@RKDS ;SIN?
2291 BEQ 88 ;NO, BRANCH
2292 ;YES, THERE WAS A SIN
2293 006002 004737 016376 JSR PC,ERR1 ;GO GET, CYLS BETW'N WHICH SK WAS TRIED
2294 006006 017737 173440 001170 MOV @RKDS,$REG3
2295 006014 017737 173434 001166 MOV @RKER,$REG2
2296 006022 104420 001602 TYPMSG ,MSG1
2297 006026 013737 001256 001172 MOV RETRY3,$REG4 ;SAVE TRY # ON 'SIN'
2298 006034 062737 000002 001172 ADD #2,$REG4
2299 ERROR 14 ;AN IMPLIED SEEK WAS TRIED
2300 ;FROM 'CYLA' TO 'CYLB' (INDICATED
2301 ;IN EROR MESSAGE), 'SIN' OCCURRED,
2302 ;2 TRIES ARE DONE BEFORE
;ABORTING
```



```

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,
                                ;READ PSUEDO-HEADER OF SEC 0,
                                ;CYLB (IN EROR MESSAGE), BUT
                                ;THE WRONG PSUEDO-HEADER WAS
                                ;RECEIVED
2422 006366 005237 001254      INC     RETRY2
2423 006372 001402                BEQ     21$
2424 006374 000137 005706      JMP     13$
2426 006400 004737 006552 21$: JSR     PC,SBR2      ;GO READ HEADERS (12) FROM
                                ;THIS CYLINDER, & CHECK THEM,
                                ;IF MISCOMPARISON INDX3 WILL
                                ;BE > 0.
2431 006404 005737 001500      TST     INDX3
2432 006410 001003                BNE     22$
2433 006412 104420 001657      TYPMSG ,MSG5        ;WRONG PSUEDO-HDR WAS READ
                                ;BUT WHEN HDRS WERE READ
                                ;FROM THE SAME CYLINDER, THEY
                                ;WERE CORRECT
2436 006416 000402                BR      23$
2438 006420 104417 000015 22$: MESSAGE ,15      ;WRONG PSUEDO-HDR WAS READ
                                ;FROM "CYLB" (IN ERROR MESSAGE),
                                ;THEN HEADERS WERE READ FROM THE
                                ;SAME CYLINDER. THEY WERE ALSO
                                ;WRONG.
2443 006424 010500 23$: 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 EXPCTD PSUEDO-HDR FROM SEC 1
2453 006474 013737 026446 001166  MOV     OUTBUF,$REG2
2454 006502 104417 000017      MESSAGE ,17        ;PSUEDO-HEADER FROM SEC 1, CYLB
                                ; (IN MESSAGE) WAS READ. THE EXPCTD
                                ;& RECVD DATA WORDS ARE REPORTED,
                                ;ALLOW ONLY LESS THAN 10 ERRORS
                                ;OF THIS TYPE (WRONG PS-HDRS)
2455
2456
2457 006506 005237 001510      INC     ERCNT3
2458
2459 006512 001440                BEQ     EXT4
2460
2461 006514 005704 24$: TST     R4            ;SEEKED IN OR OUT LAST TIME?
2462 006516 001266                BNE     19$          ;IF OUT, GO SEEK NXT INNER CYL
2463
2464 006520 005204                INC     R4            ;INDICATE THAT SEEK OUT (0)
2465 006522 000137 005656      JMP     1$            ;WILL BE DONE NOW
2466
2467
2468
2469
2470
                                ;THIS ROUTINE IS USED IN THIS TEST ONLY.
                                ;R4=0 INDICATES SEEK BEING DONE FROM
                                ;CYL 0 TO INNER CYL.

```

```

2471
2472
2473
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      TST,RWS
2480 006550 000207 1$: RTS     PC
2481
2482
2483
2484
2485
2486
2487
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
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
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"

```

```

;*****
;TEST 5 PERFORM CONVERGING-DIVERGING (IMPLIED) SEEKS
;THIS TEST PERFORMS A CONVERGING-DIVERGING SEEK PATTERN
;USING IMPLIED SEEK (READ FORMAT). THE SEEK SEQUENCE IS:
;0-312-1-311-2-310-3-307-----310-2-311-1-312
;ALL READ FORMATS ARE DONE FROM SURFACE 0.
;THE CYLINDER ADDRESSES, BETWEEN WHICH THE IMPLIED SEEK IS
;PERFORMED, ARE CONTAINED IN "OUTADR" & "INADR". IF "SIN" OCCURS
;AN ERROR IS REPORTED AND A RETRY IS DONE. ON READING INCORRECT
;HEADERS AN ERROR IS REPORTED AND A RETRY IS DONE. NOTE THAT IF
;ALL THE 12 HEADERS ARE INCORRECT, IT COULD MEAN THAT THE HEADS
;COULD NOT POSITION CORRECTLY. THIS WOULD BE CONFIRMED IF IN
;PREVIOUS TESTS BAD HEADERS WERE NOT RECEIVED FROM THE SAME
;CYLINDER. IF THAT CYLINDER GAVE BAD HEADERS IN ALL THE PREVIOUS
;TESTS THE PROBLEM COULD BE DIFFERENT.
;MAXIMUM 12 ERRORS OF ANY KIND ARE ALLOWED.
;IF MORE THAN 12 ERRORS OCCUR THE TEST IS ABORTED.
;*****

```



```

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 MOV    #14500,INADR ;INITIALIZE 'INADR' TO 312
2535
2536 006664 005704 19:   TST    R4            ;GOING IN OR OUT?
2537 006666 001005 BNE    28           ;GOING OUT,BRANCH
2538 006670 013705 MOV    INADR,R5      ;SET CYL ADRES BITS FOR GOING IN
2539 006674 053705 BIS    DRIVAD,R5    ;FORM DISK ADRES FOR INNER CYLINDER
2540 006700 000404 BR     38           ;
2541
2542 006702 013705 28:   MOV    OUTADR,R5     ;SET CYL ADRES BITS FOR GOING OUT
2543 006706 053705 BIS    DRIVAD,R5    ;FORM DISK ADRES FOR GOING OUT
2544
2545 006712 005037 001254 CLR    RETRY2        ;ALLOW 2 RETRIES
2546 006716 012737 177777 MOV    #-1,RETRY1   ;WHEN ERRORS OCCUR
2547 006724 000404 BR     76           ;
2548 006726 104415 58:   CON,RESET          ;
2549 006730 104416 DRV,RESET          ;
2550 006732 004737 JSR    PC,SBR3     ;GO REPOSITION HEADS
2551
2552 006736 012777 177764 78:   MOV    #-14,@RKWC   ;READ ALL HDRS FROM THIS CYLINDER
2553 006744 010577 MOV    R5,@RKDA     ;FROM THIS CYL, SEC 0
2554 006750 012777 MOV    #OUTBUF,@RKBA ;INTO THIS BUS ADRES
2555 006756 012737 MOV    #5,@LPERR    ;SET RETURN ADRES FOR LOOPING ON ERROR
2556
2557 006764 012777 002005 MOV    #2005,@RKCS  ;READ FORMAT,GO
2558
2559 006772 104421 CON,RDY           ;WAIT FOR CONTRL RDY
2560
2561 006774 032777 001000 BII    #1000,@RKDS  ;SIN?
2562 007002 001443 BEQ    88           ;NO, BRANCH
2563 007004 017737 MOV    @RKER,@REC2  ;SAVE RKER
2564 007012 017737 MOV    @RKDS,@REG3   ;SAVE RKDS
2565 007020 013737 MOV    RETRY1,@REG4  ;GET RETRY #
2566 007026 062737 ADD    #2,@REG4     ;
2567 007034 094737 JSR    PC,ERR2     ;GET CYL #'S BELOW 'N WHICH
2568                                     ;SEEK WAS TRIED
2569 007040 104420 TYPMSG ,MSG1       ;TYPE 'SIN'
2570 007044 194014 ERROR  14          ;
2571                                     ;'SIN' OCCURRED ON DOING IMPLIED
2572                                     ;SEEK FROM 'CYLA' TO 'CYLB' (IN
                                     ;EROR MESSAGE).
2573                                     ;DONE 2 TRIES?
2574 007046 005737 TST    RETRY1        ;YES, BRANCH
2575 007052 001403 BEQ    68           ;NO, RETRY
2576 007054 005237 INC    RETRY1        ;
2577 007060 000722 BR     58           ;
2578 007062 194415 68:   CON,RESET          ;
2579 007064 104416 DRV,RESET          ;
2580
2581
2582 007066 005237 INC    ERCNT1        ;ALLOW LESS THAN 12 ERORS OF THE
    
```

```

2583 007072 001527 BEQ    EXTS         ;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 MOV    R5,@RKDA     ;SET CYL ADRES
2590 007100 012777 MOV    #11,@RKCS    ;SEEK,GO
2591 007106 104422 TST,RWS          ;
2592 007110 000424 BR     11$        ;
2593
2594 007112 004737 JSR    PC,CHKHDRS   ;IF NO SIN, ENTER HERE
2595                                     ;GO CHECK IF CORRECT HEADERS WERE READ
2596
2597 007116 012737 MOV    #4,@LPERR    ;SET LUP ADDRESS
2598 007124 005737 TST    INDX3        ;WAS THERE A BAD HDR?
2599 007130 001414 BEQ    11$        ;NO, BRANCH
2600 007132 104020 ERROR  20          ;WRONG HEADERS WERE READ FROM
2601                                     ;SEC #'S, ON DOING AN IMPLIED
2602                                     ;SEEK (RDFMT) FROM 'CYLA' TO 'CYLB'
2603 007134 005237 INC    RETRY2        ;ALLOW 2 TRIES
2604 007140 022737 CMP    #2,RETRY2    ;
2605 007146 001263 BNE    48           ;GO TRY 2ND TIME
2606 007150 005237 INC    ERCNT2        ;ALLOW ONLY 12 ERRORS
2607 007154 001002 BNE    11$        ;
2608 007156 000137 JMP    EXTS         ;IF MORE, EXIT THIS TEST
2609
2610 007162 005704 118:  TST    R4            ;GOING WHICH WAY?
2611 007164 001006 BNE    128        ;'INADR' TO 'OUTADR', BRANCH
2612                                     ;'OUTADR' TO 'INADR'
2613                                     ;INDICATE THAT NXT TIME GOING
2614 007170 062737 INC    R4            ;FROM 'INADR' TO 'OUTADR'
2615 007176 000137 ADD    #40,OUTADR    ;INCREMENT CYLINDER ADRES
2616                                     ;GO BAK & DO IMPLIED SEEK
2617                                     ;FROM 'INADR' TO 'OUTADR'
2618
2619 007202 005004 128:  CLR    R4            ;INDICATE THAT NXT TIME GOING
2620                                     ;FROM 'OUTADR' TO 'INADR'
2621 007204 162737 SUB    #40,INADR     ;DECREMENT CYLINDER ADRES
2622 007212 005237 INC    INDX2        ;DONE ALL 312 FORWARD-BACKWARD
2623                                     ;SEEK PATTERNS
2624 007216 001457 BEQ    TST6         ;IF YES, EXIT
2625
2626 007220 000137 JMP    18           ;IF NOT, GO BAK & DO IMPLIED
2627                                     ;SEEK FROM 'OUTADR' TO 'INADR'
2628
2629 ;THIS SUBROUTINE IS ENTERED AFTER A 'SIN' ERROR OCCURED ON GOING FROM
2630 ;'CYLA' TO 'CYLB' AS INDICATED IN THE ERROR MESSAGE, BEFORE RETRYING THE
2631 ;HEADS HAVE TO BE POSITIONED BACK TO 'CYLA', NOTE THAT A DRIVE RESET
2632 ;WAS DONE TO CLEAR SIN.
2633 ;R4=0, INDICATES SEEK IS BEING DONE FROM 'OUTADR' CYLINDER TO 'INADR' CYLINDER,
2634 ;R4=1, INDICATES THAT SEEK IS BEING DONE FROM 'INADR' TO 'OUTADR'.
2635 SBR3: TST    R4            ;GOING WHICH WAY?
2636 BEQ    18           ;IF FROM 'OUTADR' TO 'INADR', BRANCH.
2637 MOV    INADR,@RKDA  ;
2638 BR     28           ;
    
```

2639 007246 012777 000011 172202 28: MOV #11,@RKCS  
2640 007254 104422 TST,RWS  
2641 007256 000207 RTS PC  
2642  
2643  
2644  
2645  
2646  
2647  
2648  
2649  
2650  
2651  
2652  
2653  
2654  
2655  
2656  
2657  
2658 007260 005000  
2659 007262 012701 026446  
2660 007266 012702 001266  
2661 007272 012703 001320  
2662 007276 010537 001120  
2663 007302 042737 160037 001120  
2664 007310 005037 001500  
2665  
2666 007314 023711 001120  
2667 007320 001406  
2668 007322 011123  
2669 007324 010022  
2670  
2671 007326 012712 177777  
2672 007332 005237 001500  
2673 007336 005721  
2674 007340 005200  
2675 007342 022700 000014  
2676 007346 001362  
2677 007350 000207  
2678  
2679 007352 004737 016772  
2680  
2681  
2682  
2683  
2684  
  
2685  
2686  
2687  
2688  
2689  
2690  
2691  
2692  
2693  
2694

```
;CHKHDRS  
;THIS ROUTINE CHECKS THAT THE HEADERS READ PREVIOUSLY WERE CORRECT.  
;IF NOT, THE BAD HEADERS AND THE SECTOR #'S FROM WHICH THEY WERE READ  
;ARE STORED.  
;ON ENTRY:  
;R5 CONTAINS DISK ADDRESS FROM WHERE HEADERS WERE READ.  
;OUTBUF = 12 HEADERS READ PREVIOUSLY ARE STORED STARTING "OUTBUF".  
;ON EXIT:  
;INDX3=0, IF THE HEADERS WERE CORRECT  
;INDX3=1, IF THE HEADERS WERE INCORRECT  
;BUFR = SECTOR #'S GIVING BAD HEADERS ARE STORED STARTING AT "BUFR".  
;BUFR1 = BAD HEADERS FOR THE ABOVE SECTORS ARE STORED STARTING "BUFR1".  
;THE BAD SECTOR TABLE IS TERMINATED BY A '177777' WORD.  
  
CHKHDRS: CLR R0 ;INITIALIZE FOR 14 HDRS  
MOV #OUTBUF,R1 ;INITIALIZE PTR TO HDRS RECVD  
MOV #BUFR,R2 ;INITIALIZE PTR TO SECTOR TABLE  
MOV #BUFR1,R3 ;INITIALIZE PTR TO BAD HDR TABLE  
MOV R5,#GDADR ;GET EXPCTD HEADER  
CLR INDX3 ;CLR FLG INDICATING BAD HDRS  
  
98: CMP #GDADR,(R1) ;HEADER OK?  
BEQ 108 ;YES,BRANCH  
MOV (R1),(R3)+ ;SAVE BAD HDR  
MOV R0,(R2)+ ;SAVE BAD SECTR #  
  
108: MOV #177777,(R2) ;PUT TERMINATR ON SECTR TABLE  
INC INDX3 ;SET FLG INDICATING BAD HDR  
TST (R1)+ ;INCRMNT PTR TO NXT HDR  
INC R0 ;ALL HDRS CHKD?  
CMP #14,R0  
BNE 98 ;IF NOT, LUP BAK  
RTS PC  
  
EXTS: JSR PC,ABRT
```

2695  
2696  
2697  
2698  
2699  
2700  
2701  
2702  
2703  
2704  
2705  
2706  
2707  
2708  
2709  
2710  
2711  
2712  
2713  
2714  
2715  
2716  
2717  
2718  
2719 007356 000004  
2720 007360 012737 177764 001504  
2721 007366 012737 177764 001506  
2722 007374 012737 177152 001474  
2723 007402 005037 001410  
2724 007406 005037 001412  
2725  
2726  
2727  
2728 007412 013737 001230 001432  
2729 007420 012737 001414 001424  
2730  
2731 007426 004737 010044  
2732 007432 017737 171766 001430  
2733 007440 004777 171764  
2734  
2735 007444 005005  
2736 007446 005737 001410  
2737 007452 100407  
2738 007454 013737 001406 001434  
2739  
2740 007462 052737 000200 001412  
2741  
2742  
2743 007470 000406  
2744  
2745 007472 013737 001404 001434  
2746  
2747  
2748 007500 052737 000200 001410  
2749  
2750 007506 012737 007514 001110 138:

```
;CYL SECTORS CYL SECTORS CYL SECTORS CYL SECTORS  
;* SUR ROUTINE SUR ROUTINE SUR ROUTINE SUR ROUTINE  
;* 0 0 0-2 PTGEN0 0 0 6-10 PTGEN1 0 0 3-5 PTGEN2 0 0 11-13 PTGEN3  
;* 0 1 0-2 PTGEN0 0 1 6-10 PTGEN1 0 1 3-5 PTGEN2 0 1 11-13 PTGEN3  
;* 1 0 0-2 PTGEN0 1 0 6-10 PTGEN1 1 0 3-5 PTGEN2 1 0 11-13 PTGEN3  
;* 1 1 0-2 PTGEN0 1 1 6-10 PTGEN1 1 1 3-5 PTGEN2 1 1 11-13 PTGEN3  
;* 2 0 0-2 PTGEN0 2 0 6-10 PTGEN1 2 0 3-5 PTGEN2 2 0 11-13 PTGEN3  
;ETC, ETC,....  
;THE ABOVE SHOWN STAGGERING (OF SECTORS TO BE WRITTEN) IS DONE TO  
;SAVE ONE REV (40MS) EVERY TIME A BLOCK (3 SECTORS) IS WRITTEN.  
;IF YOU WANT TO USE BUFFERS STARTING AT SOME OTHER MEMORY ADDRESS  
;MAKE THE FOLLOWING CHANGES:  
;CHANGE "PBUF0" TO STARTING ADDRESS OF THE FIRST 768 WORDS LONG BUFFER.  
;CHANGE "PBUF1" TO STARTING ADDRESS OF SECOND 768 WORDS LONG BUFFER.  
  
;IF YOU WANT TO WRITE YOUR OWN PATTERNS USING PATTERN GENERATOR "PTGEN0"  
;CHANGE "PTRN01" AND "PTRN02" TO THE PATTERNS YOU WANT.  
  
;TO WRITE THE SAME TWO (OR ONE) PATTERNS ON THE ENTIRE DISK CHANGE  
;LOCATION "PAT1" TO "PTGEN0" THE STARTING ADDRESS OF PAT-GENERATOR 0  
;LOCATION "PAT2" TO "PTGEN0" THE STARTING ADDRESS OF PAT-GENERATOR 0  
;LOCATION "PAT3" TO "PTGEN0" THE STARTING ADDRESS OF PAT-GENERATOR 0  
  
;*****  
TST6: SCOPE  
MOV #14,ERCNT1  
MOV #14,ERCNT2  
MOV #626,INDX1 ;SET COUNT FOR 313X2 TRACKS  
CLR BUFLG0 ;CLR FLAG FOR BUFR 0  
CLR BUFLG1 ;CLR FLAG FOR BUFR 1  
;BIT 7 OF ABOVE FLAGS WHEN SET  
;INDICATES, THAT BUFR TO BE USED  
;FOR WRITING ON DSK  
MOV DRIVAD,DSKADR ;GET DRIVE #, DISK ADRES  
MOV #PATO,PRSPAT ;INITIALIZE PTR TO THE FIRST  
;PATRN GENERATOR  
  
JSR PC,GETBUF  
MOV #PRSPAT,PGSUBR  
JSR PC,@PGSUBR ;GO GENERATE PATRNS FOR  
;3 SECTOR (400X3)  
CLR R5 ;INITIALIZE COUNT FOR 4 BLOCKS  
TST BUFLG0 ;FIND OUT WHICH BUFR TO USE  
BMI 38 ;FOR WRITING ON DSK  
MOV PBUF1,BUSADR ;USE "IOBUF1" FOR TRANSFER  
;OR THE ONE INDICATED BY THE USER  
BIS #BIT7,BUFLG1 ;SET FLAG TO INDICATE THAT  
;WRITING ON DSK WILL B DONE FROM  
;THIS BUFR (BUFR 1)  
  
BR 138  
  
38: MOV PBUF0,BUSADR ;USE "IOBUF0" FOR TRANSFER  
;OR THE ONE INDICATED BY THE USER  
  
BIS #BIT7,BUFLG0 ;INDICATE THAT "IOBUF0" WILL  
;B USED FOR WRITING ON DISK  
  
138: MOV #48,$LPERR
```

```

2751 007514 013777 001432 171742 48: 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 56 ;TO B WRITTEN
2759 007562 012737 001414 001426 MOV #PATO,NXTPAT ;KEEP GENERATING PATRNS IN THIS
2760 007570 000403 BR 66 ;WAY "PATO"="PAT1"="PAT2"="PAT3"="PATO"=
2761 007572 062737 000002 001426 58: ADD #2,NXTPAT
2762 007600 004737 010044 68: 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 BIT #140000,@RKCS ;ANY ERROR?
2768 007626 001411 BEQ 128 ;GET RKCS,ER,DS,DA
2769 007630 004737 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 128
2774 007646 000137 010432 JMP EXT6 ;IF MORE, EXIT
2775 007652 032777 001000 171572 128: BIT #BIT9,@RKDS
2776 007660 001412 BEQ 76 ;SIN, ON DOING WRITE?
2777 007662 004737 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 76
2783 007702 000137 010432 JMP EXT6
2784 ;FIGURE OUT WHICH BUFFER IS
2785 ;AVAILABLE FOR USE
2786 007706 105737 001410 78: TSTB BUFLG0 ;WAS PREVIOUS DSK-WRITE DONE
2787 007712 100003 BPL 86 ;USING BUFR 0?
2788 007714 005037 001410 CLR BUFLG0 ;YES, CLR FLAG INDICATING IT'S
2789 ;AVAILABLE NOW
2790 007720 000402 BR 96
2791 007722 005037 001412 86: CLR BUFLG1 ;CLR FLAG INDICATING BUFR1
2792 ;IS AVAILABLE NOW
2793 007726 013737 001426 001424 98: 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 MOV#B SECPTR(R0),R0 ;GET SEC #
2797 007742 042737 000017 001432 106: 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 28 ;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 118 ;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, NEXT CYLINDER
2809 010014 062737 000040 001432 ADD #40,DSKADR
2810 010022 000137 007444 JMP 18 ;GO BACK AND DO WRITE
2811 010026 005237 001474 118: INC INDX1 ;COUNT # OF TRACKS
2812 010032 052737 000020 001432 BIS #20,DSKADR ;SET BIT FOR SUR 1
2813 010040 000137 007444 JMP 16 ;GO, WRITE PATRNS ON SURFACE 1
2814 ;*GET BUF#
2815 ;*THIS ROUTINE FINDS OUT WHICH BUFFER (IOBUFO, 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 18 ;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 18: BIS #BIT15,BUFLG0 ;INDICATED, USING BUFR 0
2827 010076 013737 001404 001446 MOV PBUFO,BUFNO ;SAVE STARTING ADRES OF BUFR 0
2828 010104 000207 RTS PC ;RETURN
2829
2830 ;*PTGENO
2831 ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS LONG
2832 ;*BUFFER CONTAINING THE FOLLOWING PATTERNS
2833 ;*FIRST BLOCK OF 256 WORDS: 125252
2834 ;*SECOND BLOCK OF 256 WORDS: 052525 (COMPLEMENT OF ABOVE)
2835 ;*THIRD BLOCK OF 256 WORDS: 010421
2836 ;*YOU CAN USE ANY OTHER PATTERN(S (& ITS COMPLEMENT)
2837 ;*MAKING THE CHANGES IN THE 2 LOCATIONS SHOWN BELOW.
2838 010106 013700 001446 PTGEN0: MOV BUFRNO,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 18: MOV R1,(R0)+ ;GENERATE THE FIRST BLOCK
2843 010124 005202 INC R2 ;WITH "PAT01" PATRN
2844 010126 001375 BNE 18 ;ALL DONE?
2845
2846 010130 012702 177400 MOV #=400,R2
2847 010134 005101 COM R1 ;COMPLEMENT "PAT01" PATAN
2848 010136 010120 28: MOV R1,(R0)+ ;GENERATE 2ND BLOCK WITH
2849 010140 005202 INC R2 ;"PAT01'S COMPLEMENT PATRN
2850 010142 001375 BNE 28 ;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 38: MOV R1,(R0)+ ;GENERATE 3RD BLOCK USING
2855 ;"PAT02" PATRN
2856 010156 005202 INC R2
2857 010160 001375 BNE 38 ;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
2865 ;*PTGEN1
2866 ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS
2867 ;*LONG BUFFER CONTAINING THE FOLLOWING PATTERNS:
2868
2869 ;*FIRST BLOCK=256 WORDS 000001 FILL 1'S
2870 ;* 000003
2871 ;* 177777
2872
2873 ;*SECOND BLOCK 177776 FILL 0'S
2874 ;* 177774
2875 ;* 1
2876 ;* 000000
2877
2878 ;*THIRD BLOCK 000001 FLOAT A 1
2879 ;* 000020
2880 ;* 1
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 26
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 46
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 66
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 ;* 1
2924 ;* 000377
2925
2926 ;*SECOND BLOCK 000000 COUNT PATRN=HIGHER BYTE
2927 ;* 000400 0-377
2928 ;* 001000
2929 ;* 1
2930 ;* 177400
2931
2932 ;*THIRD BLOCK 000000 COUNT PATRN=HIGHER & LOWER BYTE
2933 ;* 000401 0-377, 0-377
2934 ;* 1
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 1$: MOV BUFNO,R0
2942 010300 010120 MOV R1,(R0)+ ;GENERATE 1ST BLOCK USING
2943 010302 105201 INCB R1 ;USING "COUNT UP LOWER BYTE"
2944 010304 001375 BNE 1$ ;DONE?
2945
2946 010306 005001 2$: CLR R1
2947 010310 010120 MOV R1,(R0)+ ;GENERATE 2ND BLOCK
2948 010312 062701 000400 ADD #400,R1 ;USING "COUNT UP HIGHER BYTE"
2949 010316 103374 BCC 26 ;DONE?
2950 010320 005001 CLR R1
2951 010322 010120 3$: MOV R1,(R0)+ ;GENERATE 3RD BLOCK USING
2952 010324 062701 000401 ADD #401,R1 ;"COUNT UP HIGHER & LOWER BYTE"
2953 010330 103374 BCC 36 ;ALL DONE?
2954 010332 000207 RTS PC ;RETURN
2955
2956 ;*PTGEN3
2957 ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS LONG
2958 ;*BUFFER CONTAINING THE FOLLOWING PATTERNS:
2959
2960 ;*FIRST BLOCK OF 256 WORDS: 167356 (COMPLEMENT OF 010421)
2961
2962 ;*SECOND BLOCK 177776 FLOAT A 0
2963 ;* 177775
2964 ;* 1
2965 ;* 077777
2966
2967 ;*THIRD BLOCK 000377 COUNT UP HIGHER BYTE 0-377
2968 ;* 000776 COUNT DOWN LOWER BYTE 377-0
2969 ;* 1
2970 ;* 177400
2971
2972 ;*BUFNO* CONTAINS THE STARTING ADDRESS OF THE BUFFER.
2973
2974

```

2975 010334 013700 001446  
2976 010340 012702 177400  
2977 010344 013701 010166  
2978 010350 005101  
2979 010352 010120  
2980 010354 005202  
2981 010356 001375  
2982  
2983  
2984 010360 012702 177760  
2985 010364 000261  
2986 010366 012701 177776  
2987 010372 010120  
2988 010374 006101  
2989 010376 103775  
2990 010400 005202  
2991 010402 001370  
2992  
2993  
2994 010404 012701 000377  
2995 010410 010102  
2996 010412 010120  
2997 010414 060201  
2998 010416 022701 177777  
2999 010422 001373  
3000 010424 000207  
3001  
3002  
3003  
3004 010426 006  
3005 010427 003  
3006 010430 011  
3007 010431 000  
3008  
3009 010432 004737 016772  
3010  
3011  
3012  
3013  
3014  
3015  
3016  
3017  
3018  
3019  
3020  
3021  
3022  
3023  
3024  
3025  
3026  
3027  
3028  
3029  
3030

PTGEN3: MOV BUFND,R0  
MOV #=400,R2  
MOV PTRN02,R1 ;GET PATTERN  
COM R1 ;COMPLEMENT "PAT02" PATRN  
48: MOV R1,(R0)+ ;GENERATE 1ST BLOCK  
INC R2  
BNE 48 ;ALL DONE?  
;2ND BLOCK  
78: MOV #=20,R2  
SEC  
MOV #177776,R1  
88: MOV R1,(R0)+  
ROL R1  
RCS 88  
INC R2  
BNE 78 ;ALL DONE?  
;3RD BLOCK  
98: MOV #377,R1  
MOV R1,R2 ;GENERATE 3RD BLOCK USING  
MOV R1,(R0)+ ;"COUNT DOWN LOWER BYTE"  
ADD R2,R1 ;"COUNT UP HIGHER BYTE"  
CMP #=1,R1  
BNE 98 ;ALL DONE?  
RTS PC ;RETURN  
;SECTOR POINTER TABLE, DATA TRANSFERS ARE DONE IN BLOCKS (3 SECTORS  
;EACH) IN THE CYCLIC ORDER: 0=2, 6=10, 3=5, 11=13, 0=2, AND SO ON,  
SECPTR: .BYTE 6  
.BYTE 3  
.BYTE 11  
.BYTE 0  
EXT6: JSR PC,ABRT

;;\*\*\*\*\*  
;TEST 7 READ, SOFTWARE COMPARE, WRITE CHECK OF THE PATTERNS  
;\*\*\*\*TEST 6 (WRITING PATTERNS ON DISK) SHOULD BE DONE BEFORE DOING\*\*\*\*  
;\*\*\*\*THIS TEST.\*\*\*\*  
;\*IN THIS TEST THE PATTERNS THAT WERE WRITTEN BEFORE (IN THE  
;\*PREVIOUS TEST) ARE READ BACK AND SOFTWARE COMPARED,  
;\*WHILE THE SOFTWARE COMPARISON IS TAKING PLACE, AN OVERLAPPING  
;\*WRITE CHECK IS DONE FOR THE SAME BLOCK. THE READING  
;\*BACK OF EACH BLOCK (4 SECTORS) IS DONE IN THE SAME  
;\*MANNER AS DESCRIBED IN THE BEGINNING OF PREVIOUS TEST,  
;\*OVERLAPPING OPERATIONS AND STAGGERING OF BLOCKS ARE DONE IN  
;\*A SIMILIAR MANNER.  
;\*THE FOLLOWING IS A DESCRIPTION OF HOW ERRORS ARE HANDLED,  
;\*IF A "SIN" OR "HE" OCCURS IT IS REPORTED AND THAT BLOCK  
;\*IS SKIPPED. IN THIS STAGE OF TESTING THESE ERRORS SHOULD NOT  
;\*NORMALLY OCCUR.  
;\*IF A CHECKSUM ERROR OCCURS, CONTROL IS TRANSFERRED TO

3031  
3032  
3033  
3034  
3035  
3036  
3037  
3038  
3039  
3040  
3041  
3042  
3043  
3044  
3045  
3046  
3047  
3048  
3049  
3050  
3051  
3052  
3053  
3054  
3055 010436 000004  
3056 010440 012737 177764 001510  
3057 010446 012737 177773 001512  
3058 010454 012737 177742 001514  
3059 010462 012737 177742 001516  
3060 010470 012737 177764 001520  
3061 010476 012737 177742 001522  
3062 010504 012737 17152 001474  
3063 010512 005037 001476  
3064  
3065 010516 012737 001414 001424  
3066 010524 013737 001230 001450  
3067  
3068  
3069 010532 005037 001504  
3070 010536 005037 001506  
3071 010542 012737 177775 001252  
3072 010550 012737 177776 001254  
3073 010556 012737 000003 001256  
3074  
3075 010564 013737 001450 001440  
3076 010572 013737 001406 001442  
3077  
3078 010600 012737 176400 001444  
3079  
3080 010606 013737 001450 001432  
3081 010614 012737 176400 001436  
3082 010622 013737 001404 001434  
3083  
3084  
3085 010630 000404  
3086

;;\*\*CSE\*\* FIRST THE CSE IS REPORTED. THE SECTOR GIVING  
;\*CSE IS READ AGAIN. IN ALL 3 TRIES ARE DONE. IF STILL  
;\*THE ERROR PERSISTS 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.  
;\*\*\*\*\*  
TST7: SCOPE  
MOV #=14,ERCNT3 ;ALLOW 12 ERRORS AT THE MOST  
MOV #=5,ERCNT4 ;ALLOW 5 ERRORS AT THE MOST  
MOV #=36,ERCNT5 ;ALLOW 10 ERRORS AT THE MOST  
MOV #=36,ERCNT6 ;ALLOW 10 ERRORS AT THE MOST  
MOV #=14,ERCNT7 ;ALLOW 12 ERRORS AT THE MOST  
MOV #=36,ERCNT8 ;ALLOW 10 ERRORS AT THE MOST  
MOV #=626,INDX1 ;SET COUNT FOR 626 TRACKS  
CLR INDX2 ;CLR COUNT FOR 4 BLOCKS ON A TRACK  
MOV #PATO,PRSPAT ;INITLZE PTR TO PATRN GENRTR  
MOV DRIVAD,ADRES ;INITLZE DRV#,ADRES  
BEGIN: CLR ERCNT1 ;IF > 0, MEANS THAT RETRIES  
CLR ERCNT2 ;DONE AFTER CSE OR CSE CHKD  
MOV #=3,RETRY1 ;RETRY COUNT FOR CSE  
MOV #=2,RETRY2 ;RETRY COUNT FOR SFTWRE MISCMP\*  
MOV #3,RETRY3 ;RETRY COUNT FOR WCE  
MOV ADRES,WDSKAD ;DISK ADRES TO WRT CHK WITH  
MOV PBUF0,WBUSAD ;USE THIS BUFR 1 TO WRT CHK  
;OR THE BUFR INDICATED BY THE USER  
MOV #=1400,WDRDCN ;WRT CHK 1 BLOCK=3SECS=1400 WRDS  
READ: MOV ADRES,DSKADR ;DISK ADRES TO READ FROM  
MOV #=1400,WRCMT ;1 BLOCK = 3 SECTORS = 1400 WRDS  
MOV PBUF0,BUSADR ;USE "IOBUFO" TO READ INTO  
;OR THE BUFR INDICATED BY THE USER  
BR RDAGAIN

```

3087 010632 104415
3088 010634 104416
3089 010636 000401
3090
3091 010640 104415
3092
3093 010642 013777 001432 170614
3094 010650 013777 001436 170602
3095 010656 013777 001434 170576
3096
3097 010664 012777 000405 170564
3098
3099
3100 010672 013737 001406 001446
3101 010700 017737 170520 001430
3102 010706 004777 170516
3103
3104
3105
3106 010712 104421
3107
3108
3109
3110
3111 010714 032777 040000 170534
3112 010722 001416
3113
3114 010724 012737 010640 001110
3115 010732 004737 016214
3116 010736 104023
3117 010740 104415
3118 010742 005237 001510
3119 010746 001002
3120 010750 000137 012170
3121 010754 000137 011512
3122 010760 032777 001000 170464
3123 010766 001417
3124
3125 010770 012737 010632 001110
3126 010776 004737 016214
3127 011002 104011
3128 011004 104415
3129 011006 104416
3130 011010 005237 001512
3131 011014 001002
3132 011016 000137 012170
3133 011022 000137 011512
3134
3135 011026 005737 001504
3136 011032 001031
3137 011034 005237 001504
3138
3139
3140 011040 032777 000002 170406
3141 011046 001423
3142 011050 012737 010532 001110
LUPSIN: CON,RESET
          DRV,RESET
          BR          RDAGAIN
LUPHE: CON,RESET
RDAGAIN: MOV      MOV      DSKADR,@RKDA      ;READ FROM THIS DSK=ADRES
          WRDcnt,@RKWC      ;THIS # OF WORDS
          BUSADR,@RKBA      ;INTO THIS BUFR
          MOV      #405,@RKCS      ;READ,SSE,GO
          MOV      PBUF1,BUFND      ;SET UP STARTING ADPES
          JSR      @PRSPAT,PGSUBR
          JSR      PC,@PGSUBR      ;GO GENERATE A BUFFER
          ;OF 1400 WORDS USING THIS
          ;PATRN GENRTR
CON,RDY      ;DONE WITH PATRN GENRTRG,
             ;WAIT FOR CNT RDY TO SET
             ;(FROM PREVIOUS READ)
          ;CNT RDY SET
          BIT      #BIT14,@RKCS      ;HARD ERROR?
          BEQ      NOHE
          MOV      #LUPHE,$LPERR
          JSR      PC,GETINF
          ERROR    23      ;HARD ERROR
          CON,RESET
          INC      ERCNT3      ;ALLOW 12 ERRORS AT MOST
          BNE     18
          JMP      EXT7      ;IF MORE, EXIT
          JMP      FINISH      ;IF MORE, EXIT
          NOHE:   BIT      #BIT9,@RKDS      ;SIN SET?
          BEQ      NOSIN        ;NO
          MOV      #LUPSIN,$LPERR
          JSR      PC,GETINF
          ERROR    11      ;SIN ON READ
          CON,RESET
          DRV,RESET
          INC      ERCNT4      ;ALLOW 5 ERRORS AT MOST
          BNE     18
          JMP      EXT7      ;IF MORE, EXIT
          JMP      FINISH
          NOSIN:  TST      ERCNT1      ;CHECKING CSE FOR 1ST TIME
          BNE     WRTCHK        ;NO,BRANCH
          INC      ERCNT1      ;INDICATE THAT CSE HAS BEEN
          ;CHECKED ONCE
          BIT      #BIT1,@RKER      ;CHECK SUM EROR?
          BEQ      WRTCHK
          MOV      #BEGIN,$LPERR

```

```

3143 011056 004737 016214
3144 011062 013737 001252 001202
3145 011070 062737 000004 001202
3146 011076 104024
3147 011100 005237 001514
3148 011104 001002
3149 011106 000137 012170
3150 011112 000137 011654
3151
3152 011116 005037 001506
3153
3154 011122 022737 000003 001256
3155 011130 001016
3156
3157 011132 013777 001440 170324
3158 011140 013777 001444 170312
3159 011146 013777 001442 170306
3160
3161 011154 012777 000407 170274
3162
3163 011162 005337 001256
3164
3165 011166 005737 001506
3166 011172 001060
3167
3168 011174 005237 001506
3169
3170
3171
3172
3173
3174
3175
3176 011200 012702 001266
3177
3178 011204 012703 001320
3179 011210 012704 001352
3180
3181
3182 011214 005037 001500
3183
3184 011220 013700 001404
3185 011224 012737 177764 001502
3186 011232 013701 001406
3187 011236 012737 176400 001260
3188 011244 021011
3189 011246 001402
3190 011250 000137 012016
3191 011254 005720
3192 011256 005721
3193 011260 005237 001260
3194 011264 001367
3195
3196 011266 005737 001500
3197
3198 011272 001420
          JSR      PC,GETINF
          MOV      RETRY1,$REG10      ;GET THE RETRY #
          ADD      #4,$REG10          ;SAVE IT FOR TYPEOUT
          ERROR    24      ;CSE
          INC      ERCNT5      ;ALLOW 10 ERRORS AT MOST
          BNE     18
          JMP      EXT7      ;IF MORE, EXIT
          JMP      CSEROR        ;GO, SERVICE CSE
          WRTCHK: CLR      ERCNT2      ;CLR FLAG INDICATING SOFTWARE
          ;COMPARE DONE
          CMP      #3,RETRY3
          BNE     SFTCMP
          ;WRT CHK DONE BEFORE OR
          ;IT'S 1ST TIME?
          ;IF DONE,BRANCH OTHERWISE DO IT
          WCCAGAIN: MOV    WDSKAD,@RKDA      ;WRT CHK FROM THIS DSK=ADRES
          MOV      WRDcnt,@RKWC      ;THIS # FO WORDS
          MOV      WBUSAD,@RKBA      ;WITH THIS BUFFER
          MOV      #407,@RKCS      ;WRT CHK,GO,SSE
          DEC      RETRY3      ;INDICATE WRT CHK DONE
          SFTCMP: TST      ERCNT2      ;SOFTWARE CMPARE DONE ONCE BEFORE?
          BNE     WCREPT          ;IF SFTWARE CMPARE HAS BEEN DONE
          ;ONCE DON'T DO IT AGAIN, OTHERWISE,
          INC      ERCNT2          ;DO IT, INDICATE IT IS DONE.
          ;MORE THAN ONCE BEFORE.
          ;IF THIS IS 1ST TIME THRU &
          ;WRT CHK WAS DONE ONCE BEFORE
          ;DO SOFTWARE COMPARISON OF
          ;THE DATA THAT WAS READ FROM
          ;THE DISK
          MOV      #BUFR,R2          ;INITLZE PTR TO BUFR STORING
          ;ADRES OF BAD DATA
          MOV      #BUFR1,R3        ;STORE EXPTCD DATA STARTING HERE
          MOV      #BUFR2,R4        ;STORE RECVD (BAD) DATA
          ;STARTING HERE
          CLR      INDX3            ;CLR FLAG INDICATING MISCMPRE
          COMPAR: MOV      PBUFO,R0      ;INITLZE PTR TO 'RECVD DATA' BUFR
          MOV      #14,INDX4        ;STORE AND REPORT 12 OR LESS DATA ERRORS
          MOV      PBUF1,R1        ;INITLZE PTR TO 'EXPTCD DATA' BUFR
          MOV      #1400,INADR      ;SET COUNT
          CMPAGAN: CMP      (R0),(R1)    ;CORRECT WORD READ FROM DISK?
          BEQ      18
          JMP      MISCMP          ;BRANCH IF MISCMPRE ERROR
          ;INCRMNT PTRS
          TST      (R0)+
          TST      (R1)+
          INC      INADR          ;TO NXT WORDS
          BNE     CMPAGAN        ;DONE WITH CMPRISON?
          ;IF NOT, COMPARE THE REST
          TST      INDX3          ;WAS THERE A BAD DATA WORD
          ;(EVEN AFTR RETRYING)
          BEQ      WCREPT          ;NO, BRANCH

```

```

3199 011274 012737 010606 001110 REPMSC: MOV #READ,SLPERR
3200 011302 104025 ERROR 25 ;DATA ERROR
3201 011304 005237 001516 INC ERCNT6 ;ALLOW 10 ERRORS AT MOST
3202 011310 001002 BNE 18
3203 011312 000137 012170 JMP EXT7 ;IF MORE, EXIT
3204 011316 005737 001254 18: 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,ORKDA ;WRT CHK WITH THIS DSK-ADRES
3217 011360 013777 001444 170072 MOV WWRDCN,ORKWC ;THIS # OF WORDS
3218 011366 013777 001442 170066 MOV WBUSAD,ORKBA ;THIS BUS ADRES
3219 011374 012777 000407 170054 MOV #407,ORKCS
3220
3221 011402 104421 CON,RDY
3222 011404 012737 011350 001110 ERWCHK: MOV #LUPWCE,SLPERR
3223 011412 032777 040000 170032 BIT #BIT14,ORKDS ;HARD ERROR?(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,ORKER ;WRITE CHECK ERROR?
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 18
3240 011502 000137 012170 JMP EXT7 ;IF MORE, EXIT
3241 011506 000137 012062 18: JMP WCEROR
3242
3243 ;THERE WAS NO WCE, DONE
3244
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
3251 011520 001404 BEQ 18
3252 011522 062737 000002 001424 ADD #2,PRSPAT ;FIND OUT THE NXT PATRN GENRATR
3253 011530 000403 BR 28 ;TO USE FOR GENERATING THE
3254 011532 012737 001414 001424 18: MOV #PAT0,PRSPAT ;BUFR,STORE POINTER TO
;GENRATR ROUTINE IN 'PRSPAT'
;NOTE THER R 4 PAT-GENRATRS

```

```

3255 ;PATO-PAT1-PAT2-PAT3---PATO-
3256 011540 013701 001476 28: MOV INDX2,R1 ;FORM SECTR # TO BE USED NEXT
3257 011544 116101 010426 MOVB SECPTR(R1),R1 ;GET SECTR #
3258 011550 042737 000017 001450 38: BIC #17,ADRES ;MASK SECTR BITS
3259 011556 050137 001450 BIS R1,ADRES ;FORM THE NEW DISK-ADRES
3260 ;FORM THE NXT BLOCK TO BE
3261 ;CHECKED,
3262 011562 005237 001476 INC INDX2 ;DONE ALL 4 BLOCKS(3 SECS
3263 ;EACH) ON THIS TRACK?
3264 011566 022737 000004 001476 CMP #4,INDX2
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 DOSUR1 ;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
3288
3289 011654 CSEROR: ;THIS IS THE ENTRY POINT
;FOR SERVICE ROUTINE FOR CHECK
;SUM EROR, CONTROL IS XFERED
;HERE ONCE CSE OCCURS
3290
3291
3292 011654 017700 167604 MOV ORKDA,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 18 ;WHICH GAVE CSE
3296 011670 162700 000004 SUB #4,R0
3297 011674 005300 18: 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 28 ;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 38 ;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      BNE      88
3314 011722 000137 011116    JMP      WRTCHK
3315 011726 162702 000003    SUB      #3,R2
3316 011732 100372      BPL      78
3317
3318 011734 010100      MOV      R1,R0
3319 011736 012737 177775 001252    MOV      #-3,RETRY1
3320 011744 000403      BR       38
3321
3322 011746 012737 177776 001252 28:  MOV      #-2,RETRY1      ;ALLOW 2 MORE TRIES FOR
                                     ;THE CSE ON SAME DISK-ADRES
3323
3324
3325 011754 010037 001432      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      DEC      R0            ;TO BE USE ON RETRY-
3330 011770 001407      BEQ      58            ;READ
3331 011772 062737 001000 001434    ADD      #1000,BUSADR
3332 012000 062737 000400 001436    ADD      #400,WRDCNT
3333 012006 000767      BR       48
3334
3335 012010 104415      CON,RESET 58:  CON,RESET      ;CLR THE CSE IN RKER
3336 012012 000137 010642    JMP      RDAGAIN      ;GO BACK, READ AGAIN
3337                                     ;RETRY
3338
3339
3340
3341                                     ;MISCMP
3342                                     ;THIS IS THE ENTRY POINT
3343                                     ;FOR SERVICE ROUTINE, FOR
3344                                     ;DATA ERROR (MISCOMPARISON)
3345                                     ;ON SOFTWARE COMPARISON
3346                                     ;OF DATA READ FROM THE DISK BLOCK
3347
3348 012016 104421      MISCMP: CON,RDY      ;WAIT FOR CNTRL RDY FROM
3349                                     ;PREVIOUS WRT CHK
3350
3351 012020 032777 040000 167430    BIT      #BIT14,0RKCS
3352 012026 001401      BEQ      18
3353 012030 104415      CON,RESET      ;CLR WCE IN RKER
3354 012032 010022      MOV      R0,(R2)+     ;BUT STILL DATA EROR ENTER HERE
3355                                     ;STORE MEM ADRES WHERE
3356 012034 012123      MOV      (R1)+,(R3)+ ;DATA EROR OCCURED
                                     ;SAVE EXPCTD DATA
3357
3358 012036 012024      MOV      (R0)+,(R4)+ ;SAVE DATA RECVD (BAD)
3359 012040 005237 001500      INC      INDX3      ;INDICATE MISCMPRISON
3360
3361 012044 005237 001502      INC      INDX4      ;STORE ONLY 12(DEC) ERORS
3362 012050 001402      BEQ      48          ;IF 12 ERORS, GO REPORT THEM
3363
3364 012052 000137 011244      JMP      CMPAGAN     ;GO BACK & CMPARE THE REST
3365
3366 012056 000137 011274      JMP      REPMSC      48:
  
```

```

3367
3368
3369                                     ;THIS THE ENTRY POINT FOR
3370                                     ;EROR SERVICE ON GETTING A
3371                                     ;WRITE CHECK EROR.
3372
3373 012062 005737 001256      WCEORR: TST      RETRY3
3374 012066 003038      BGT      CLRERR      ;DONE 3 TRIES?
3375                                     ;IF NOT SKIP, OTHERWISE REPORT
3376                                     ;W C EROR
3377 012070 012737 000003 001256    MOV      #3,RETRY3    ;SET CONT FOR RETRIES
3378
3379 012076 017700 167362      MOV      0RKDA,R0     ;IF WCE WAS IN THE LAST SECTOR
3380 012102 010002      MOV      R0,R2       ;OF THE BLOCK, NO MORE SECS
3381 012104 042702 177760      BIC      #177760,R2   ;TO CHECK, GO TO 'FINISH'
3382 012110 001002      BNE      48          ;IF IT WASN'T LAST SECTOR OF THE
3383 012112 000137 011512      JMP      FINISH      ;BLOCK, THEN CHECK REMAINING
3384 012116 162702 000003      SUB      #3,R2       ;SECTORS, (STARTING FROM THE
3385 012122 100372      BPL      38          ;SEC AFTER THE ONE GIVING WCE)
3386 012124 010001      MOV      R0,R1       ;SAVE DISK ADRES
3387 012126 163700 001440      SUB      WDSKAD,R0   ;
3388 012132 010137 001440      MOV      R1,WDSKAD   ;GET SAVED DISK ADRES
3389 012136 005200      INC      R0
3390 012140 005300      DEC      R0
3391 012142 001407      BEQ      CLRERR
3392 012144 062737 001000 001442    ADD      #1000,WBUSAD ;FORM THE NEW BUS ADRES
3393 012152 062737 000400 001444    ADD      #400,WRDCN  ;FORM THE NEW WORD COUNT
3394 012160 000767      BR       28
3395 012162 104415      CLRERR: CON,RESET
3396 012164 000137 011132    JMP      WCAGAIN
3397
3398 012170 004737 016772      EXT7:  JSR      PC,ABRT
3399
3400
3401                                     ;*****
3402                                     ;*TEST 10 WRITE, WRITE CHECK ON CYLINDERS 127, 128
3403                                     ;*THIS TEST WRITES 12 UNIQUE PATTERNS (ONE FOR EACH
3404                                     ;*SECTOR) ON CYLINDERS 127 AND 128, THEN WRITE
3405                                     ;*CHECK IS DONE TO SEE IF THEY WERE WRITTEN
3406                                     ;*CORRECTLY. IT SHOULD BE NOTED THAT THERE IS
3407                                     ;*CHANGE IN 'WRITE' CURRENT AT THIS CYLINDER.
3408                                     ;*PATTERNS ARE RELOCATED ON THE CYLINDERS AFTER EACH
3409                                     ;*WRITE/WRITE CHECK CYCLE, THUS THE FIRST TIME
3410                                     ;*PATTERN 'SP1' IS WRITTEN ON SECTOR 0, PATTERN 'SP2' ON
3411                                     ;*SECTOR 2, ETC. AFTER THIS WRITE/WRITE CHECK CYCLE
3412                                     ;*IS OVER PATTERN 'SP2' IS WRITTEN ON SECTOR 0, PATTERN
3413                                     ;*'SP3' ON SECTOR 1 AND SO ON, THIS WRITE/WRITE
3414                                     ;*CHECK CYCLE IS REPEATED 12 TIMES, THUS
3415                                     ;*THE LAST WRITE/WRITE CHECK IS DONE USING
3416                                     ;*PATTERN 'SP12' ON SECTOR 0, PATTERN 'SP1' IS
3417                                     ;*WRITTEN ON SECTOR 1, PATTERN 'SP2' ON SECTOR 2,
3418                                     ;*ETC. IF YOU WANT TO WRITE ANY OTHER PATTERNS
3419                                     ;*FILL IN THE PATTERNS YOU WANT IN LOCATIONS
3420                                     ;*'SP1', 'SP2',...ETC.
3421                                     ;*****
3422 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 012702              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                      WRLO1: CLR    R3
3432 012240 104415                      WRERR: CON,RESET
3433
3434 012242 010077 167216              WRLO:  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    48
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    48
3447 012324 000137 012732              JMP    EXT10
3448
3449 012330 005200                      48:  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    38
3453 012342 012701 012700              MOV    #SP1-2,R1
3454 012346 005721                      TST   (R1)+           ;INCREMENT POINTERS TO NEXT PATTERN
3455 012350 020327 000014              CMP    R3,#14        ;DONE SURFACE 0?
3456 012354 002732                      BLT   WRLO            ;NO
3457 012356 001005                      BNE    28             ;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              28:  CMP    R3,#30      ;DONE WITH WRITING SURFACE 1?
3463 012376 001321                      BNE   WRLO            ;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    WRLO1           ;GO BACK
3470
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   18
3487 012500 004737 016214              JSR    PC,GETINF     ;NO
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 18:  BIT    #BIT0,@PKER   ;WCE?
3492 012522 001425                      BEQ   48
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 ;WCE ON DOING WRITE CHECK, WITH
3496 012544 062737 000004 001202      ADD    #4,$REG10     ;PATTERN STORED IN R1
3497 012552 104027                      ERROR  27             ;DO 3 TIMES IN ALL
3498 012554 005237 001252              INC    RETRY1        ;
3499 012560 001330                      BNE   WCERR          ;
3500 012562 005237 001510              INC   WCERR          ;ALLOW 15 ERRORS ONLY
3501 012566 001461                      BEQ   EXT10          ;IF MORE, EXIT
3502 012570 012737 177775 001252      MOV    #3,RETRY1
3503
3504 012576 005200                      48:  INC    R0
3505 012600 005203                      INC    R3             ;KEEP TRACK OF DISK-ADRES
3506 012602 022701 012730              CMP    #SP12,R1      ;AND COUNT
3507 012606 001002                      BNE   38
3508 012610 012701 012700              MOV    #SP1-2,R1    ;USE PATTERNS IN CYCLIC
3509 012614 005721                      TST   (R1)+           ;FASHION
3510 012616 020327 000014              CMP    R3,#14        ;INCREMENT POINTER TO NEXT PATTERN
3511 012622 002711                      BLT   WCHI            ;DONE SURFACE 0?
3512 012624 001005                      BNE   28             ;NO
3513 012626 010201                      MOV    R2,R1
3514 012630 042700 000017              BIC   #17,R0         ;IF CHANGING HEADS (0-1), SET CORRECT
3515 012634 052700 000020              BIS   #20,R0         ;ADRES BITS
3516
3517 012640 020327 000030              28:  CMP    R3,#30      ;DONE WRITE CHECKING SURFACE 1?
3518 012644 001300                      BNE   WCHI            ;NO, GO BACK
3519
3520 012646 032700 007700              WCL0:  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
    
```

3535  
3536 012702 177777  
3537 012704 052525  
3538 012706 111111  
3539 012710 010421  
3540 012712 102041  
3541 012714 010101  
3542 012716 040201  
3543 012720 000401  
3544 012722 031463  
3545 012724 070707  
3546 012726 007417  
3547 012730 041020  
3548  
3549 012732 004737 016772  
3550  
3551  
3552  
3553  
3554  
3555  
3556  
3557  
3558  
3559  
3560  
3561  
3562  
3563  
3564  
3565  
3566

SP1: ,WORD 177777 ;IF YOU WANT TO WRITE ANY  
SP2: ,WORD 052525 ;OTHER PATTERNS, CHANGE THESE  
SP3: ,WORD 111111 ;12 LOCATIONS TO ANY PATTERN  
SP4: ,WORD 010421 ;YOU WANT.  
SP5: ,WORD 102041  
SP6: ,WORD 010101  
SP7: ,WORD 040201  
SP8: ,WORD 000401  
SP9: ,WORD 031463  
SP10: ,WORD 070707  
SP11: ,WORD 007417  
SP12: ,WORD 041020  
EXT10: JSR PC,ABRT

3567 012736 000004  
3568 012740 032777 000100 166172  
3569 012746 001002  
3570 012750 000137 014106  
3571 012754 104415  
3572 012756 104416  
3573  
3574 012760 012737 177771 001252  
3575  
3576  
3577 012766 104401 012774  
3578 012772 000424  
3579  
3580 013044  
3581 013044 104401 013052  
3582 013050 000421  
3583  
3584 013114  
3585 013114 104401 013122  
3586 013120 000421  
3587  
3588 013164  
3589  
3590 013164 012737 001542 001260

```
*****  
;TEST 11 SEEK FUNCTION TIMER  
;SEEK TIMER  
;IN THIS PART OF THE PROGRAM SEEKS ARE TIMED BETWEEN A PARTICULAR SET  
;OF CYLINDERS, BOTH IN THE FORWARD DIRECTION (0-312) AND REVERSE(312-0).  
;*****CAUTION*****  
;IT SHOULD BE NOTED THAT THE SECTOR COUNTER (IN RKDS) IS USED AS THE REAL  
;TIME CLOCK TO DO THE SEEK TIMING, FOR THE TIMES TO BE RELIABLE, THE  
;SECTOR COUNTER SHOULD BE ACCURATE WITHIN THE SPECIFICATIONS OF THE DISK  
;SPEED: 1500 RPM = 40 MILI SECS/REV =3.33 MILI SECS/SECTOR.  
;VARIATION: +/-30 RPM  
;*****  
TST11: SCOPE  
BIT #SW6,#SWR ;INHIBIT TIMER?  
BNE #+6  
JMP PLTGRPH  
CON,RESET  
DRV,RESET  
MOV #-7,RETRY1 ;COUNT FOR 7 DIFRNT SEEK TIMES  
;TO BE RECORDED  
TYPE ,658 ;;TYPE ASCIZ STRING  
BR 648 ;;GET OVER THE ASCIZ  
;1658: ,ASCIZ <15><12>/SEEK TIME SCALE FACTOR=0,01 MILI SECS/  
648:  
TYPE ,678 ;;TYPE ASCIZ STRING  
BR 668 ;;GET OVER THE ASCIZ  
;1678: ,ASCIZ <15><12><12>/ # OF SEEK # OF SEEK/  
668:  
TYPE ,698 ;;TYPE ASCIZ STRING  
BR 688 ;;GET OVER THE ASCIZ  
;1698: ,ASCIZ <15><12>/ SEEKS TIME SECS TIME/<15><12>  
688:  
MOV #SIAD,INADR ;INITLZE PTR TO INNER ADRES
```

3591 013172 012737 001524 001262  
3592  
3593 013200 017777 166056 166256  
3594  
3595 013206 053777 001230 166250  
3596 013214 012777 000011 166234  
3597 013222 104421  
3598 013224 104422  
3599  
3600 013226 005037 001474  
3601 013232 012704 027026  
3602 013236 012705 027426  
3603 013242 012737 177634 001476  
3604  
3605  
3606 013250 013702 001464  
3607 013254 005737 001474  
3608 013260 001005  
3609  
3610 013262 017712 165772  
3611 013266 053712 001230  
3612 013272 000404  
3613  
3614 013274 017712 165762  
3615 013300 053712 001230  
3616 013304 004737 014776  
3617  
3618  
3619  
3620 013310 005737 001474  
3621 013314 001004  
3622 013316 010324  
3623 013320 005237 001474  
3624 013324 000751  
3625  
3626 013326 010325  
3627 013330 005037 001474  
3628  
3629 013334 005237 001476  
3630 013340 001343  
3631  
3632  
3633  
3634  
3635  
3636  
3637  
3638  
3639  
3640  
3641  
3642  
3643  
3644  
3645  
3646

```
MOV #SOAD,OUTADR ;INITLZE PTR TO OUTER ADRES  
REPTIM: MOV @OUTADR,@RKDA ;POSITION HEADS TO OUTER CYLINDER  
;BEFORE STARTING TO TIME  
BIS DRIVAD,@RKDA ;SET DRIVE # BITS  
MOV #11,@RKCS ;SEEK, GO  
CON,RDY ;WAIT FOR CNTRL RDY  
TST,RWS ;WAIT FOR R/W/S RDY  
CLR INDX1 ;INDX1 = 0, GOING IN; OTHERWISE OUT  
MOV #BUF10,R4 ;STORE FRWD SEEK TIMES IN THIS BUFR  
MOV #BUF11,R5 ;STORE REVRSE " "  
MOV #-144,INDX2 ;SET COUNT FOR # OF SEEKS  
BEGSK: MOV RKDA,R2  
TST INDX1 ;GOING FRWD OR REVRSE?  
BNE 18 ;REVRSE, BRANCH  
MOV @INADR,@R2 ;FRWD, SET INNER CYL ADRES  
BIS DRIVAD,@R2 ;SET DRIVE # BITS  
BR 28  
18: MOV @OUTADR,@R2 ;SET OUTER CYL ADRES  
BIS DRIVAD,@R2 ;SET DRIVE # BITS  
28: JSR PC,@TIMSEK ;GO,-TIME THE SEEK FROM CYLINDER  
;0 TO THE ABOVE CYL. RETURN WITH  
;R3 CONTAINING THE TIME (MS, SCALE  
;FACTOR= 0,01) REQUIRED FOR THE SEEK.  
TST INDX1  
BNE 38  
MOV R3,(R4)+ ;STORE TIME TAKEN FOR FRWD SEEK  
INC INDX1 ;SET FLAG FOR DOING REVRSE SEEK  
BR BEGSK ;GO DO IT  
38: MOV R3,(R5)+ ;STORE TIME TAKEN FOR REVRSE SEEK  
CLR INDX1 ;CLR FLG FOR DOING FRWD SEEK  
INC INDX2 ;RECORDED 144 SEEK TIMES  
BNE BEGSK ;IF NOT, GO BAK
```

```
;AT THIS POINT 100 SEEKS HAVE BEEK PERFORMED BETWEEN TWO  
;CYLINDERS (FORWARD & REVERSE DIRECTION), FORWARD SEEK  
;TIMES ARE STORED IN TABLE STARTING AT "BUF10" REVERSE  
;STARTING AT "BUF11". THE FOLLOWING CODE FINDS OUT THE  
;NUMBERS OF TIMES A PARTICULAR "SEEK TIME" WAS OBTAINED,  
;EXAMPLE: OUT OF 100 TIMES THE SEEK WAS DONE BETWEEN  
;CYLINDER 0 & LAST.  
;70 TIMES IT TOOK 95 MILI SECS  
;20 TIMES IT TOOK 85 MILI SECS  
;10 TIMES IT TOOK 100 MILI SECS  
;THIS INDICATES HOW CONSISTENT WAS THE SEEKING TIME,  
;NOTE THAT ONLY 5 DIFFERENT "SEEK TIMES" WILL BE TYPED  
;OUT.  
;SORTING ROUTINE
```

```

3647
3648 013342 012705 177776 SORT: MOV #2,R5 ;COUNT FOR FRWRD, REVRSE
3649 013346 012737 027026 001162 MOV #BUF10,R10 ;INTLZE PTR TO 'SEEK TIME'
3650 013354 012737 027030 001164 MOV #BUF10+2,R10
3651 013362 012737 026526 001166 MOV #BUF10,R10
3652 013370 012737 026606 001170 MOV #BUF10,R10 ;INTLZE PTR TO '# OF TIMES'
3653
3654 013376 013700 001162 18: 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 SORT
3657 013412 005003 CLR R3
3658
3659 013414 021011 28: CMP (R0),(R1) ;SORT THE ITEMS & PUT THEM
3660 013416 003404 BLE R3 ;IN DESCENDING ORDER
3661 013420 011004 MOV (R0),R4 ;LARGER ITEMS AT TOP OF LIST,
3662 013422 011110 MOV (R1),(R0) ;SMALLER AT THE BOTTOM
3663 013424 010411 MOV R4,(R1)
3664 013426 005203 INC R3
3665 013430 005720 38: TST (R0)+
3666 013432 005721 TST (R1)+
3667 013434 005202 INC R2
3668 013436 001366 BNE 28
3669 013440 005703 TST R3 ;SORTED ALL ITEMS?
3670 013442 001355 BNE 18 ;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 48: CMP (R0)+,(R1)
3686 013510 001411 BEQ 58 ;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 68
3694 013532 000765 BR 48
3695
3696 013534 005212 58: INC (R2) ;INCRMNT '# OF TIMES'
3697 013536 005203 INC R3 ;ALL DONE?
3698 013540 001362 BNE 48 ;IF NOT, GO BAK
3699
3700 013542 005205 68: INC R5 ;SORTED BOTH FRWRD, REVRSE
3701 013544 001415 BEQ GOTYPE ;'SEEK TIMES', IF YES GO TYPE
3702
    
```

```

3703 013546 012737 027426 001162 MOV #BUF11,$REG0 ;IF NOT, INITLZE PTR TO 'SEEK TIME'
3704 013554 012737 027430 001164 MOV #BUF11+2,$REG1
3705 013562 012737 026666 001166 MOV #BUF11,$REG2 ;SAVE 'SEEK TIME'
3706 013570 012737 026746 001170 MOV #BUF11,$REG3 ;SAVE '# OF TIMES' HERE
3707
3708 013576 000677 BR 18 ;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 GOTYPE: TYPE
3713 013602 001213 SCRLF
3714 013604 104401 013612 TYPE ,65$ ;TYPE ASCIZ STRING
3715 013610 000403 BR 64$ ;GET OVER THE ASCIZ
3716 ;65$: ,ASCIZ /CYLS;/
3717 64$:
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$: ,ASCIZ /=/
3730 66$:
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$: ,ASCIZ <15><12>/ FRWRD/
3742 68$:
3743 013714 104401 002101 TYPE ,BLNK$9
3744 013720 104401 013726 TYPE ,71$ ;TYPE ASCIZ STRING
3745 013724 000404 BR 70$ ;GET OVER THE ASCIZ
3746 ;71$: ,ASCIZ /REVRSE/
3747 70$:
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 18: TYPE
3755 013744 001213 SCRLF
3756 013746 016046 026606 MOV BUFR5(R0),=(SP) ;GET '# OF SEEKS', IF NONE (0)
3757 013752 001424 BEQ 38 ;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(RO),=(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 28:  MOV   BUFR7(RO),=(SP) ;GET '% OF SEEKS', IF NONE (0)
3766 013774 001416          BEQ   48 ;SKIP TYPING (REVRSE SEEK)
3767 013776 005705          TST   R5
3768 014000 001402          BEQ   68
3769 014002 104401 002075  TYPE   ,BLNK13
3770 014006 104405          TYPDS ;TYPE OUT IN DECIMAL
3771 014010 104401          TYPE
3772 014012 002110          BLNKS2
3773 014014 016046 026666  MOV   BUFR6(RO),=(SP) ;GET 'SEEK TIME' & TYPE IT
3774 014020 104405          TYPDS ;OUT IN DECIMAL
3775 014022 000406          BR    58
3776
3777 014024 005726          38:  TST   (SP)+ ;POP STACK
3778 014026 005205          INC   R5
3779 014030 000757          BR    28
3780
3781 014032 005726          48:  TST   (SP)+ ;POP STACK
3782 014034 005705          TST   R5
3783 014036 001004          BNE   TIMDON
3784
3785 014040 005720          58:  TST   (RO)+ ;INCREMENT PTR TO TABLES
3786 014042 020027 000012  CMP   RO,#12 ;ALL DONE?
3787 014046 001335          BNE   18 ;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   #=313,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 18:  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   #1777,@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   18
3832
3833 ;PLOT A GRAPH USING 'SEEK TIMES' RECORDED BEFORE
3834
3835 014222          PLOT:
3836 014222 104401 014230          TYPE   ,658 ;TYPE ASCIZ STRING
3837 014226 000422          BR    648 ;GET OVER THE ASCIZ
3838 ;658: .ASCIZ <15><12><12><12>/X AXIS = SEEK TIME = MILI SECS/
3839 648:
3840 014274 104401 014302          TYPE   ,678 ;TYPE ASCIZ STRING
3841 014300 000423          BR    668 ;GET OVER THE ASCIZ
3842 ;678: .ASCIZ <15><12>/Y AXIS = CYLINDER SEEKED FROM 0/<15><12><12>
3843 668:
3844
3845 014350 104401          TYPE
3846 014352 002103          BLNKS7
3847 014354 005000          CLR   RO
3848 014356 010046          MOV   RO,=(SP) ;TYPE OUT THE TIME UNITS
3849 014360 104424          TYPDS ;(MILI SECS) FOR THE X=AXIS
3850 014362 005700          TST   RO ;LIKE THIS:
3851 014364 001411          BEQ   28 ;0 20 30 40.....
3852 014366 022700 000144  CMP   #144,RO
3853 014372 003010          BGT   48
3854 014374 022700 000170  CMP   #170,RO
3855 014400 002412          BLT   58
3856 014402 104401          TYPE
3857 014404 002110          BLNKS2
3858 014406 000404          BR    38
3859 014410 104401          28:  TYPE
3860 014412 002111          BLNKS1
3861 014414 104401          48:  TYPE
3862 014416 002107          BLNKS3
3863 014420 062700 000012  38:  ADD   #12,RO
3864 014424 000754          BR    18
3865
3866 014426 104401          58:  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                                18:
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    16
3879                                ;IF SW 4 IS SET THEN TYPE THE COMPLETE GRAPH, IF NOT TYPE THE SMALL GRAPH.
3880
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 18:  BIT     $SW5,$SWR      ;SKIP REST OF GRAPH?
3890 014502 001445                                BEQ    58              ;YES
3891 014504 104401                                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                                BR     16
3918 014602 022700 000312 4$:  CMP     #312,R0       ;PLOTTED ALL CYLS?
3919 014606 001403                                BEQ    5$
3920                                ADD     #2,R0
3921 014614 000727                                BR     16
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                                TYPE
3979 014744 104401                                BLNKS1
3980 014746 002111                                BR
3981 014750 000772                                PLTPT
3982 014752 062703 000144 7$:  ADD     #144,R3
3983 014756 002402                                BLT    8$
    
```

3983 014760 104401  
3984 014762 002111  
3985  
3986 014764  
3987 014764 104401 014772  
3988 014770 000401  
3989  
3990 014774  
3991 014774 000207  
3992  
3993  
3994  
3995  
3996  
3997  
3998  
3999  
4000  
4001  
4002  
4003 014776 010246  
4004 015000 005003  
4005 015002 013701 001452  
4006 015006 011102  
4007 015010 032702 000400  
4008 015014 001774  
4009  
4010 015016 032702 000010  
4011 015022 001771  
4012  
4013  
4014 015024 011102  
4015 015026 032702 000400  
4016 015032 001774  
4017 015034 021102  
4018 015036 001372  
4019 015040 032702 000017  
4020 015044 001367  
4021  
4022  
4023 015046 012777 000011 164402  
4024  
4025 015054 104421  
4026  
4027 015056 011102  
4028 015060 032702 000400  
4029 015064 001774  
4030 015066 020211  
4031 015070 001372  
4032 015072 032702 000100  
4033 015076 001025  
4034 015100 032702 000017  
4035 015104 001764  
4036  
4037 015106 011102  
4038 015110 032702 000400

TYPE  
BLNKS1  
08: 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  
TIMSEK: MOV R2,=(SP) ;R3 WILL COUNT REVOLUTIONS OF  
CLR R3 ;DISK (FROM INDEX MARK TO INDEX MARK)  
MOV RKDS,R1 ;40 MILI SECS FOR EACH REV  
18: 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  
28: 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  
;ISSUE A SEEK, START TIMING  
;THE SEC COUNTER  
;WAIT FOR CNTRL RDY  
38: 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  
48: MOV @R1,R2  
BIT #400,R2 ;WAIT FOR SOK

4039 015114 001774  
4040 015116 020211  
4041 015120 001372  
4042 015122 032702 000100  
4043 015126 001005  
4044 015130 032702 000017  
4045 015134 001364  
4046  
4047 015136 005203  
4048 015140 000746  
4049  
4050 015142 032702 000017  
4051 015146 001001  
4052 015150 005203  
4053  
4054  
4055 015152  
4056 015152 012746 000014  
4057 015156 010346  
4058 015160 004737 020500  
4059 015164 012616  
4060 015166 012603  
4061 015170 042702 177760  
4062 015174 060203  
4063  
4064 015176 012746 000512  
4065 015202 010346  
4066 015204 004737 020500  
4067 015210 012616  
4068 015212 012603  
4069 015214 062703 000245  
4070  
4071  
4072  
4073  
4074  
4075  
4076 015220 012602  
4077 015222 000207  
4078  
4079  
4080  
4081  
4082  
4083  
4084  
4085 015224 000004  
4086 015226 105237 001223  
4087 015232 123737 001223 001224  
4088 015240 001402  
4089 015242 000137 003760  
4090  
4091  
4092  
4093  
4094

BEQ 4\$  
CMP R2,@R1  
BNE 4\$  
BIT #100,R2 ;R/W/S RDY SET, SEEK DONE?  
BNE 5\$ ;YES, BRANCH  
BIT #17,R2 ;IF NOT KEEP TRACK OF SEC  
BNE 4\$ ;COUNTER, INCREMENT R3 AT  
;EVERY INDEX MARK,EVERY  
;40 MILI SECS  
;GO BAK, KEEP TIME  
58: BIT #17,R2 ;CHECK, IS IT INDEX MARK -SEC 0  
BNE SKDON ;IF NOT, SKIP  
INC R3 ;IF YES, INCREMENT COUNT  
;SEEK DONE, SAVE RKDS-SEC COUNTER.  
SKDON: MOV #14,=(SP) ;PUT THE MULTIPLIER ON THE STACK  
MOV R3,=(SP) ;PUT THE MULTIPLICAND ON THE STACK  
JSR PC,@\$MULT ;CALL THE MULTIPLY ROUTINE  
MOV (SP)+,(SP) ;DISREGARD THE MSB'S  
MOV (SP)+,R3 ;GET THE LSB'S OF THE PRODUCT  
BIC #177760,R2 ;SEEK, TOTAL TIME=(IN DECIMAL)  
ADD R2,R3 ;((R3)X12+SEC COUNTER)X330X0.01  
;NOTE THERE IS A SCALE FACTOR  
MOV #512,=(SP) ;PUT THE MULTIPLIER ON THE STACK  
MOV R3,=(SP) ;PUT THE MULTIPLICAND ON THE STACK  
JSR PC,@\$MULT ;CALL THE MULTIPLY ROUTINE  
MOV (SP)+,(SP) ;DISREGARD THE MSB'S  
MOV (SP)+,R3 ;GET THE LSB'S OF THE PRODUCT  
ADD #245,R3 ;ASSUMPTION THAT EACH SECTOR  
;TAKES 3,3 MILI SECS, IF THE  
;DISK SPEED IS VERY MUCH DIFRNT  
;FROM THE SPEC SPEED OF  
;1500 RPM (40 MS/REV), THEN  
;SEC COUNTER WOULD NOT BE AN  
;ACCURATE TIME CLOCK.  
MOV (SP)+,R2 ;POP R2 BAK  
RTS PC ;RETURN  
;\*\*\*\*\*  
;TEST 12 END OF PROGRAM  
;THIS IS NOT A TEST BUT IS JUST A LINKAGE  
;PROVIDED TO TEST ALL THE DRIVES,  
;\*\*\*\*\*  
TST12: SCOPE  
INCB DRVNON  
BTEOP: CMPB DRVNON,DRIVS  
BEQ +6  
JMP NXTDRV  
;BTTL END OF PASS ROUTINE  
;\*\*\*\*\*  
;INCREMENT THE PASS NUMBER (PASS)

```

4095 ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
4096 ;*TYPE "END PASS #XXXX" (WHERE XXXX 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 ;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 $PASS FOR TYPEOUT
4113 015312 104405 TYPE ;GO TYPE--DECIMAL ASCII WITH SIGN
4114 015314 104401 015344 TYPDS ;$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 377 000 $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

```

```

4130 ;COMMON SUBROUTINES AND HANDLERS
4131
4132
4133
4134 ;SBTTL ESR15
4135 ;ESR15
4136 ;THIS ROUTINE IS USED TO TYPE OUT ERROR DATA FOR ITEM 15
4137 ;OF THE ERROR TABLE. AT THE TIME OF ENTRY INTO THIS
4138 ;ROUTINE R5 CONTAINS THE DISK ADDRESS FROM WHICH THE 12
4139 ;HEADERS WERE READ, THE SECTOR #'S WHICH GAVE BAD HEADERS HAVE
4140 ;BEEN STORED STARTING AT "BUF1", THE CORRESPONDING BAD HEADERS
4141 ;HAVE BEEN STORED STARTING AT "BUF1R".
4142
4143 ;THE PRINTOUT LOOKS LIKE:
4144 ;SEC# HDR RECVD
4145 ;AA BBBBBB AA=BAD SEC # BBBBBB=BAD HEADER
4146 ;
4147 ;EXPCTD HDR=XXXXXX TRY# = Y
4148
4149 015364 ESR15:
4150 015364 010146 MOV R1,-(SP) ;PUSH R1 ON STACK
4151 015366 010246 MOV R2,-(SP) ;PUSH R2 ON STACK
4152 015370 012701 001266 MOV #BUF1,R1 ;SEC #'S STORED HERE PREVIOUSLY
4153 015374 012702 001320 MOV #BUF1R,R2 ;BAD HDRS STORED HERE PRVSLY
4154 015400 012146 16: MOV (R1)+,-(SP)
4155 015402 104403 TYPOS ;GO TYPE OUT BAD SEC # (OCTAL)
4156 015404 002 ;ONLY 2 DIGITS
4157 015405 000 ;SUPRES LOG 0'S
4158 015406 104401 TYPE ;TYPE 3 BLNKS
4159 015410 002107 BLNKS3
4160 015412 012246 MOV (R2)+,-(SP) ;GO TYPE OUT BAD HEADER
4161 015414 104402 TYPOC
4162 015416 104401 TYPE
4163 015420 002106 BLNKS4
4164 015422 104401 TYPE
4165 015424 001213 $CRLF
4166 015426 022711 177777 CMP #177777,(R1) ;ALL BAD SEC #'S TYPD OUT?
4167 015432 001362 BNE 16 ;IF NOT GO BAK
4168
4169 015434 104401 TYPE
4170 015436 001716 MSG6
4171 015440 010546 MOV R5,-(SP) ;TYPE OUT EXPCTD HEADER FOR
4172 015442 042716 160037 BIC #160037,(SP)
4173 015446 104402 TYPOC ;THAT CYLINDER
4174
4175 015450 012602 MOV (SP)+,R2 ;POP STACK INTO R2
4176 015452 012601 MOV (SP)+,R1 ;POP STACK INTO R1
4177 015454 000207 RTS PC
4178
4179
4180 ;SBTTL ESR13
4181 ;ESR13
4182 ;THIS ROUTINE IS USED WITH "ERROR 13" TO TYPEOUT OUT ERROR
4183 ;DATA. THE SECTOR #'S WHICH GAVE BAD HEADERS HAVE BEEN STORED
4184 ;STARTING AT "BUF1", THE CORRESPONDING BAD HEADERS HAVE
4185 ;BEEN STORED STARTING AT "BUF1R". R5 CONTAINS THE EXPECTED

```

4186  
4187  
4188  
4189  
4190  
4191  
4192  
4193 015456 004737 015364  
4194 015462 104401 015470  
4195 015466 000404  
4196  
4197 015500  
4198 015500 005046  
4199 015502 032705 000020  
4200 015506 001401  
4201 015510 005216  
4202 015512 104402  
4203  
4204 015514 104401 002053  
4205 015520 103746 001254  
4206 015524 005216  
4207 015526 104402  
4208 015530 000207  
4209  
4210  
4211  
4212  
4213  
4214  
4215  
4216  
4217  
4218  
4219 015532 004737 015456  
4220 015536 004737 016320  
4221  
4222 015542 104401 015550  
4223 015546 000404  
4224  
4225 015560  
4226 015560 103746 001162  
4227 015564 104403  
4228 015566 003  
4229 015567 000  
4230 015570 104401 015576  
4231 015574 000404  
  
4232  
4233 015606  
4234 015606 103746 001164  
4235 015612 104403  
4236 015614 003  
4237 015615 000  
4238 015616 000207  
4239  
4240  
4241

;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  
  
ESR13: JSR PC,ESR15  
TYPE ,65# ;;TYPE ASCIZ STRING  
BR 64# ;;GET OVER THE ASCIZ  
;65# : ,ASCIZ / SUR=  
64# : CLR -(SP)  
BIT #20,R5 ;SUR 0 OR 11?  
BEO 1#  
INC (SP)  
1# : TYPOC  
  
TYPE ,MSG13  
MOV RETRY2,-(SP)  
INC (SP)  
TYPOC  
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.  
  
ESR20: JSR PC,ESR13 ;GO TYPE OUT SEC #'S, BAD HDRS  
JSR PC,ERR2 ;GET CYL #'S BETWN WHICH SEEK  
; WAS TRIED  
TYPE ,65# ;;TYPE ASCIZ STRING  
BR 64# ;;GET OVER THE ASCIZ  
;65# : ,ASCIZ / CYLA=  
64# : MOV \$REG0,-(SP) ;GO TYPE CYL # FROM WHERE  
TYPOS ;SEEK BEGAN  
;BYTE 3 ;TYPE 3 DIGITS  
;BYTE 0 ;SUPRES LDG 0'S  
TYPE ,67# ;;TYPE ASCIZ STRING  
BR 66# ;;GET OVER THE ASCIZ  
  
;67# : ,ASCIZ / CYLB=  
66# : MOV \$REG1,-(SP) ;TYPE CYL # TO WHICH SEEK  
TYPOS ;WAS DONE  
;BYTE 3 ;TYPE 3 DIGITS  
;BYTE 0 ;SUPRES LDG 0'S  
RTS PC ;RETURN  
  
;SBTTL ESR25

4242 015620 010205  
4243  
4244 015622 012702 001266  
4245  
4246 015626 012703 001320  
4247 015632 012704 001352  
4248  
4249 015636 032777 020000 163274 1#  
4250 015644 001076  
4251 015646 104401  
4252 015650 001213  
4253  
4254 015652 163712 001404  
4255 015656 006212  
4256 015660 011246  
4257  
4258  
4259  
4260 015662 104403  
4261 015664 004  
4262 015665 000  
4263 015666 104401  
4264 015670 002107  
4265  
4266 015672 012346  
4267 015674 104402  
4268 015676 104401  
4269 015700 002110  
4270 015702 012446  
4271 015704 104402  
4272 015706 104401  
4273 015710 002110  
4274  
4275 015712 012700 000400  
4276 015716 021200  
4277 015720 002405  
4278 015722 062700 000400  
4279 015726 022700 002400  
4280 015732 001371  
4281  
4282 015734 000300  
4283 015736 005300  
4284 015740 063700 001450  
4285  
4286 015744 010037 001170  
4287  
4288 015750 004737 016220  
4289  
4290  
4291 015754 013746 001174  
4292 015760 104403  
4293 015762 003  
4294 015763 000  
4295 015764 104401  
4296 015766 002107  
4297

ESR25: MOV R2,R5 ;SAVE ADRES OF TERMINATOR  
MOV #BUFR,R2 ;INITLZE PTR TO TABLE STORING  
;ADRES OF BAD DATA  
MOV #BUFR1,R3 ;INITLZE PTR TO 'EXPCTD' DATA  
MOV #BUFR2,R4 ;INITLZE PTR TO 'RECVD' DATA  
  
1# : BIT #SW13,0SWR ;INHIBIT TYPE OUT?  
BNE 4# ;YES, EXIT  
TYPE ;TYPE CR,LF  
\$CRLF  
  
SUB PBUF0,(R2) ;GET WORD # IN BUFR (0,1,2,...)  
ASR (R2)  
MOV (R2),-(SP) ;WHICH WAS BAD, NOTE YOU  
;CAN HAVE THE ACTUAL MEMORY  
;ADRES BY ADDING 'IOBUF0'  
;TO THIS  
;GO TYPE WORD # THAT WAS BAD  
  
TYPOS  
;BYTE 4  
;BYTE 0  
TYPE  
BLNKS3 ;2 BLANKS  
  
MOV (R3)+,-(SP) ;GET EXPCTD DATA  
TYPOC ;GO TYPE IT  
TYPE  
BLNKS2  
MOV (R4)+,-(SP) ;GET RECVD DATA (BAD)  
TYPOC ;GO TYPE IT  
TYPE  
BLNKS2  
  
2# : MOV #400,R0 ;GET THE DISK ADRES FROM  
CMP (R2),R0 ;WHICH THIS (BAD) DATA WAS  
BLT 3# ;READ  
ADD #400,R0  
CMP #2400,R0  
BNE 2#  
  
3# : SWAB R0  
DEC R0  
ADD ADRES,R0 ;R0 CONTAINS THE DISK  
;ADRES FROM WHICH THE (BAD)  
;DATA WAS READ  
MOV R0,\$REG3  
JSR PC,BRKDA ;GO BREAK ABOVE DISK ADRES  
;INTO CYL#, SUR#, SEC#  
  
MOV \$REG5,-(SP) ;GET THE CYL#  
TYPOS ;TYPE IT  
;BYTE 3 ;ONLY 3 DIGITS  
;BYTE 0 ;NO LEADING 0'S  
TYPE  
BLNKS3



4298 015770 013746 001176  
4299 015774 104403  
4300 015776 001  
4301 015777 000  
4302  
4303 016000 104401  
4304 016002 002106  
4305  
4306 016004 013746 001200  
4307 016010 104403  
4308 016012 002  
4309 016013 000  
4310  
4311 016014 005722  
4312 016016 020205  
4313  
4314 016020 001306  
4315 016022 104401  
4316 016024 002053  
4317 016026 013746 001254  
4318 016032 062716 000003  
4319 016036 104403  
4320 016040 001  
4321 016041 000  
4322  
4323 016042 000207  
4324  
4325  
4326  
4327  
4328  
4329  
4330  
4331  
4332  
4333  
4334  
4335  
4336  
4337  
4338  
4339  
4340 016044 032777 020000 163066  
4341 016052 001012  
4342 016054 011637 001116  
4343 016060 162737 000002 001116  
  
4344 016066 117637 000000 001114  
4345 016074 004737 017446  
4346  
4347 016100 062716 000002  
4348 016104 000002  
4349  
4350  
4351  
4352  
4353

```
MOV $REG6,=(SP) ;GET SUR #
TYP0S ;TYPE
, BYTE 1 ;1 DIGIT ONLY
, BYTE 0
TYPE
BLNK54
MOV $REG7,=(SP) ;GET SEC#
TYP0S ;TYPE
, BYTE 2 ;2 DIGITS
, BYTE 0
TST (R2)+ ;INCREMNT PTR
CMP R2,R5 ;TYPED OUT ALL BAD DATA
;INFO?
;IF NOT LUP BAK
BNE 16
TYPE
MSG13 ;" TRY #!"
MOV RETRY2,-(SP) ;GET RETRY COUNT
ADD #3,(SP) ;FORM THE RETRY NO.
TYP0S ;TYPE IT OUT
, BYTE 1
, BYTE 0
4$! RTS PC ;IF YES, RETURN
;MESSAGE HANDLER
;THE MESSAGE HANDLER IS USED FOR TYPING OUT MESSAGES & DATA
;RELATED TO THE MESSAGE, IF SW13 IS SET, THE TYPEOUT IS
;INHIBITED, THE CALL IS:
; MSGC ,XX
;XXX IS THE MESSAGE NUMBER & PROVIDES AN INDEX TO THE
;'ERROR ITEMS TABLE' WHERE THAT MESSAGE ITEM
;IS LOCATED,
;THE MESSAGE ITEM CONTAINS:
; MS: POINTER TO THE ASCII MESSAGE
; DH: POINTER TO THE DATA HEADER
; DT: POINTER TO THE DATA
; 0 TERMINATOR
;IF 'DT' IS 0 THE DATA IS TO BE PRINTED USING THE SUBROUTINE
;INDICATED IN PLACE OF THE TERMINATOR
MSGC: BIT #SW13,$SWR ;INHIBIT TYPEOUT?
BNE 16 ;IF YES, EXIT
MOV $ERRPC ;GET ADRES OF "MESSAGE" CALL
SUB #2,$ERRPC ;STORE IT
MOV $ITEMB ;GET MESSAGE # (INDEX TO ITEM TABLE)
JSR PC,$ERRTYP ;GO TO "ERRTYP" & TYPE OUT
;INFO
;ADJUST RETURN ADDRESS
RTI ;EXIT
;THIS ROUTINE IS USED FOR TYPING OUT ASCII MESSAGES, BEFORE
;THE MESSAGE IS TYPED SW13 IS CHECKED & IF SET THE
;TYPEOUT IS INHIBITED & AN EXIT IS MADE,
;THE CALL FOR THIS ROUTINE IS "TYPMSG", AN ENCODED
```

4354  
4355  
4356  
4357  
4358 016106 032777 020000 163024  
4359 016114 001005  
4360 016116 017637 000000 016126  
4361 016124 104401  
4362 016126 000000  
4363 016130 062716 000002  
4364  
4365 016134 000002  
4366  
4367  
4368  
4369  
4370  
4371  
4372  
4373 016136 017746 163322  
4374 016142 042716 160037  
4375 016146 006316  
4376 016150 006316  
4377 016152 006316  
4378 016154 000316  
4379 016156 112637 001172  
4380  
4381  
4382  
4383  
4384  
4385  
4386  
4387  
4388 016162 017737 163270 001162  
4389 016170 017737 163260 001164  
4390 016176 017737 163250 001166  
4391 016204 017737 163254 001170  
4392 016212 000207  
4393  
4394  
4395  
4396  
4397  
4398  
4399  
4400 016214 004737 016162  
4401 016220 010046  
4402 016222 010146  
4403 016224 010246  
4404 016226 012700 001202  
4405 016232 013701 001170  
4406 016236 010102  
4407 016240 042702 177760  
4408 016244 010240  
4409 016246 006201

```
;TRAP INSTRUCTION,
;THE POINTER TO THE ASCII MESSAGE TO BE TYPED IS LOCATED IN THE
;WORD FOLLOWING THE "TYPMSG" CALL.
TY,MSG: BIT #SW13,$SWR ;INHIBIT TYPEOUT?
BNE 26 ;YES, EXIT
MOV @($P),1$ ;GET POINTER TO ASCII MESSAGE
TYPE ;GO TYPE ASCII STRING
1$: 0
2$: ADD #2,($P) ;ADJUST RETURN ADRES, SKIP OVER
;POINTER ON RETURN
RTI ;EXIT
;GT5RG
;THIS ROUTINE EXTRACTS THE CYLINDER # FROM RKDA AND STORES IT
;IN $REG4, THEN TRANSFERS RKCS, ER, DS, DA TO $REG0, $REG1, $REG2, $REG3
GT5RG: MOV @RKDA,-(SP) ;PUSH RKDA ONTO STACK
BIC #160037,($P) ;MASK OUT NON-CYLINDER BITS
ASL ($P) ;SHIFT 8 BITS OF CYL ADRES TO LO BYTE
ASL ($P)
ASL ($P)
SWAB ($P)
MOVB ($P)+,$REG4 ;UP STACK
;GT4RG
;THIS ROUTINE TRANSFERS THE CONTENTS OF RKCS, RKER, RKDS
;RKDA TO $REG0, $REG1, $REG2, $REG3 RESPECTIVELY, $REG'S
;ARE USED FOR TYPING OUT THERE CONTENTS AT THE TIME OF ERROR
GT4RG: MOV @RKCS,$REG0 ;GET RKCS
MOV @RKER,$REG1 ; RKER
MOV @RKDS,$REG2 ; RKDS
MOV @RKDA,$REG3 ; RKDA
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.
GETINF: JSR PC,GT4RG
BRKDA: MOV R0,-(SP)
MOV R1,-(SP)
MOV R2,-(SP)
MOV $REG7-2,R0
MOV $REG3,R1
MOV R1,R2
BIC #177760,R2
MOV R2,-(R0)
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
4434
4435
4436
4437
4438

```

.SBTTL ERR2

```

;ERR2
;THIS ROUTINE GETS THE CYLINDER NUMBERS BETWEEN WHICH (IMPLIED) SEEK
;WAS DONE. (R4)=0 INDICATES SEEK FROM 'OUTADR' TO 'INADR'
;(R4)=1 INDICATES SEEK TO 'OUTADR'. ON EXIT $REGO CONTAINS CYL #
;FROM WHICH SEEK WAS INITIATED, $REG1 CONTAINS CYL # TO WHICH SEEK WAS DONE
ERR2: MOV INADR,$REGO ;GET CYL ADRES
JSR PC,GCYL ;GO GET CYL# FROM IT
MOV $REGO,$REG1 ;SAVE
MOV OUTADR,$REGO ;GET CYL ADRES
JSR PC,GCYL ;GO GET CYL # FROM IT
TST R4 ;GOING WHICH WAY?
BEQ 1$ ;'OUTADR' TO 'INADR', BRANCH
MOV $REGO,=(SP) ;EXCHANG CYL" TO GET
MOV $REG1,$REGO ;CORRECT 'TO' & 'FROM' CYLS
MOV (SP)+,$REG1
1$: RTS PC ;RETURN

```

.SBTTL ERR1

```

;ERR1
;THIS SUBROUTINE FINDS OUT THE CYLINDER NOS. BETWEEN WHICH THE SEEK
;IS DONE,THE CYLINDER # WHERE THE HEADS WERE PRIOR TO MOVING, IS
;DEPOSITED IN $REGO. THE CYLINDER # WHERE THE HEADS SHOULD BE AFTER
;MOVEMENT, IS DEPOSITED IN $REG1. R4 INDICATES WHICH DIRECTION THE
;HEADS WERE MOVING, IN OR OUT. R5 CONTAINS THE
;DISK ADDRESS (IN OR OUT AS THE CASE MAY BE).

```

```

4459 016376 010537 001162 ERR1: MOV R5,$REGO
4461 016402 004737 016434 JSR PC,GCYL ;GO GET CYL #
4462 016406 005704 TST R4 ;WAS GOING IN OR OUT?
4463 016410 001006 BNE 1$ ;OUT
4464 016412 013737 001162 001164 MOV $REGO,$REG1
4465 016420 005037 001162 CLR $REGO

```

```

4466 016424 000207 1$: RTS PC
4467 016426 005037 001164 CLR $PEG1
4468 016432 000207 RTS PC
4469
4470
4471
4472
4473
4474

```

.SBTTL GCYL

```

;GCYL
;THIS ROUTINE EXTRACTS THE CYLINDER NO. FROM THE DISK ADDRESS
;CONTAINED IN '$REGO', AND THEN STORES IT BACK IN '$REGO'
GCYL: MOV RO,=(SP) ;PUSH RO ONTO STACK
MOV $REGO,RO
BIC #160037,RO ;MASK OUT DRV # BITS &
;SUR, SEC BITS IF PRESENT
ASR RO
ASR R0 ;SHIFT CYL BITS RIGHT
ASR R0 ;BY 5
ASR R0
ASR R0
MOV RO,$REGO ;STORE CYL # IN $REGO
MOV (SP)+,R0 ;POP RO FROM STACK
RTS PC ;EXIT

```

.SBTTL DRV,RESET = DRIVE RESET ROUTINE  
 .SBTTL RESDON = WAIT FOR DRIVE RESET TO BE DONE

```

;DR,RST
;THIS ROUTINE DOES A DRIVE RESET ON THE DRIVE WHOOSE ADDRESS IS IN
;RKDA. MULTIPLE RETURN ADDRESSES FOR THIS ROUTINE ARE PROVIDED.
;IF THERE IS NO ERROR (R/W/S RDY SETS WITHIN CERTAIN TIME) , THEN
;A NORMAL EXIT IS MADE. IF R/W/S RDY DOES NOT SET ERROR IS REPORTED.

```

```

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 $P
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 ACTUALLY
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 'STRAP'.
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 993 MS FOR 11/20
4550 ;175 MS FOR 11/45 WITH BIPOLAR MEMORY.
4551 016602 012777 000001 162646 CN,RST: MOV    #1,0RKCS      ;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      16:  TSTB   0RKCS        ;DID CNTRL-RDY SET?
4556 016630 100431                BMI     26              ;YES, EXIT
4557 016632 005237 001170      INC     $REG3          ;WAITED LONG?
4558 016636 001372                BNE    16              ;IF NOT, GO BAK & WAIT
4559 016640 104420                TYPMSG
4560 016642 061742                MSG10
4561 016644 104401 016652      TYPE    ,658           ;TYPE ASCIZ STRING
4562 016650 000403                BR      648            ;GET OVER THE ASCIZ
4563
4564          ,ASCIZ <15><12>/PC=/
4565          648:
4566          MOV    (SP),-(SP)
4567          SUB    #2,(SP)
4568          TYPOC                ;GO TYPE PC IN THE MAIN PROGRAM,
4569
4570
4571
4572
4573
4574
4575
4576
4577
4578
4579
4580
4581
4582
4583
4584
4585
4586
4587
4588          016716 005037 001264      TSTRWS: CLR    TIMER
4589          016722 032777 000100 162522 16:  BIT     #100,0RKDS
4590          016730 001017                BNE    26
4591          016732 005237 001264      INC     TIMER
4592          016736 001371                BNE    16
4593          016740 032777 020000 162172      BIT     $BIT13,$SWR
4594          016746 001010                BNE    26
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      26:  RTI
4601
4602
4603
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
4612
4613
4614
4615
4616
4617
4618
4619
4620
4621
4622
4623
4624
4625          017006
4626          017006 104407
4627          017010 032777 000400 162122      CKSWR
4628          017016 001053                BIT     $SW8,$SWR      ;;TEST FOR CHANGE IN SOFT-SWR
4629
4630          017020 032777 040000 162112      BNE    $OVER          ;;WAS SW8 USED TO SELECT
4631          017026 001047                ;;A TEST? IF YES, SKIP OVER
4632
4633          017030 000416      16:  BIT     $BIT14,$SWR  ;;THE REST, U ARE LOOPING ON
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673
4674
4675
4676
4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800

```

```

4578
4579
4580
4581
4582
4583
4584
4585
4586
4587
4588          016716 005037 001264      TSTRWS: CLR    TIMER
4589          016722 032777 000100 162522 16:  BIT     #100,0RKDS
4590          016730 001017                BNE    26
4591          016732 005237 001264      INC     TIMER
4592          016736 001371                BNE    16
4593          016740 032777 020000 162172      BIT     $BIT13,$SWR
4594          016746 001010                BNE    26
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      26:  RTI
4601
4602
4603
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
4612
4613
4614
4615
4616
4617
4618
4619
4620
4621
4622
4623
4624
4625          017006
4626          017006 104407
4627          017010 032777 000400 162122      CKSWR
4628          017016 001053                BIT     $SW8,$SWR      ;;TEST FOR CHANGE IN SOFT-SWR
4629
4630          017020 032777 040000 162112      BNE    $OVER          ;;WAS SW8 USED TO SELECT
4631          017026 001047                ;;A TEST? IF YES, SKIP OVER
4632
4633          017030 000416      16:  BIT     $BIT14,$SWR  ;;THE REST, U ARE LOOPING ON
4634
4635
4636
4637
4638
4639
4640
4641
4642
4643
4644
4645
4646
4647
4648
4649
4650
4651
4652
4653
4654
4655
4656
4657
4658
4659
4660
4661
4662
4663
4664
4665
4666
4667
4668
4669
4670
4671
4672
4673
4674
4675
4676
4677
4678
4679
4680
4681
4682
4683
4684
4685
4686
4687
4688
4689
4690
4691
4692
4693
4694
4695
4696
4697
4698
4699
4700
4701
4702
4703
4704
4705
4706
4707
4708
4709
4710
4711
4712
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
4738
4739
4740
4741
4742
4743
4744
4745
4746
4747
4748
4749
4750
4751
4752
4753
4754
4755
4756
4757
4758
4759
4760
4761
4762
4763
4764
4765
4766
4767
4768
4769
4770
4771
4772
4773
4774
4775
4776
4777
4778
4779
4780
4781
4782
4783
4784
4785
4786
4787
4788
4789
4790
4791
4792
4793
4794
4795
4796
4797
4798
4799
4800

```

```

4634      017032 013746 000004      MOV    @@ERRVEC,-(SP)    ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
4635      017036 012737 017056 000004      MOV    #58,@@ERRVEC    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
4636      017044 005737 177060      TST    @#177060        ;;SET FOR TIMEOUT
4637      017050 012637 000004      MOV    (SP)+,@@ERRVEC  ;;TIME OUT ON XOR?
4638      017054 000421 000004      BR     $SVLAD           ;;RESTORE THE ERROR VECTOR
4639      017056 022626 000004      58:   CMP    (SP)+,(SP)+  ;;GO TO THE NEXT TEST
4640      017060 012637 000004      MOV    (SP)+,@@ERRVEC  ;;CLEAR THE STACK AFTER A TIME OUT
4641      017064 000407 000004      BR     78              ;;RESTORE THE ERROR VECTOR
4642      017066 105737 001103      68:   TSTB   $ERFLG      ;;LOOP ON THE PRESENT TEST
4643      017072 032777 001000 162036      28:   BEQ    $SVLAD       ;;TESTER****
4644      017074 032777 001000 162036      BIT    $BIT09,$SWR     ;;HAS AN ERROR OCCURRED?
4645      017102 001404 001110 001106      BEQ    48              ;;BR IF NO
4646      017104 013737 001110 001106      78:   MOV    $LPERR,$LPADR ;;LOOP ON ERROR?
4647      017112 000415 001110 001106      BR     $OVER          ;;BR IF NO
4648      017114 105037 001103      48:   CLR    $ERFLG      ;;SET LOOP ADDRESS TO LAST SCOPE
4649      017120 105237 001102      $SVLAD: INCB   $STNM    ;;ZERO THE ERROR FLAG
4650      017124 011637 001106      MOV    (SP), $LPADR   ;;COUNT TEST NUMBERS
4651      017130 011637 001110      MOV    (SP), $LPERR   ;;SAVE SCOPE LOOP ADDRESS
4652      017134 005037 001204      CLR    $ESCAPE       ;;SAVE ERROR LOOP ADDRESS
4653      017140 112737 000001 001115      MOV    #1,$ERMAX     ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
4654      017146 013777 001102 161766      $OVER: MOV    $STNM,$DISPLAY ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
4655      017154 013716 001106      MOV    $LPADR,(SP)   ;;DISPLAY TEST NUMBER
4656      017160 000002 000002      RTI                  ;;FUJGE RETURN ADDRESS
4657      017160 000002 000002      RTI                  ;;FIXES PS
4658
4659
4660      ;*****
4661
4662      ,SBTTL  ERROR HANDLER ROUTINE
4663
4664      ;*SW15=1      HALT ON ERROR
4665      ;*SW13=1      INHIBIT ERROR TYPEOUTS
4666      ;*SW10=1      BELL ON ERROR
4667      ;*SW09=1      LOOP ON ERROR
4668      ;*SW12=1     CYCLE ON ERROR TO PREVIOUS 'SCOPE' STATEMENT
4669      ;GO TO ERRTP ON ERROR
4670      ;NOT FROM SYSMAC
4671
4672      $ERROR: CKSWR      ;CHECK FOR SOFTWARE SWITCH REGISTER REQUEST
4673      017162 104407 001103      78:   INCB   $ERFLG      ;SET THE ERROR FLAG
4674      017164 105237 001103      BEQ    78              ;DON'T LET THE FLAG GO TO ZERO
4675      017170 001775 001102 161742      MOV    $STNM,$DISPLAY
4676      017172 013777 001102 161732      BIT    $SW10,$SWR
4677      017200 032777 002000 161732      BEQ    18              ;
4678      017206 001402 001206      TYPE   $BELL
4679      017210 104401 001206      TYPE   $BELL
4680
4681      18:   INC    $ERTTL
4682      017214 005237 001112      MOV    (SP),$ERRPC
4683      017220 011637 001116      MOV    (SP),$ERRPC
4684
4685      38:   BIT    $SW2,$SWR    ;DROP THE DRIVE?
4686      017224 032777 000004 161706      BEQ    58              ;SW NOT SET, SKIP
4687      017232 001404 000006      CMP    $ERTTL,#6     ;MORE THAN 6 ERRORS ON THIS DRIVE?
4688      017234 023727 001112 000006      BHI    68              ;YES, DROP THE DRIVE
4689      017242 101044 000002 001116      58:   SUB    #2,$ERRPC
4690      017244 162737 000002 001116      MOV    @@ERRPC,$ITEMB
4691      017252 117737 161640 001114      MOV    @@ERRPC,$ITEMB

```

```

4690      017260 032777 020000 161652      BIT    $SW13,$SWR
4691      017266 001004 001004      28:   BNE    28
4692      017270 004737 017446      JSR    PC,@@ERRTP
4693      017274 104401 001213      TYPE   $CRLF
4694      017300 023737 000042 000046      28:   CMP    @#42,@#46     ;ARE WE IN ACT11 AUTO MODE?
4695      017306 001403 000004 000046      BEQ    +10            ;YES, HALT ON ERROR
4696      017310 005777 161624      TST    @SWR
4697      017314 100002 000002 161624      BPL    38             ;SWR15 (HALT ON ERROR) SET?
4698      017316 000000 000000      HALT
4699      017320 104407 000000 000000      CKSWR      ;BRANCH IF NOT
4700      017322 032777 010000 161610      38:   BIT    $SW12,$SWR    ;CHECK FOR SOFTWARE SWITCH REGISTER REQUEST
4701      017330 001402 001106      BEQ    +6
4702      017332 013716 001106      MOV    $LPADR,(SP)
4703      017336 032777 001000 161574      BIT    $SW09,$SWR
4704      017344 001402 001110      BEQ    48
4705      017346 013716 001110      MOV    $LPERR,(SP)
4706      017352 000002 000002      48:   RTI
4707
4708      68:   MOV    DRVPTX,-(SP)  ;GET POINTER TO DRIVE #
4709      017354 013746 001226      SUB    #2,(SP)
4710      017360 162716 000002 000002      BIC    #377,@(SP)+  ;CLEAR THE DRIVE PRESENT FLAG
4711      017364 042736 000377      TYPE   $MSG14
4712      017370 104401 002064      MOV    DRIVAD,-(SP)
4713      017374 013746 001230      MOV    DRIVAD,-(SP)
4714      017400 000241 000000      CLC
4715      017402 006116 000000      ROL    (SP)
4716      017404 006116 000000      ROL    (SP)
4717      017406 006116 000000      ROL    (SP)
4718      017410 006116 000000      ROL    (SP)
4719      017412 104402 000000      TYPCC
4720      017414 104401 017422      TYPE   ,65$          ;GET THE DRIVE #
4721      017420 000405 000000      BR     64$          ;TYPE IT OUT
4722      017424 000405 000000      BR     64$          ;TYPE ASCIZ STRING
4723      017434 105337 001224      ;65$: ,ASCIZ / DROPPED/ ;GET OVER THE ASCIZ
4724      017440 022626 000000      64$:   DECB   DRIVS      ;DECRMNT # OF DRIVS PRESENT
4725      017442 000137 015232      CMP    (SP)+,(SP)+  ;RESTORE STACK
4726      017446 000137 015232      JMP    BTEOP        ;EXIT
4727
4728      ERRTP: TYPE   $CRLF    ;"CARRIAGE RETURN" & LINE FEED"
4729      017446 104401 001213      MOV    RO,-(SP)     ;SAVE RO
4730      017452 010046 000000      CLR    RO           ;PICKUP THE ITEM INDEX
4731      017454 005000 001114      BISB  @#$ITEMB,RO  ;IF ITEM NUMBER IS ZERO, JUST
4732      017456 153700 001114      BNE    18
4733      017462 001011 000000      ;TYPE THE PC OF THE ERROR
4734      017464 013746 001116      MOV    $ERRPC,-(SP) ;SAVE $ERRPC FOR TYPEOUT
4735      017470 104402 001116      TYPCC              ;ERROR ADDRESS
4736      017472 104401 001116      TYPE   $MSG7        ;GO TYPE=OCTAL ASCII(ALL DIGITS)
4737      017474 001733 001116      MOV    $ERRPC,-(SP)
4738      017476 013746 001116      MOV    $ERRPC,-(SP)
4739      017502 104402 000000      TYPCC
4740      017504 000440 000000      BR     68
4741      017506 005300 000000      18:   DEC    R0
4742      017510 006300 000000      ASL   R0
4743      017512 006300 000000      ASL   R0
4744
4745

```

```

4746 017514 006300 ASL RO ;FORM TABLE POINTER
4747 017516 062700 002122 ADD #ERRTB,RO ;PICKUP "ERROR MESSAGE" POINTER
4748 017522 012037 017532 MOV (RO)+,2# ;SKIP TYPEOUT IF NOT POINTER
4749 017526 001404 BEQ 3# ;TYPE THE "ERROR MESSAGE"
4750 017530 104401 TYPE ;"CARRIAGE RETURN" & LINE FEED"
4751 017532 000000 2#; .WORD 0 ;PICKUP "DATA HEADER" POINTER
4752 017534 104401 001213 TYPE ,#CRLF ;DUMP OUT ALL RK REGISTERS
4753 017540 032777 004000 161372 3#; BIT #SW11,@SWR ;YES, BRANCH
4754 017546 001042 BNE 10# ;PICKUP "DATA HEADER" POINTER
4755 017550 012037 017560 MOV (RO)+,4# ;SKIP TYPEOUT IF 0
4756 017554 001412 BEQ 5# ;TYPE THE "DATA HEADER"
4757 017556 104401 .WORD 0 ;"DATA HEADER" POINTER GOES HERE
4758 017560 000000 4#; TYPE ,#CRLF ;"CARRIAGE RETURN" & LINE FEED"
4759 017562 104401 001213 ADD #2,RO ;FORM POINTER TO TERMINATOR
4760 017566 062700 000002 TST (RO) ;IS THE TERMINATOR 0?
4761 017572 005710 BNE 9# ;IF NOT, BRANCH
4762 017574 001017 SUB #2,RO ;YES, IT IS 0. REPOINT TO "DATA"
4763 017576 162700 000002 ;GO TYPE OUT DATA AS USUAL
4764 ;PICKUP "DATA TABLE" POINTER
4765 017602 011000 5#; MOV (RO),RO ;GO TYPE THE DATA
4766 017604 001004 BNE 7# ;RESTORE RO
4767 017606 012600 6#; MOV (SP)+,RO ;"CARRIAGE RETURN" & LINE FEED"
4768 017610 104401 001213 TYPE ,#CRLF ;RETURN
4769 017614 000207 RTS
4770 017616 7#;
4771 017616 013046 7#; MOV @(RO)+,-(SP) ;SAVE @(RO)+ FOR TYPEOUT
4772 017620 104402 TYPDC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
4773 017622 005710 TST (RO) ;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 9#; JSR PC,@(RO) ;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 TYPDC
4788 017652 000755 BR 6# ;GO BACK, TO THE EXIT POINT
4789 ;FOR "ERRIYP"
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
    
```

```

4802 ;65#; .ASCIZ <15><12>/ PC RKDS RKER RKCS RKWC RKBA RKDA RKDB/<
4803 017772 64#;
4804 017772 013746 001116 MOV #ERRPC,-(SP)
4805 017776 104402 TYPDC
4806 020000 104401 002110 TYPE ,BLNKS2
4807 020004 010046 MOV RO,-(SP)
4808 020006 012700 001452 MOV #RKDS,RO
4809 020012 013046 1#; MOV @(RO)+,-(SP)
4810 020014 104402 TYPDC
4811 020016 104401 002110 TYPE ,BLNKS2
4812 020022 020027 001466 CMP RO,#RKDB
4813 020026 003771 BLE 1#
4814 020030 012600 MOV (SP)+,RO
4815 020032 000207 RTS PC
4816
4817
4818
4819
4820
4821 ;SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
4822
4823 ;*****
4824 ;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
4825 ;SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
4826 ;NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
4827 ;BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
4828 ;REPLACED WITH SPACES.
4829 ;CALL:
4830 ;* MOV NUM,-(SP) ;PUT THE BINARY NUMBER ON THE STACK
4831 ;* TYPDS ;GO TO THE ROUTINE
4832
4833 ;TYPDS:
4834 020034 MOV RO,-(SP) ;PUSH RO ON STACK
4835 020036 010146 MOV R1,-(SP) ;PUSH R1 ON STACK
4836 020040 010246 MOV R2,-(SP) ;PUSH R2 ON STACK
4837 020042 010346 MOV R3,-(SP) ;PUSH R3 ON STACK
4838 020044 010546 MOV R5,-(SP) ;PUSH R5 ON STACK
4839 020046 012746 020200 MOV #20200,-(SP) ;SET BLANK SWITCH AND SIGN
4840 020052 016605 000020 MOV 20(SP),R5 ;GET THE INPUT NUMBER
4841 020056 100004 BPL 1# ;BR IF INPUT IS POS.
4842 020060 005405 NEG R5 ;MAKE THE BINARY NUMBER POS.
4843 020062 112766 000055 000001 MOVB #"-",1(SP) ;MAKE THE ASCII NUMBER NEG.
4844 020070 005000 CLR RO ;ZERO THE CONSTANTS INDEX
4845 020072 012703 020250 MOV #DBLK,R3 ;SETUP THE OUTPUT POINTER
4846 020076 112723 000040 MOVB #",{R3}+ ;SET THE FIRST CHARACTER TO A BLANK
4847 020102 005002 2#; CLR R2 ;CLEAR THE BCD NUMBER
4848 020104 016001 020240 MOV #DTBL(RO),R1 ;GET THE CONSTANT
4849 020110 160105 3#; SUB R1,R5 ;FORM THIS BCD DIGIT
4850 020112 002402 BLT 4# ;BR IF DONE
4851 020114 005202 INC R2 ;INCREASE THE BCD DIGIT BY 1
4852 020116 000774 BR 3#
4853 020120 060105 4#; ADD R1,R5 ;ADD BACK THE CONSTANT
4854 020122 005702 TST R2 ;CHECK IF BCD DIGIT=0
4855 020124 001002 BNE 5# ;FALL THROUGH IF 0
4856 020126 105716 TSTB (SP) ;STILL DOING LEADING 0'S?
4857 020130 100407 BMI 7# ;BR IF YES
    
```

```

4858 020132 106316          50: ASLB (SP)          ;;MSD?
4859 020134 103003          BCC 6#             ;;BR IF NO
4860 020136 116663 000001 177777 MOVB 1(SP),-1(R3)  ;;YES--SET THE SIGN
4861 020144 032702 000060 68: BLS #'0,R2      ;;MAKE THE BCD DIGIT ASCII
4862 020150 032702 000040 78: BLS #' ,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 CMPB R0,#10       ;;CHECK THE TABLE INDEX
4866 020164 002745          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          STB (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          CLRB (R3)        ;;SET THE TERMINATOR
4874 020210 012603          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 000004 TYPE $DBLK        ;;NOW TYPE THE NUMBER
4880 020226 016666          MOV 2(SP),4(SP)  ;;ADJUST THE STACK
4881 020234 012616          RTI              ;;RETURN TO USER
4882 020236 000002          $DTBL: 10000,
4883 020240 023420          1000,
4884 020242 001750          100,
4885 020244 000144          10,
4886 020246 000012          $DBLK: ,BLKW 4
4887 020250 000004
4888
4889
4890
4891
4892
4893
4894
4895
4896
4897
4898
4899
4900
4901
4902
4903
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          MOVB R0,-(SP)   ;;SAVE R0
4911 020274 017600 000002 1#: MOV $2(SP),R0    ;;GET ADDRESS OF ASCII STRING
4912 020300 112046          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          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#          ;;BRANCH IF NOT <CRLF>
4920 020324 122716 000200 CMPB $CRLF,(SP)  ;;BRANCH IF NOT <CRLF>
4921 020330 001006          BNE 5#          ;;BRANCH IF NOT <CRLF>
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 $CHARCNT    ;;CLEAR CHARACTER COUNT
4926 020344 000755          BR 2#           ;;GET NEXT CHARACTER
4927 020346 004737 020430 5#: JSR PC,$TYPEPC ;;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 $NULL,-(SP)  ;;GET # OF FILLER CHARS. NEEDED
4931
4932 020364 105366 000001 7#: DECB 1(SP)     ;;DOES A NULL NEED TO BE TYPED?
4933 020370 002770          BLT 6#          ;;BR IF NO--GO POP THE NULL OFF OF STACK
4934 020372 004737 020430 JSR PC,$TYPEPC  ;;GO TYPE A NULL
4935 020376 105337 020474 DECB $CHARCNT   ;;DO NOT COUNT AS A COUNT
4936 020402 000770          BR 7#           ;;LOOP
4937
4938
4939
4940 020404 112716 000040 8#: MOVB #' ,(SP)  ;;REPLACE TAB WITH SPACE
4941 020410 004737 020430 9#: JSR PC,$TYPEPC ;;TYPE A SPACE
4942 020414 132737 000007 020474 BITB #',$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 $TYPEPC: TSTB $#TPS   ;;WAIT UNTIL PRINTER IS READY
4947 020434 100375          BPL $TYPEPC
4948 020436 116677 000002 160506 MOVB 2(SP),#6TPB ;;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 $TYPEPC      ;;EXIT
4953 020462 122766 000012 000002 1#: CMPB $LF,2(SP) ;;IS CHARACTER A LINE FEED?
4954 020470 001402          BEQ $TYPEPC     ;;BRANCH IF YES
4955 020472 105227          INCB (PC)+      ;;COUNT THE CHARACTER
4956 020474 000000          $CHARCNT: WORD 0 ;;CHARACTER COUNT STORAGE
4957 020476 000207          $TYPEPC: RTS PC
4958
4959
4960
4961
4962
4963
4964
4965
4966
4967
4968
4969

```

```

4970
4971
4972
4973
4974
4975 020500
4976 020500 010046
4977 020502 010146
4978 020504 010246
4979 020506 005046
4980 020510 016601 000012
4981 020514 100002
4982 020516 005216
4983 020520 005401
4984 020522 016602 000014
4985 020526 100002
4986 020530 005316
4987 020532 005402
4988 020534 012746 000021
4989 020540 005000
4990 020542 103001
4991 020544 060200
4992 020546 006000
4993 020550 006001
4994 020552 005316
4995 020554 001372
4996 020556 022616
4997 020560 001403
4998 020562 005400
4999 020564 005401
5000 020566 005600
5001 020570 005726
5002 020572 010066 000012
5003 020576 010166 000010
5004 020602 012602
5005 020604 012601
5006 020606 012600
5007 020610 000207
5008
5009
5010
5011
5012
5013 020612 000000
5014 020614 000000
5015 020616 000000
5016 020620 000001
5017 020621
5018 020622
5019
5020
5021
5022
5023
5024
5025

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

SMULT:
MOV RO,=(SP) ;PUSH RO ON STACK
MOV R1,=(SP) ;PUSH R1 ON STACK
MOV R2,=(SP) ;PUSH R2 ON STACK
CLR =(SP) ;CLEAR THE SIGN KEY
MOV 12(SP),R1 ;GET THE MULTIPLICAND
BPL 16 ;BR IF PLUS
INC (SP) ;SET THE SIGN KEY
NEG R1 ;MAKE THE MULTIPLICAND POSTIVE
18: MOV 14(SP),R2 ;GET THE MULTIPLIER
BPL 28 ;BR IF PLUS
DEC (SP) ;UPDATE THE SIGN KEY
NEG R2 ;MAKE THE MULTIPLIER POSTIVE
28: MOV #17,=(SP) ;SET THE LOOP COUNT
CLR RO ;SETUP FOR THE MULTIPLY LOOP
36: BCC 48 ;DON'T ADD IF MULTIPLICAND = 0
ADD R2,RO
48: ROR RO ;POSITION THE PARTIAL PRODUCT AND
ROR R1 ;THE MULTIPLICAND
DEC (SP) ;HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
BNE 38 ;BR IF NO
CMP (SP)+,(SP) ;SHOULD PRODUCT BE NEGATIVE?
BEQ 58 ;GO TO EXIT IF NO
NEG RO ;YES--SO MAKE IT SO
NEG R1
SBC RO
58: TST (SP)+ ;CLEAR SIGN INFO, OFF OF STACK
MOV RO,12(SP) ;PUT THE PRODUCT ON THE STACK (MSB'S)
MOV R1,10(SP) ;LSB'S
MOV (SP)+,R2 ;POP STACK INTO R2
MOV (SP)+,R1 ;POP STACK INTO R1
MOV (SP)+,RO ;POP STACK INTO RO
RTS PC

,SBTTL TTY INPUT ROUTINE
;*****
,ENABL LSB
$TKCNT: ,WORD 0 ;NUMBER OF ITEMS IN QUEUE
$TKQIN: ,WORD 0 ;INPUT POINTER
$TKQOUT: ,WORD 0 ;OUTPUT POINTER
$TKQSR: ,BLKB 1 ;TTY KEYBOARD QUEUE
$TKQEND=
,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
5027
5028 020622 005037 020612
5029 020626 012737 020620 020614
5030 020634 013737 020614 020616
5031 020642 012737 020672 000060
5032 020650 012737 000200 000062
5033 020656 005777 160264
5034 020662 012777 000100 160254
5035 020670 000207
5036
5037
5038
5039
5040
5041
5042 020672 117746 160250
5043 020676 042716 177600
5044 020702 021627 000007
5045 020706 001004
5046 020710 022737 000176 001140
5047 020716 001500
5048
5049 020720
5050 020720 022737 000001 020612
5051 020726 001004
5052 020730 104401 001206
5053 020734 005726
5054 020736 000451
5055 020740 021627 000023
5056 020744 001021
5057 020746 005077 160172
5058 020752 005726
5059 020754 105777 160164
5060 020760 100375
5061 020762 117746 160160
5062 020766 042716 177600
5063 020772 022627 000021
5064 020776 001366
5065 021000 012777 000100 160136
5066 021006 000002
5067 021010 005237 020612
5068 021014 021627 000140
5069 021020 002405
5070 021022 021627 000175
5071 021026 003002
5072 021030 042716 000040
5073 021034 112677 177554
5074 021040 005237 020614
5075 021044 023727 020614 020621
5076 021052 001003
5077 021054 012737 020620 020614
5078 021062 000002
5079
5080
5081

;* RETURN
;
$TKINT: CLR $TKCNT ;CLEAR COUNT OF ITEMS IN QUEUE
MOV $TKQSR,$TKQIN ;MOVE THE STARTING ADDRESS OF THE
;QUEUE INTO THE INPUT & OUTPUT POINTERS,
MOV $TKSRV,$TKVEC ;INITIALIZE THE KEYBOARD VECTOR
MOV #200,$TKVEC+2 ;"BR" LEVEL 4
TST @TKB ;CLEAR DONE FLAG
MOV #100,$TKS ;ENABLE TTY KEYBOARD INTERRUPT
RTS PC ;RETURN TO CALLER

;TK SERVICE ROUTINE
;THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
;BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
;IT IN THE QUEUE.
;
$TKSRV: MOVB @TKB,=(SP) ;PICKUP THE CHARACTER
BIC #"C177,(SP) ;STRIP THE JUNK
18: CMP (SP),#7 ;IS IT A CONTROL G?
BNE 28 ;BRANCH IF NO
CMP $SWREG,$SWR ;IS SOFT-SWR SELECTED?
BEQ 68 ;GO TO SWR CHANGE
28: CMP #1,$TKCNT ;IS THE QUEUE FULL?
BNE 38 ;BRANCH IF NO
;RING THE TTY BELL
TST (SP)+ ;CLEAN CHARACTER OFF OF STACK
BR 58 ;EXIT
38: CMP (SP),#23 ;IS IT A CONTROL-S?
BNE 328 ;BRANCH IF NO
CLR @TKS ;DISABLE TTY KEYBOARD INTERRUPTS
TST (SP)+ ;CLEAN CHAR OFF STACK
318: TSTB @TKS ;WAIT FOR A CHAR
BPL 316 ;LOOP UNTIL ITS THERE
MOVB @TKB,=(SP) ;GET THE CHARACTER
BIC #"C177,(SP) ;MAKE IT 7-BIT ASCII
CMP (SP)+,#21 ;IS IT A CONTROL-Q?
BNE 318 ;BRANCH IF NO
MOV #100,$TKS ;REENABLE TTY KEYBOARD INTERRUPTS
;RETURN
328: INC $TKCNT ;COUNT THIS CHARACTER
CMP (SP),#140 ;IS IT UPPER CASE?
BLT 48 ;BRANCH IF YES
CMP (SP),#175 ;IS IT A SPECIAL CHAR?
BGT 48 ;BRANCH IF YES
BIC #40,(SP) ;MAKE IT UPPER CASE
48: MOVB (SP)+,$TKQIN ;AND PUT IT IN QUEUE
INC $TKQIN ;UPDATE THE POINTER
CMP $TKQIN,$TKQEND ;GO OFF THE END?
BNE 58 ;BRANCH IF NO
MOV $TKQSR,$TKQIN ;RESET THE POINTER
58: RTI ;RETURN

;*****
;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 $KCSWR: CMP $SWREG,SWR ;*IS THE SOFT-SWR SELECTED
5086 BNE 15$ ;*EXIT IF NOT
5087 021074 001104 105777 160044 TSTB @8TKS ;*IS A CHAR WAITING?
5088 021100 100101 BPL 15$ ;*IF NOT, EXIT
5089 021102 117746 MOV B @8TKB,-(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 JSP PC,$TKINT ;*FLUSH THE TTY INPUT QUEUE
5103 021136 005077 CLR @8TKS ;*DISABLE TTY KEYBOARD INTERRUPTS
5104 021142 112737 000001 001135 MOV B #1,$INTAG ;*SET INTERRUPT MODE INDICATOR
5105
5106 021150 104401 021727 $GTSWR: TYPE ;*SCNTLG ;*ECHO THE CONTROL-G ("G")
5107 021154 104401 021734 ;*MSWR ;*TYPE CURRENT CONTENTS
5108 021160 013746 000176 MOV SWREG,-(SP) ;*SAVE SWREG FOR TYPEOUT
5109 021164 104402 TYPCC ;*GO TYPE=OCTAL ASCII(ALL DIGITS)
5110 021166 104401 021745 TYPE ;*$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 @8TKS ;*CHAR THERE?
5114 021202 100375 BPL 7$ ;*IF NOT TRY AGAIN
5115
5116 021204 117746 157736 MOV B @8TKB,-(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 ;*SCNTLU ;*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 118: ADD #6,SP ;*CLEAR UP STACK
5134 021262 104401 001213 14$: TYPE ;*SCRLF ;*ECHO <CR> AND <LF>
5135 021266 123727 001135 000001 CMPB $INTAG,#1 ;*RE-ENABLE TTY KRD INTERRUPTS?
5136 021274 001003 BNE 15$ ;*BRANCH IF NOT
5137 021276 012777 000100 157640 MOV #100,@8TKS ;*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
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 000004 000002 $RDCHR: MOV (SP),-(SP) ;*PUSH DOWN THE PC AND
5167 021370 016666 MOV 4(SP),2(SP) ;*THE PS
5168 021376 005066 000004 CLR 4(SP) ;*GET READY FOR A CHARACTER
5169 021402 005046 CLR -(SP) ;*PUT NEW PS ON STACK
5170 021404 012746 021412 MOV #64$,-(SP) ;*PUT NEW PC ON STACK
5171 021410 000002 RTI ;*POP NEW PC AND PS
5172
5173 021412 005737 020612 64$: 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 MOV B @8TKQOUT,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 #8TKQSR,$TKQOUT ;*RESET THE POINTER
5181 021454 000002 2$: 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 #8TTYIN,R3 ;*GET ADDRESS
5192 021466 022703 021722 2$: CMP #8TTYIN+8,R3 ;*BUFFER FULL?
5193 021472 101456 BLOS 4$ ;*BR IF YES
```



```
5194 021474 104410 RDCHR ;GO READ ONE CHARACTER FROM THE TTY
5195 021476 112613 MOV (SP)+,(R3) ;GET CHARACTER
5196 021500 122713 000177 108: CMPB #177,(R3) ;IS IT A RUBOUT
5197 021504 001022 BNE 58 ;BR IF NO
5198 021506 005716 TST (SP) ;IS THIS THE FIRST RUBOUT?
5199 021510 001007 BNE 68 ;BR IF NO
5200 021512 112737 000134 021710 MOVB #' \,98 ;TYPE A BACK SLASH
5201 021520 104401 021710 TYPE ,98
5202 021524 102716 177777 MOV #1,(SP) ;SET THE RUBOUT KEY
5203 021530 005303 68: DEC R3 ;BACKUP BY ONE
5204 021532 020327 021712 CMP R3,#8TTYIN ;STACK EMPTY?
5205 021536 103434 BLO 48 ;BR IF YES
5206 021540 111337 021710 MOVB (R3),98 ;SETUP TO TYPEOUT THE DELETED CHAR.
5207 021544 104401 021710 TYPE ,98 ;GO TYPE
5208 021550 000746 BR 28 ;GO READ ANOTHER CHAR.
5209 021552 005716 58: TST (SP) ;RUBOUT KEY SET?
5210 021554 001406 BEQ 78 ;BR IF NO
5211 021556 112737 000134 021710 MOVB #' \,98 ;TYPE A BACK SLASH
5212 021564 104401 021710 TYPE ,98
5213 021570 005016 CLR (SP) ;CLEAR THE RUBOUT KEY
5214 021572 122713 000025 78: CMPB #25,(R3) ;IS CHARACTER A CTRL U?
5215 021576 001003 BNE 88 ;BR IF NO
5216 021600 104401 021722 TYPE ,8CNTLU ;TYPE A CONTROL "U"
5217 021604 000726 BR 18 ;GO START OVER
5218 021606 122713 000022 88: CMPB #22,(R3) ;IS CHARACTER A "R"?
5219 021612 001011 BNE 36 ;BRANCH IF NO
5220 021614 105013 CLR (R3) ;CLEAR THE CHARACTER
5221 021616 104401 001213 TYPE ,8CRLF ;TYPE A "CR" & "LF"
5222 021622 104401 021712 TYPE ,8TTYIN ;TYPE THE INPUT STRING
5223 021626 000717 BR 28 ;GO PICKUP ANOTHER CHARACTER
5224 021630 104401 001212 48: TYPE ,8QUES ;TYPE A "?"
5225 021634 000712 BR 18 ;CLEAR THE BUFFER AND LOOP
5226 021636 111337 021710 MOVB (R3),98 ;ECHO THE CHARACTER
5227 021642 104401 021710 TYPE ,98
5228 021646 122723 000015 CMPB #15,(R3)+ ;CHECK FOR RETURN
5229 021652 001305 BNE 28 ;LOOP IF NOT RETURN
5230 021654 105063 177777 CLRB -1(R3) ;CLEAR RETURN (THE 15)
5231 021660 104401 001214 TYPE ,8LF ;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 UN IT
5236 021700 012766 021712 000004 MOV #8TTYIN,4(SP)
5237 021706 000002 RTI ;RETURN
5238 021710 000 .BYTE 0 ;STORAGE FOR ASCII CHAR. TO TYPE
5239 021711 000 .BYTE 0 ;TERMINATOR

5240 021712 000010 8TTYIN: .BLKB 8 ;RESERVE 8 BYTES FOR TTY INPUT
5241 021722 052536 005015 000 8CNTLU: .ASCIZ /"U"<15><12> ;CONTROL "U"
5242 021727 136 006507 000012 8CNTLG: .ASCIZ /"G"<15><12> ;CONTROL "G"
5243 021734 005015 053523 020122 8MSWR: .ASCIZ <15><12>/SWR = /
5244 021742 020075 000
5245 021745 040 047040 053505 8MNEW: .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 $HIOCT
5257
5258 021756 011646 8RDOCT: MOV (SP),=(SP) ;PROVIDE SPACE FOR THE
5259 021760 016666 000004 000002 MOV 4(SP),2(SP) ;INPUT NUMBER
5260 021766 010046 MOV R0,=(SP) ;PUSH R0 ON STACK
5261 021770 010146 MOV R1,=(SP) ;PUSH R1 ON STACK
5262 021772 010246 MOV R2,=(SP) ;PUSH R2 ON STACK
5263 021774 104411 18: RDLIN ;READ AN ASCII LINE
5264 021776 012600 MOV (SP)+,R0 ;GET ADDRESS OF 1ST CHARACTER
5265 022000 005001 CLR R1 ;CLEAR DATA WORD
5266 022002 005002 CLR R2
5267 022004 112046 28: MOV (R0)+,=(SP) ;PICKUP THIS CHARACTER
5268 022006 001412 BEQ 38 ;IF ZERO GET OUT
5269 022010 006301 ASL R1 ;*2
5270 022012 006102 ROL R2 ;*4
5271 022014 006301 ASL R1 ;*8
5272 022016 006102 ROL R2 ;*16
5273 022020 006301 ASL R1 ;*32
5274 022022 006102 ROL R2 ;*64
5275 022024 042716 177770 BIC #'C7,(SP) ;STRIP THE ASCII JUNK
5276 022030 062601 ADD (SP)+,R1 ;ADD IN THIS DIGIT
5277 022032 000764 BR 28 ;LOOP
5278 022034 005726 38: TST (SP)+ ;CLEAN TERMINATOR FROM STACK
5279 022036 010166 MOV R1,12(SP) ;SAVE THE RESULT
5280 022042 010237 022056 MOV R2,$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 ;*TYPOG: MOV      @(SP),-(SP)      ;PICKUP THE MODE
5313 022064 116637 000001 022303 ;*      MOVB     1(SP),%0FILL    ;LOAD ZERO FILL SWITCH
5314 022072 112637 022305 ;*      MOVB     (SP)+,%0MODE+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,%0FILL    ;SET THE ZERO FILL SWITCH
5318 022112 112737 000006 022305 ;*      MOVB     #6,%0MODE+1    ;SET FOR SIX(6) DIGITS
5319 022120 112737 000005 022302 ;*TYPON: MOVB     #5,%0CNT     ;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     %0MODE+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,%0MODE     ;SAVE IT FOR USE
5327 022152 113704 022303 ;*      MOVB     %0FILL,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 ;*      ROL      R5           ;ROTATE MSB INTO "C"
5331 022166 000404 ;*      BR       #8           ;GO DO MSB
5332 022170 006105 ;*      ROL      R5           ;FORM THIS DIGIT
5333 022172 006105 ;*      ROL      R5
5334 022174 006105 ;*      ROL      R5
5335 022176 010503 ;*      MOV      R5,R3
5336 022200 006103 ;*      ROL      R3           ;GET LSB OF THIS DIGIT
5337 022202 105337 022304 ;*      DECB     %0MODE        ;TYPE THIS DIGIT?
5338 022206 100016 ;*      BPL      76           ;BR IF NO
5339 022210 042703 177770 ;*      BIC      #177770,R3    ;GET RID OF JUNK
5340 022214 001002 ;*      RNE      48           ;TEST FOR 0
5341 022216 005704 ;*      TST      R4           ;SUPPRESS THIS 0?
5342 022220 001403 ;*      BEQ      56           ;BR IF YES
5343 022222 005204 ;*      INC      R4           ;DON'T SUPPRESS ANYMORE 0'S
5344 022224 052703 000060 ;*      BIS      #'0,R3       ;MAKE THIS DIGIT ASCII
5345 022230 052703 000040 ;*      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 ;*      DECB     %0CNT        ;COUNT BY 1
5349 022250 003347 ;*      BGT      26           ;BR IF MORE TO DO
5350 022252 002402 ;*      BLT      68           ;BR IF DONE
5351 022254 005204 ;*      INC      R4           ;INSURE LAST DIGIT ISN'T A BLANK
5352 022256 000744 ;*      BR       26           ;GO DO THE LAST DIGIT
5353 022260 012605 ;*      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
5359 022300 0000 ;*      .BYTE   0             ;STORAGE FOR ASCII DIGIT
5360 022301 0000 ;*      .BYTE   0             ;TERMINATOR FOR TYPE ROUTINE
5361 022302 0000 ;*      .BYTE   0             ;OCTAL DIGIT COUNTER

```

```

5362 022303 0000 ;%0FILL: .BYTE 0 ;ZERO FILL SWITCH
5363 022304 000000 ;%0MODE: .WORD 0 ;NUMBER OF DIGITS TO TYPE
5364
5365 ;SBTTL TYPDSS - TYPE DECIMAL, LEADING ZEROES SUPPRESSED
5366
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),%18 ;GET THE NUMBER
5375 022314 012746 022346 ;*      MOV      #18,-(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 ;*      .WORD   0,0
5386
5387 ;SBTTL TYPE NUMERICAL ASCII STRING SUPPRESS LEADING ZEROS
5388
5389 ;*****
5390 ;THIS ROUTINE IS USED TO TYPE AN ASCII NUMBER SUPPRESSING THE
5391 ;LEADING NUMBERS.
5392 ;CALL:
5393 ;*      MOV      #NUMADR,-(SP) ;FIRST ADDRESS OF ASCII STRING
5394 ;*      JSR      PC,##$SUPRS
5395
5396
5397
5398 022352 010046 ;$SUPRS: MOV      R0,-(SP) ;SAVE R0
5399 022354 016600 000004 ;*      MOV      4(SP),R0 ;PICKUP THE POINTER
5400 022360 105710 ;*      TSTB    (R0) ;TERMINATEOR?
5401 022362 001403 ;*      BEQ      28 ;BR IF YES
5402 022364 122720 000060 ;*      CMPB    #'0,(R0)+ ;IS THIS AN ASCII "0" ?
5403 022370 001773 ;*      BEQ      18 ;BR IF YES
5404 022372 005300 ;*      DEC      R0 ;BACKUP BY "1"
5405 022374 010037 022402 ;*      MOV      R0,%36 ;SAVE FOR TYPING
5406 022400 104401 ;*      TYPE     ;GO TYPE
5407 022402 000000 ;*      .WORD   0 ;ASCII 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

```

```

5418 ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
5419 ;*
5420 ;*TOP---(+16)
5421 ;* +2---(+18)
5422 ;* +4---R5
5423 ;* +6---R4
5424 ;* +8---R3
5425 ;*+10---P2
5426 ;*+12---R1
5427 ;*+14---R0
5428
5429 022412 $SAVREG:
5430 022412 010046 MOV R0,=(SP) ;;PUSH R0 ON STACK
5431 022414 010146 MOV R1,=(SP) ;;PUSH R1 ON STACK
5432 022416 010246 MOV R2,=(SP) ;;PUSH R2 ON STACK
5433 022420 010346 MOV R3,=(SP) ;;PUSH R3 ON STACK
5434 022422 010446 MOV R4,=(SP) ;;PUSH R4 ON STACK
5435 022424 010546 MOV R5,=(SP) ;;PUSH R5 ON STACK
5436 022426 016646 000022 MOV 22(SP),=(SP) ;;SAVE PS OF MAIN FLOW
5437 022432 016646 000022 MOV 22(SP),=(SP) ;;SAVE PC OF MAIN FLOW
5438 022436 016646 000022 MOV 22(SP),=(SP) ;;SAVE PS OF CALL
5439 022442 016646 000022 MOV 22(SP),=(SP) ;;SAVE PC OF CALL
5440 022446 000002 RTI
5441
5442 ;*RESTORE R0-R5
5443 ;*CALL:
5444 ;* RESREG
5445 022450 $RESREG:
5446 022450 012666 000022 MOV (SP)+,22(SP) ;;RESTORE PC OF CALL
5447 022454 012666 000022 MOV (SP)+,22(SP) ;;RESTORE PS OF CALL
5448 022460 012666 000022 MOV (SP)+,22(SP) ;;RESTORE PC OF MAIN FLOW
5449 022464 012666 000022 MOV (SP)+,22(SP) ;;RESTORE PS OF MAIN FLOW
5450 022470 012605 MOV (SP)+,R5 ;;POP STACK INTO R5
5451 022472 012604 MOV (SP)+,R4 ;;POP STACK INTO R4
5452 022474 012603 MOV (SP)+,R3 ;;POP STACK INTO R3
5453 022476 012602 MOV (SP)+,R2 ;;POP STACK INTO R2
5454 022500 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
5455 022502 012600 MOV (SP)+,R0 ;;POP STACK INTO R0
5456 022504 000002 RTI
5457
5458 ;SBTTL DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE
5459
5460 ;*****
5461 ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
5462 ;*DECIMAL (ASCII) NUMBER, THE SIGN OF THE BINARY NUMBER MUST BE
5463 ;*POSITIVE.
5464
5465 ;*CALL
5466 ;* MOV #PNTR,=(SP) ;;POINTER TO LOW WORD OF BINARY NUMBER
5467 ;* JSR PC,#$DB2D ;;THE FIRST ADDRESS OF ASCIIZ
5468 ;* RETURN ;;IS ON THE STACK
5469
5470
5471 022506 104413 $DB2D: SAVREG ;;SAVE REGISTERS
5472 022510 016602 000002 MOV 2(SP),R2 ;;PICKUP THE DATA POINTER
5473 022514 012700 022666 MOV #SDECVL,R0 ;;GET ADDRESS OF "$DECVL" STRING

```

```

5474 022520 010066 000002 MOV R0,2(SP) ;;PUT ADDRESS OF ASCIIZ 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,,48 ;;SET UP TO DO 10 CONVERSIONS
5478 022536 012704 022616 MOV #6TNPWR,R4 ;;ADDRESS OF TEN POWER
5479 022542 012705 022620 MOV #6TNPWR+2,R5
5480 022546 005003 19: CLR R3 ;;CLEAR PARTIAL
5481 022550 161401 28: SUB (R4),R1 ;;SUBTRACT TEN POWER
5482 022552 005602 SBC R2
5483 022554 161502 SUB (R5),R2
5484 022556 002402 BLT 36 ;;BR IF TEN POWER TO LARGE
5485 022560 005203 INC R3 ;;ADD 1 TO PARTIAL
5486 022562 000772 BR 26 ;;LOOP
5487 022564 062401 38: 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 HIS #0,R3 ;;CHANGE PARTIAL TO ASCII
5492 022600 110320 MOVB R3,(R0)+ ;;SAVE IT
5493 022602 005327 DEC (PC)+ ;;DONE?
5494 022604 000000 48: WORD 0
5495 022606 001357 BNE 16 ;;BR IF NO
5496 022610 105020 CLR (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 ASCIIZ STRING
5520
5521
5522 ;SBTTL TRAP DECODER
5523
5524 ;*****
5525 ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
5526 ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
5527 ;*OF THE DESIRED ROUTINE, THEN USING THE ADDRESS OBTAINED IT WILL
5528 ;*GO TO THAT ROUTINE.
5529

```

```

5530
5531 022702 010046          $TRAP1: 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          MOVVB  (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
5541                          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
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          $PDOCT ;;CALL=PDOCT     TRAP+12(104412) PEAD AN OCTAL NUMBER FROM TTY
5566 022764 022412          $SAVREG ;;CALL=SAVREG    TRAP+13(104413) SAVE RO-R5 ROUTINE
5567 022766 022450          $RESREG ;;CALL=RESREG    TRAP+14(104414) RESTORE RO-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=MESSAGE   TRAP+17(104417) MESSAGE HANDLER
5574
5575 022776 016106          TY,MSG ;;CALL=TYPMSG    TRAP+20(104420) MESSAGE TYPEOUT ROUTINE
5576
5577 023000 016620          CN,PDY ;;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 LQG 0'S
5584
5585

```

```

5586 .SBTTL POWER DOWN AND UP ROUTINES
5587
5588
5589 ;*****
5590 ;POWER DOWN ROUTINE
5591 023010 012737 023154 000024 $PWRDN: MOV    $#ILLUP,##PWRVEC ;;SET FOR FAST UP
5592 023016 012737 000340 000026  MOV    #340,##PWRVEC+2 ;;PRIO:7
5593 023024 010046          MOV    R0,=(SP)          ;;PUSH R0 ON STACK
5594 023026 010146          MOV    R1,=(SP)          ;;PUSH R1 ON STACK
5595 023030 010246          MOV    R2,=(SP)          ;;PUSH R2 ON STACK
5596 023032 010346          MOV    R3,=(SP)          ;;PUSH R3 ON STACK
5597 023034 010446          MOV    R4,=(SP)          ;;PUSH R4 ON STACK
5598 023036 010546          MOV    R5,=(SP)          ;;PUSH R5 ON STACK
5599 023040 017746 156074          MOV    @SWR,=(SP)        ;;PUSH @SWR ON STACK
5600 023044 010637 023160          MOV    SP,$SAVR6         ;;SAVE SP
5601 023050 012737 023062 000024  MOV    ##PWRUP,##PWRVEC ;;SET UP VECTOR
5602 023060 000776          HALT
5603 BR      =-2                ;;HANG UP
5604
5605 ;*****
5606 ;POWER UP ROUTINE
5607 023062 012737 023154 000024 $PWRUP: MOV    ##ILLUP,##PWRVEC ;;SET FOR FAST DOWN
5608 023070 013706 023160          MOV    $SAVR6,SP         ;;GET SP
5609 023074 005037 023160          CLR    $SAVR6           ;;WAIT LOOP FOR THE TTY
5610 023100 005237 023160 1$: INC    $SAVR6           ;;WAIT FOR THE INC
5611 023104 001375          BNE    1$                ;;OF WORD
5612 023106 012677 156026          MOV    (SP)+,@SWR        ;;POP STACK INTO @SWR
5613 023112 012605          MOV    (SP)+,R5         ;;POP STACK INTO R5
5614 023114 012604          MOV    (SP)+,R4         ;;POP STACK INTO R4
5615 023116 012603          MOV    (SP)+,R3         ;;POP STACK INTO R3
5616 023120 012602          MOV    (SP)+,R2         ;;POP STACK INTO R2
5617 023122 012601          MOV    (SP)+,R1         ;;POP STACK INTO R1
5618 023124 012600          MOV    (SP)+,R0         ;;POP STACK INTO R0
5619 023126 012737 023010 000024  MOV    ##PWRDN,##PWRVEC ;;SET UP THE POWER DOWN VECTOR
5620 023134 012737 000340 000026  MOV    #340,##PWRVEC+2 ;;PRIO:7
5621 023142 104401          TYPE  ;;;REPORT THE POWER FAILURE
5622 023144 023162          $PWRMG: .WORD  $POWER    ;;POWER FAIL MESSAGE POINTER
5623 023146 012716          MOV    (PC)+,(SP)       ;;RESTART AT PWRFL
5624 023150 004244          $PWRAD: .WORD  PWRFL     ;;RESTART ADDRESS
5625 023152 000002          RTI
5626 023154 000000          $ILLUP: HALT            ;;THE POWER UP SEQUENCE WAS STARTED
5627 023156 000776          BR      =-2                ;; BEFORE THE POWER DOWN WAS COMPLETE
5628 023160 000000          $SAVR6: 0                ;; PUT THE SP HERE
5629 023162 005015 047520 042527 $POWER: ,ASCIZ <15><12>"POWER"
5630
5631 ,EVEN
5632

```

```

5633          ,SBTTL FUNCTION SELECTION PROGRAM
5634
5635          ;THIS IS THE FUNCTION SELECTION PROGRAM.
5636          ;ON ENTERING THIS SUB-PROGRAM THE FIRST QUESTION ASKED IS
5637          ;FUNCTION? IN REPLY TYPE IN ONE OF THE FOLLOWING COMMANDS.
5638          ;COMMANDS: CR - CONTROL RESET
5639                   DR - DRIVE RESET
5640                   SK - SEEK
5641                   WR - WRITE
5642                   RD - READ
5643                   WC - WRITE CHECK
5644                   RC - READ CHECK
5645          ;TERMINATE EVERY COMMAND WITH <CR>, FURTHER QUESTIONS (RKBA? , RKDA? ,
5646          ;#WORDS? ) WILL BE ASKED, TYPE IN THE BUS ADDRESS (OCTAL), DISK ADDRESS
5647          ;(OCTAL), AND NUMBER OF WORDS TO BE TRANSFERRED (OCTAL), IF A NON-EXISTENT
5648          ;CYLINDER OR A NON-EXISTENT SECTOR IS TYPED IN, THE QUESTION (RKDA?) WILL
5649          ;BE ASKED AGAIN.
5650
5651          ;IN CASE OF SEEK FUNCTION, SEEK IS DONE BETWEEN A SET OF CYLINDERS GIVEN
5652          ;BY THE USER (CYL1? , CYL2?), IN REPLY TO (CYL1? , CYL2?) TYPE IN THE OCTAL
5653          ;CYLINDER NUMBERS BETWEEN WHICH THE SEEK IS TO BE DONE, SET SWITCH
5654          ;REGISTER BITS <15-13> TO THE DRIVE # ON WHICH SEEK IS TO BE DONE.
5655
5656          ;IN CASE OF A WRITE FUNCTION IF SW 0 IS SET TO 1 THE PROGRAM WILL ASK
5657          ;THE USER FOR THE PATTERN TO BE WRITTEN:
5658          ; PATRN?125252<CR>
5659          ;THE USER SHOULD TYPE IN THE (OCTAL) PATTERN HE WANTS TO WRITE.
5660          ;NOTE THAT THE NUMBER OF WORDS TRANSFERRED SHOULD BE WITHIN THE
5661          ;BOUNDS OF THE SYSTEM.
5662
5663          ;IN CASE OF A WRITE CHECK FUNCTION IF SW0 IS SET TO 1 THE PROGRAM
5664          ;WILL ASK THE USER FOR THE PATTERN TO BE WRITE CHECKED:
5665          ; PATRN?125252<CR>
5666          ;THE USER SHOULD TYPE IN THE (OCTAL) PATTERN TO BE WRITE CHECKED.
5667
5668          ;LOCATIONS "IOBUF0" ONWARDS ARE RESERVED FOR DATA BUFFERS. YOU CAN USE
5669          ;THESE LOCATIONS FOR DOING DATA TRANSFERS IN THIS PROGRAM (OR YOU CAN
5670          ;USE ANY OTHER BUFFER FOR DATA TRANSFER AS LONG AS IT DOES NOT OVERLAY
5671          ;THE PROGRAM).
5672
5673          ;THE SAME FUNCTION IS PERFORMED AGAIN AND AGAIN, THUS PROVIDING
5674          ;A VERY GOOD SCOPE LOOP. IF YOU WANT TO GIVE A NEW FUNCTION PUT SW 3 UP.
5675          ;THE QUESTION (FUNCTION?) WILL BE ASKED AND YOU CAN START ALL OVER AGAIN.
5676          ;IF ON EXECUTING A FUNCTION AN ERROR OCCURS, IT IS REPOTRED , WITH
5677          ;RELEVANT REGISTER CONTENTS GIVEN.
5678
5679
5680
5681
5682
5683
5684
5685
5686
5687
5688
5689
5690
5691
5692
5693
5694
5695
5696
5697
5698
5699
5700
5701
5702
5703
5704
5705
5706
5707
5708
5709
5710
5711
5712
5713
5714
5715
5716
5717
5718
5719
5720
5721
5722
5723
5724
5725
5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744

```

```

;R2 CONTAINS RKCS CONTENTS
;R3 CONTAINS RKDA CONTENTS
;R4,R5 CONTAIN THE CYLINDER #S BETWEEN
;WHICH SEEK IS TO BE DONE.
FUNBEG:
TYPE      ,656          ;;TYPE ASCIZ STRING
BR        646          ;;GET OVER THE ASCIZ
;656:    ,ASCIZ <15><12>/FUNCTION? /
646:
RDLIN

```

```

5689 023220 012600      MOV      (SP)+,R0
5690 023222 112001      MOVBR  (R0)+,R1
5691 023224 120127 000127  CMPB   R1,#'W
5692 023230 001026      BNE    28
5693 023232 121027 000122  CMPB   (R0),#'R
5694 023236 001010      BNE    18
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  98:   MOV    #3,R2
5700 023256 000521      BR     NXTBA
5701 023260 121027 000103  18:   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  28:   CMPB   R1,#'R
5711 023312 001014      BNE    38
5712 023314 121027 000104  CMPB   (R0),#'D
5713 023320 001003      BNE    88
5714 023322 012702 000005  MOV    #5,R2
5715 023326 000475      BR     NXTBA
5716 023330 121027 000103  88:   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  38:   CMPB   R1,#'D
5722 023350 001006      BNE    48
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  48:   CMPB   R1,#'C
5729 023372 001006      BNE    58
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  58:   CMPB   R1,#'S
5736 023414 001266      BNE    FUNBEG
5737 023416 121027 000113  CMPB   (R0),#'K
5738 023422 001263      BNE    FUNBEG
5739 023424 012702 000011  MOV    #11,R2
5740
5741 023430
5742 023430 104401 023436  68:   TYPE    ,678          ;;TYPE ASCIZ STRING
5743 023434 000404      BR     668          ;;GET OVER THE ASCIZ
5744
;678:    ,ASCIZ /CYL1? /

```

```

5745 023446
5746 023446 004737 024032
5747 023452 000766
5748 023454 010004
5749
5750 023456
5751 023456 104401 023464
5752 023462 000404
5753
5754 023474
5755 023474 004737 024032
5756 023500 000766
5757 023502 010005
5758 023504 017700 155430
5759 023510 042700 017777
5760 023514 050004
5761 023516 050005
5762 023520 000466
5763
5764
5765 023522
5766 023522 104401 023530
5767 023526 000404
5768
5769 023540
5770 023540 104412
5771 023542 012637 001476
5772
5773 023546
5774 023546 104401 023554
5775 023552 000404
5776
5777 023564
5778 023564 104412
5779 023566 012600
5780 023570 010001
5781 023572 006201
5782 023574 006201
5783 023576 006201
5784 023600 006201
5785 023602 006201
5786 023604 042701 177400
5787 023610 020127 000312
5788 023614 003354
5789 023616 010001
5790 023620 042701 177760

5791 023624 020127 000013
5792 023630 003346
5793 023632 010003
5794 023634 022702 000015
5795 023640 001416
5796
5797
5798 023642
5799 023642 104401 023650
5800 023646 000405

668: JSR PC,INPT
BR 68
MOV R0,R4

78: TYPE ,698 ;;TYPE ASCIZ STRING
BR 688 ;;GET OVER THE ASCIZ
;;698: ,ASCIZ /CYL2? /
688: JSR PC,INPT
BR 78
MOV R0,R5
MOV @SWR,R0 ;GET DRIVE # FROM SW REG<15-13>
BIC #17777,R0 ;CLR UNWANTED BITS
BIS R0,R4 ;SET DRIVE # IN DSK ADRES
BIS R0,R5
BR EXEC

NXTBA: TYPE ,658 ;;TYPE ASCIZ STRING
BR 648 ;;GET OVER THE ASCIZ
;;658: ,ASCIZ /RKBA? /
648: RDOCT
MOV (SP)+,INDX2

NXTDA: TYPE ,658 ;;TYPE ASCIZ STRING
BR 648 ;;GET OVER THE ASCIZ
;;658: ,ASCIZ /RKDA? /
648: RDOCT
MOV (SP)+,R0
MOV R0,R1
ASR R1
ASR R1
ASR R1
ASR R1
ASR R1
BIC #177400,R1
CMP R1,#312
BGT NXTDA
MOV R0,R1
BIC #177760,R1

CMP R1,#13
BGT NXTDA
MOV R0,R3
CMP #15,R2
BFQ EXEC

NXTWC: TYPE ,658 ;;TYPE ASCIZ STRING
BR 648 ;;GET OVER THE ASCIZ
    
```

```

5801
5802 023662
5803 023662 104412
5804 023664 005416
5805 023666 012637 001502
5806 023672 000401
5807
5808 023674 104415
5809 023676 022702 000011
5810 023702 001005
5811 023704 020403
5812 023706 001402
5813 023710 010403
5814 023712 000401
5815 023714 010503
5816 023716 013777 001476 155536
5817 023724 010377 155534
5818 023730 013777 001502 155522
5819 023736 010277 155514
5820 023742 105777 155510
5821 023746 100375
5822 023750 022702 000001
5823 023754 001401
5824 023756 104423
5825
5826 023760 032777 140000 155470
5827 023766 001006
5828 023770 032777 000010 155142
5829 023776 001737
5830 024000 000137 023172
5831
5832
5833 024004 012737 023674 001110
5834 024012 012737 023674 001106
5835 024020 004737 016162
5836 024024 104030
5837 024026 104415
5838 024030 000757
5839
5840 024032 104412
5841 024034 012600
5842 024036 020027 000312
5843 024042 003007
5844 024044 006300
5845 024046 006300
5846 024050 006300
5847 024052 006300
5848 024054 006300
5849 024056 062716 000002
5850 024062 000207
5851
5852 024064 032777 000001 155046
5853 024072 001416
5854
5855 024074 104401 024102
5856 024100 000404

;;658: ,ASCIZ /#WORDS? /
648: RDOCT
NEG (SP)
MOV (SP)+,INDX4
BR EXEC

EXEC1: CON,RESET ;CLR ERROR, CONTROL RESET
EXEC: CMP #11,R2 ;SEEK FUNCTION?
BNE 28 ;NO
CMP R4,R3 ;IF SEEK, INSERT THE RIGHT
BEQ 38 ;CYLINDER ADDRESS
MOV R4,R3
BR 26
38: MOV R5,R3
28: MOV INDX2,@RKBA
MOV R3,@RKDA
MOV INDX4,@RKWC
MOV R2,@RKCS
18: TSTB @RKCS ;WAIT FOR CNTRL RDY
BPL 18
CMP #1,R2 ;IF IT'S CON RESET FUNCTION
BEQ 48 ;DONT WAIT FOR R/W/S RDY
RESDON ;R/W/S RDY?

48: BIT #140000,@RKCS ;ERROR?
BNE FUNERR ;YES
BIT #SW3,@SWR ;TERMINATE THIS FUNCTION OR REPEAT?
BEQ EXEC ;REPEAT
JMP FUNBEG ;TERMINATE

FUNERR: MOV #EXEC1,@LPERR ;SET UP FOR LUPING
MOV #EXEC1,@LPADR
JSR PC,GTARG
ERROR 30 ;REPORT ERROR
CON,RESET ;CLR ERROR
BR CHSW

INPT: RDOCT
MOV (SP)+,R0
CMP R0,#312
BGT #18
ASL R0
ASL R0
ASL R0
ASL R0
ADD #2,(SP)
18: RTS PC

CHSW0: BIT #SW0,@SWR ;WRITE A PATTERN GIVEN BY USER?
BEQ 18 ;NO
;YES, ASK FOR PATTERN
TYPE ,658 ;;TYPE ASCIZ STRING
BR 648 ;;GET OVER THE ASCIZ
    
```

```
5857          ;ASCIZ /PATRN7/
5858 024112    648:
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    18:      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 ADDR
5869 024150    032737    020000    001230    BIT      #20000,DRIVAD    ;SEE IF ODD
5870 024156    001404    REQ      18
5871 024160    042737    020000    001230    BIC      #20000,DRIVAD    ;MAKE EVEN
5872 024166    000403    BR       28
5873 024170    052737    020000    001230    16:     BIS      #20000,DRIVAD    ;MAKE ODD
5874 024176    013777    001230    155260    28:     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       ;HEADS IN MOTION?
5880 024232    001001    BNE      38
5881 024234    005725    TST      (R5)+
5882 024236    013737    002120    001230    38:     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      #DRIV0,R0        ;TABLE OF AVAIL DRIVES
5887 024260    105710    46:     TSTB    (R0)              ;THIS DRIVE HERE?
5888 024262    001413    BEQ      28
5889 024264    105760    000002    TSTB    2(R0)            ;COMPLEMENT HERE?
5890 024270    001410    BEQ      28
5891 024272    004537    024136    JSR      RS,FCHECK        ;SEE IF F MODEL
5892 024276    000405    BR       28
5893 024300    052710    000002    RIS      #2,(R0)         ;SET SIGN FOR F
5894 024304    052760    000002    000002    BIS      #2,2(R0)        ;BOTH DRIVES
5895 024312    005720    28:     TST      (R0)+
5896 024314    005720    TST      (R0)+
5897 024316    062737    040000    001230    ADD      #40000,DRIVAD    ;NEXT PAIR OF DRIVES
5898 024324    022700    001251    CMP      #DRIV7+1,R0      ;NEXT ACTUL ADDR
5899 024330    003353    BGT      46
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"/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 /ERROR 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 EPRORS/
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						







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