IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZRKL-D-D
PRODUCT NAME:	RK11/RK05 DYNAMIC TEST
DATE CREATED:	DECEMBER, 1976
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	JIM KAPADIA
REVISED BY:	PERVEZ ZAKI TOM SAWYER CHUCK HESS

TABLE OF CONTENTS

1.0 ABSTRACT

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

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

2.0 REQUIREMENTS

2.1 EQUIPMENT

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

2.2 PRELIMINARY PROGRAMS

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

2.3 EXECUTION TIME

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

3.0 STARTING ADDRESS

200 FOR ANY NORMAL MODE OF OPERATION. ALL 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-D

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

DRIVES PRESENT
0
1

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

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

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

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

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

4.2 RKDP DUMP MODE

4.2.1 THE PROGRAM IS LOADED BY THE RKDP MONITOR.

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

4.2.3 THE PROGRAM IDENTIFIES ITSELF AND PRINTS OUT:

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

4.3 RKDP CHAIN MODE

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

'DRIVE 'N' NOT TESTED'

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

4.4 ACT11 MODE

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

5.0 DRIVE SELECTION

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

6.0 SWITCH OPTIONS

IF THE PROGRAM IS BEING RUN ON A SWITCHLESS PROCESSOR (I.E. AN 11/34) THE PROGRAM WILL DETERMINE THAT THE HARDWARE SWITCH REGISTER IS NOT PRESENT AND WILL USE A 'SOFTWARE' SWITCH REGISTER. THE 'SOFTWARE' SWITCH REGISTER IS LOCATED AT LOCATION 176 (8). THE SETTINGS OF THE 'SOFTWARE' SWITCHES ARE CONTROLLED THROUGH A KEYBOARD ROUTINE WHICH IS CALLED BY TYPING A 'CONTROL G'. THE PROGRAM WILL RECOGNIZE THE 'CONTROL G' whenever the program enters the scope routine or begins a new test, the 'SOFTWARE' SWITCH VALUES ARE ENTERED AS AN OCTAL NUMBER IN RESPONSE TO THE PROMPT FROM THE SWITCH ENTRY ROUTINE:

'SWR = NNNNNN NEW ='

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

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

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

6.1 SW<15>

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

6.2 SW<14>

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

6.3 SW<13>

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

6.4 SW<12>

THIS SWITCH ALLOWS THE PROGRAM TO CYCLE FROM THE POINT OF ERROR TO THE PREVIOUS SCOPE STATEMENT. NOTE THAT IN DOING SO ANY INITIALIZATION BEING DONE AT THE BEGINNING OF THE SUBTEST WILL BE DONE AGAIN AND AGAIN. SEE SEC. 6.7 FOR A DIFFERENT KIND OF SCOPE LOOP.

6.5 SW<11>

THIS SWITCH ALLOWS DUMPING OF ALL RK11 REGISTERS ON

ENCOUNTERING AN ERROR.

6.6 SW<10>

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

6.7 SW<09>

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

6.8 SW<08>

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

OCTAL TEST#?

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

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

6.9 SW<06>

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

6.10 SW<05>

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

SKIP PLOTTING REST OF THE POINTS.

6.11 SW<04>

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

6.12 SW<03>

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

6.13 SW<02>

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

6.14 SW<00>

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

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 CONVIENCE OF THE USER. RETRIES ARE DONE WHENEVER AN ERROR OCCURS.

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

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

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

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

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

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

7.1 PERMISSIBLE USER PROGRAM MODIFICATIONS

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

7.1.1

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

7.1.2

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

7.1.3

INPUT/OUTPUT DATA BUFFERS (FROM WHERE DATA TRANSFERS WILL BE DONE TO AND FROM THE DISK) CAN BE RELOCATED TO ANYWHERE IN THE 28K OF MEMORY (DO NOT OVERLAY THE PROGRAM). THIS CAN BE DONE BY CHANGING THE CONTENTS

OF LOCATIONS 'PBUF0' AND 'PBUF1' TO THE STARTING ADDRESSES OF THE TWO USER SELECTED BUFFERS. IT SHOULD BE NOTED THAT EACH OF THE TWO BUFFERS SHOULD BE 768 (DECIMAL) WORD LONG.

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

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

7.1.6 IN TEST 10, IF YOU WANT TO WRITE AND CHECK CYLINDERS 127 AND 128 WITH PATTERNS OTHER THAN THE 12 USED, CHANGE ANY OR ALL OF THE 12 POINTERS 'SP1' THROUGH 'SP12' TO CONTAIN PATTERNS YOU WANT.

8.0 SEEK TIMER & GRAPHS

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

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

IF YOU WANT TO TIME SEEK BETWEEN ANY OTHER SET OF

CYLINDERS, YOU CAN DO BY FOLLOWING THE INSTRUCTIONS AT LOCATION 'SOAD' IN LISTINGS. SEE SEC 7.1

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

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

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

9.0 FUNCTION SELECTION PROGRAM

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

STARTING ADDRESS=210

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

FUNCTION?

THE REPLY SHOULD BE:

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

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

FOLLOWING QUESTIONS APPEAR:

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

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

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

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

THE BUFFER SIZE AVAILABLE.

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

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

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

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

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

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

10.0 ERROR INFORMATION

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

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

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

11.0 UNEXPECTED TIMEOUTS AND RK11 INTERRUPTS

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

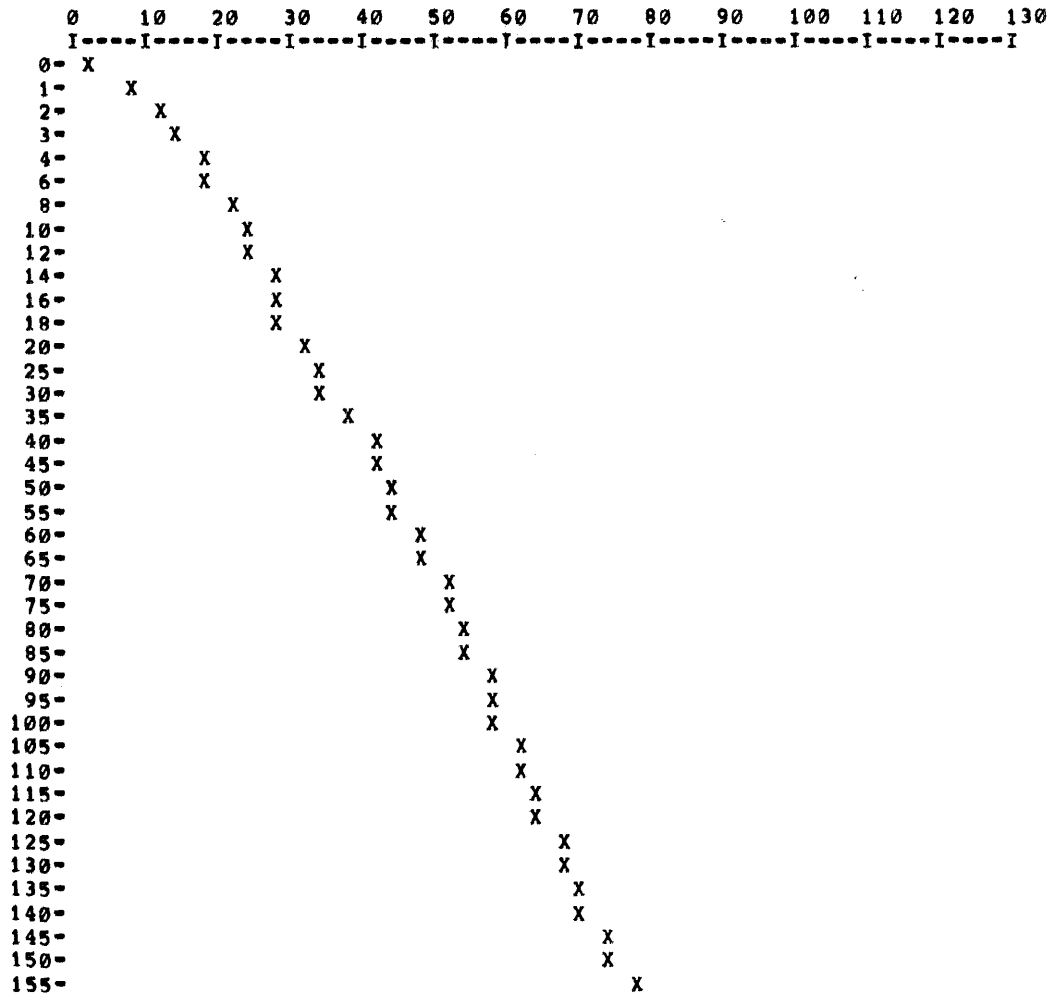
13.0 SAMPLE GRAPH AND SEEK TIMER OUTPUTS

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

SEEK TIME SCALE FACTOR=0.01 MILI SECS

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

```
CYLS:0-16
  FRWRD             REVRSE

    100      3135     100      3135

CYLS:0-32
  FRWRD             REVRSE

    100      3795     100      3795

CYLS:0-100
  FRWRD             REVRSE

    100      5775     100      5775

X AXIS - SEEK TIME - MILI SECS          **SAMPLE OUTPUT**
Y AXIS - CYLINDER SEEKED FROM 0
```




160-
165-
170-
175-
180-
185-
190-
195-
200-
202-

```
                        .TITLE  MAINDEC-11-DZRKL-D
                        ;*COPYRIGHT (C) 1974,1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY JIM KAPADIA
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                        ;*
                        ;*JANUARY 1975
                        ;*
                        ;*REVISED MARCH 1976 BY TOM SAWYER
                        ;*REVISED BY CHUCK HESS, AUGUST, 1976

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


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



```
                        .SBTTL  BASIC DEFINITIONS

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

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

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

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

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

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

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                        BIT15=  100000
                        BIT14=  40000
                        BIT13=  20000
                        BIT12=  10000
                        BIT11=  4000
                        BIT10=  2000
                        BIT09=  1000
                        BIT08=  400
                        BIT07=  200
                        BIT06=  100
                        BIT05=  40
                        BIT04=  20
                        BIT03=  10
                        BIT02=  4
                        BIT01=  2
                        BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

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

                        .SBTTL  TRAP CATCHER
```



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

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

                        ;;*****************************************************************
                        ;HOOKS REQUIRED BY ACT11
                        $SVPC=.                 ;SAVE PC
                        .=46
000046  015254          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                        .=52
000052  000000          .WORD   0               ;;2)SET LOC.52 TO ZERO
                        .=@#$VPC                ;; RESTORE PC
```

```
                                .SBTTL  COMMON TAGS

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

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



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

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

                                ;TAGS AND GENERAL DATA AREA

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


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

                        001226          .EVEN


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

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


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


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


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

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

001404  026362                  PBUF0:  IOBUF0          ;POINTER TO THE STARTING ADDRESS OF THE
                                                        ;BUFFER USED TO READ INTO FROM DIBK.
001406  031362                  PBUF1:  IOBUF1          ;POINTER TO STARTING ADDRESS OF BUFFER
                                                        ;IN WHICH PATTERNS ARE GENERATED. (WRITING
                                                        ;IS DONE FROM THIS BUFFER)
001410  000000                  BUFLG0: .WORD   0       ;FLAG FOR 'IOBUF0'
001412  000000                  BUFLG1: .WORD   0       ;FLAG FOR 'IOBUF1'


001414  010032                  PAT0:   PTGEN0          ;ADRES OF 'PATRN GENERATOR 0'
                                                        ;ROUTINE
001416  010114                  PAT1:   PTGEN1          ;ADRES OF 'PATRN GENERATOR 1'

001420  010216                  PAT2:   PTGEN2          ;ADRES OF 'PATRN GENRATOR 2'

001422  010260                  PAT3:   PTGEN3          ;ADRES OF 'PATRN GENRATOR 3'

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

001430  000000                  PGSUBR: .WORD   0

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

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

001436  000000                  WRDCNT: .WORD   0       ;CONTAINS WORD COUNT

001440  000000                  WDSKAD: .WORD   0       ;CONTAINS DISK ADRES

001442  000000                  WBUSAD: .WORD   0       ;CONTAINS BUS ADRES

001444  000000                  WWRDCN: .WORD   0       ;CONTAINS WORD COUNT

001446  000000                  BUFNO:  .WORD   0       ;CONTAINS STARTING ADRES

001450  000000                  ADRES:  .WORD   0       ;OF A BUFFER

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


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



```
001464  177412                  RKDA:   177412
001466  177416                  RKDB:   177416

001470  000200                  RKPRI:  200             ;CONTAINS THE CPU LEVEL (4) AT WHICH
                                                        ;RK11 NORMALLY INTERRUPTS.  THIS WORD
                                                        ;SHOULD BE CHANGED IF RK11 IS DESIGNATED
                                                        ;A BR LEVEL OTHER THAN 5.EXP: IF IT
                                                        ;IS CHANGED TO 6, THE CPU LEVEL WOULD
                                                        ;BE 1 LESS (5) & HENCE THIS WORD
                                                        ;SHOULD BE 240 (BIT POSITIONS ARE
                                                        ;IDENTICAL TO THE PRIORITY BITS IN PSW)
001472  000220                  RKVEC:  220             ;CONTAINS THE NORMAL VECTOR ADDRESS
                                                        ;TO WHICH THE RK11 INTERRUPTS.  IF THE
                                                        ;VECTOR ADDRESS HAS BEEN CHANGED, MODIFY
                                                        ;THIS WORD.


001474  000000                  INDX1:  0               ;GENERAL INDEX REGISTERS
001476  000000                  INDX2:  0
001500  000000                  INDX3:  0
001502  000000                  INDX4:  0

001504  000000                  ERCNT1: 0               ;GENERAL REGISTERS
001506  000000                  ERCNT2: 0               ;GENERAL REGISTERS
001510  000000                  ERCNT3: 0               ;GENERAL REGISTERS
001512  000000                  ERCNT4: 0
001514  000000                  ERCNT5: 0
001516  000000                  ERCNT6: 0
001520  000000                  ERCNT7: 0
001522  000000                  ERCNT8: 0


                                ;*THE FOLLOWING TABLE CONTAINS THE CYLINDERS BETWEEN WHICH THE SEEKS WILL BE
                                ;*TIMED. THEY HAVE BEEN SELECTED TO GIVE SOME TYPICAL SEEKS TIMES FOR THE
                                ;*3 SEEK SPEEDS. IF FOR  ANY REASON YOU WANT  TO TIME SEEKS BETWEEN ANY
                                ;*OTHER SET OF CYLINDERS, MAKE CHANGES IN THE CORRESPONDING SEEK CYLINDER
                                ;*ADDRESSES.

                                ;*OUTER CYLINDER ADDRESS, FROM WHERE SEEK WILL BE DONE
001524  000000                  SOAD:   0               ;CYLINDER 0
001526  000000                          0               ;  "  0
001530  013140                          13140           ;  "  179
001532  000000                          0               ;  "  0
001534  000000                          0               ;  "  0
001536  000000                          0               ;  "  0
001540  000000                          0               ;  "  0

                                ;*INNER ADDRESS, TO WHICH SEEK WILL BE DONE
001542  014500                  SIAD:   14500           ;CYLINDDER 202, LAST
001544  000040                          40              ;  "  1
001546  013240                          13240           ;  "  181
001550  000140                          140             ;  "  3
001552  001000                          1000            ;  "  16
001554  002000                          2000            ;  "  32
```

```
                                        6200            ; "  100
                                ;FOLLOWING POINTERS ARE USED TO TRANSFER CONROL TO THE
                                ;TEST SELECTED BY USING SW 8,  IF ANY MORE TESTS ARE
                                ;ADDED TO THIS PROGRAM ADDITIONAL POINTERS SHOULD BE INSERTED.
001560  004206                  PT1:    TST1+2
001562  004552                  PT2:    TST2+2
001564  005050                  PT3:    TST3+2
001566  005540                  PT4:    TST4+2
001570  006546                  PT5:    TST5+2
001572  007304                  PT6:    TST6+2
001574  010364                  PT7:    TST7+2
001576  012122                  PT10:   TST10+2
001600  012664                  PT11:   TST11+2


                                ;MESSAGES & ASCII STRINGS
001602  005015  044523  000116  MSG1:   .ASCIZ  <15><12>/SIN/

001610  005015  045523  000105  MSG2:   .ASCIZ  <15><12>/SKE/

001616  005015  042524  052123  MSG3:   .ASCIZ  <15><12>/TEST # ABORTED!/
001624  021440  040440  047502
001632  052122  042105  000072

001640  005015  051120  043517  MSG4:   .ASCIZ  <15><12>/PROG ABORTED/
001646  040440  047502  052122
001654  042105     000

001657     015  051012  040505  MSG5:   .ASCIZ  <15><12>/READ HDRS OK FROM CYLB ABOVE/
001664  020104  042110  051522
001672  047440  020113  051106
001700  046517  041440  046131
001706  020102  041101  053117
001714  000105

001716  054105  041520  042124  MSG6:   .ASCIZ  /EXPCTD HDR# /
001724  044040  051104  020075
001732     000

001733     040  050040  036503  MSG7:   .ASCIZ  /  PC# /
001740  000040

001742  005015  047103  051124  MSG10:  .ASCIZ  <15><12>/CNTRL RDY DIDN'T SET/
001750  020114  042122  020131
001756  044504  047104  052047
001764  051440  052105     000

001771     123  041505  051124  MSG11:  .ASCIZ  /SECTR  EXPC P=HDR  RECV P=HDR/
001776  020040  054105  041520
002004  050040  044055  051104
002012  020040  042522  053103
002020  050040  044055  051104
002026     000
```



```
002027     015  051012  053457  MSG12:  .ASCIZ  <15><12>"R/W/S RDY NOT SET"
002034  051457  051040  054504
002042  047040  052117  051440
002050  052105     000

002053     040  052040  054522  MSG13:  .ASCIZ  /  TRY #!/
002060  021440  000072

002064  005015  051104  053111  MSG14:  .ASCIZ  <15><12>/DRIVE /
002072  020105     000

002075     040  020040          BLNK13: .ASCII  /   /
002100     040                  BLNK10: .ASCII  / /
002101     040                  BLNKS9: .ASCII  / /
002102     040                  BLNKS8: .ASCII  / /
002103     040                  BLNKS7: .ASCII  / /
002104     040                  BLNKS6: .ASCII  / /
002105     040                  BLNKS5: .ASCII  / /
002106     040                  BLNKS4: .ASCII  / /
002107     040                  BLNKS3: .ASCII  / /
002110     040                  BLNKS2: .ASCII  / /
002111     040     000          BLNKS1: .ASCIZ  / /

        002114                          .EVEN
002114  000000                  FDRIVE: 0
002116  000000                  FDRVE1: 0
002120  000000                  DRHOLD: 0
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


002122                          $ERRTB:


                                        ;ERROR ITEMS TABLE
                                        ;
                                        ;
                                        ;
                                ;ITEM   1

002122  024250                          EM1     ;CNTRL RDY DIDN'T SET AFTER SEEK
002124  025414                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002126  026250                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002130  000000                          0

                                ;ITEM   2

002132  024307                          EM2     ;SIN ON SEEK
002134  025414                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002136  026250                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002140  000000                          0

                                ;ITEM   3

002142  024323                          EM3     ;DRE ON SEEK
002144  025414                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002146  026250                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002150  000000                          0

                                ;ITEM   4

002152  024337                          EM4     ;'ERR' ON SEEK
002154  025414                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002156  026250                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002160  000000                          0

                                ;ITEM   5

002162  024355                          EM5     ;'DRU' ON SEEK; PUT DRIVE ON 'LOAD' BACK TO 'RUN'
002164  025414                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002166  026250                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002170  000000                          0
```



```
                                ;ITEM   6

002172  024424                          EM6     ;R/W/S RDY NOT SET AFTER SEEK
002174  025414                          DH1     ;PC     RKCS    RKER    RKDS    RKDA
002176  026250                          DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
002200  000000                          0

                                ;ITEM   7

002202  024461                          EM7     ;SIN ON WRITE FMT
002204  025512                          DH7     ;PC     RKCS    RKER    RKDS    RKDA    CYLINDER
002206  026264                          DT7     ;$ERRPC $REG0   $REG1   $REG2   $REG3   $REG4
002210  000000                          0

                                ;ITEM   10

002212  024500                          EM10    ;'ERR' ON DOING WRITE FMT
002214  025512                          DH7     ;PC     RKCS    RKER    RKDS    RKDA    CYLINDER
002216  026264                          DT7     ;$ERRPC $REG0   $REG1   $REG2   $REG3   $REG4
002220  000000                          0

                                ;ITEM   11

002222  024531                          EM11    ;SIN ON READ FMT
002224  025567                          DH11    ;PC     RKCS    RKER    RKDS    RKDA;
                                                ;DRV#   CYL     SUR     SEC
002226  026302                          DT11    ;$ERRPC $REG0   $REG1   $REG2
                                                ;$REG4  $REG5   $REG6   $REG7
002230  000000                          0

                                ;ITEM   12

002232  024547                          EM12    ;'ERR' ON READ FMT
002234  025512                          DH7     ;PC     RKCS    RKER    RKDS    RKDA    CYLINDER
002236  026264                          DT7     ;$ERRPC $REG0   $REG1   REG2    $REG3   $REG4
002240  000000                          0

                                ;ITEM   13

002242  024571                          EM13    ;WRONG HEADERS FROM 'SEC #
002244  025664                          DH13    ;SECTOR #       HEADER RECVD
002246  000000                          0
002250  015402                          ESR13   ;USE THIS SUBROUTINE FOR TYPING OUT ERROR DATA

                                ;ITEM   14

002252  024620                          EM14    ;ERROR ON IMPLIED SEEK FROM CYLA TO CYLB
002254  025703                          DH14    ;PC     CYLA    CYLB    RKER    RKDS    TRY#
002256  026264                          DT7     ;$ERRPC $REG0   $REG1   $REG2   $REG3   $REG4
002260  000000                          0

                                ;ITEM   15

002262  024673                          MS15    ;READ WRONG HDRS FROM CYLB ABOV
002264  025664                          DH13    ;SEC#   HEADER RECVD
002266  000000                          0
```

```
  602  002270  015310                    ESR15   ;GO TO "ESR15" FOR TYPING OUT
  603
  604                                  ;ITEM   16
  605
  606  002272  024735                    EM16    ;READ WRONG, FIRST WORD FROM SECTOR 0, 'CYLB' (ON IMPLIED SEEK FROM CYLA
  607  002274  025762                    DH16    ;PC     CYLA    CYLB    EXPCT   RECVD   TRY#
  608  002276  026264                    DT7     ;$ERRPC $REG0   $REG1   $REG2   $REG3   $REG4
  609  002300  000000                    0
  610
  611                                  ;ITEM   17
  612
  613  002302  025042                    MS17    ;READ FIRST WORD FROM SECTOR 1, 'CYLB' ABOVE
  614  002304  026037                    DH17    ;PC     CYLB    EXPCT   RECVD
  615  002306  026324                    DT17    ;$ERRPC $REG0   $REG1   $REG2
  616  002310  000000                    0
  617
  618                                  ;ITEM   20
  619
  620  002312  025110                    EM20    ;READ WRONG HEADER ON IMPLIED SEEK FROM 'CYLA' TO 'CYLB'
  621  002314  025664                    DH13    ;SECTOR # HEADER RECVD
  622  002316  000000                    0
  623  002320  015456                    ESR20   ;USE THIS SUBROUTINE FOR TYPING OUT ERROR DATA
  624
  625
  626                                  ;ITEM   21
  627
  628  002322  025176                    EM21    ;EROR ON DOING WRITE ON DSK
  629  002324  025414                    DH1     ;PC     RKCS    RKER    RKDS    RKDA
  630  002326  026250                    DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
  631  002330  000000                    0
  632
  633                                  ;ITEM   22
  634
  635  002332  025232                    EM22    ;SIN ON DOING WRITE
  636  002334  025414                    DH1     ;PC     RKCS    RKER    RKDS    RKDA
  637  002336  026250                    DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
  638  002340  000000                    0
  639
  640                                  ;ITEM   23
  641
  642  002342  025255                    EM23    ;HE ON DOING READ
  643  002344  025567                    DH11    ;PC     RKCS    RKER    RKDS    RKDA:
  644                                             ;DRV#   CYL     SUR     SEC
  645  002346  026302                    DT11    ;$ERRPC $REG0   $REG1   $REG2
  646                                             ;$REG4  $REG5   $REG6   $REG7
  647  002350  000000                    0
  648
  649                                  ;ITEM   24
  650
  651  002352  025276                    EM24    ;CSE ON READ
  652  002354  026075                    DH24    ;PC     TRY#    RKCS    RKER    RKDS    RKDA:
  653                                             ;DRV#   CYL     SUR     SEC
  654  002356  026336                    DT24    ;$ERRPC $REG10  $REG0   $REG1   $REG2
  655                                             ;$REG4  $REG5   $REG6   $REG7
  656  002360  000000                    0
  657
```



```
  658                                  ;ITEM   25
  659
  660  002362  025312                    EM25    ;DATA ERROR ON READ FROM DISK ADDRESS
  661  002364  026202                    DH25    ;WORD#  EXPCT   RECVD   CYL     SUR     SEC
  662  002366  000000                    0
  663  002370  015544                    ESR25   ;USE THIS ROUTINE FOR ERROR REPORTING
  664
  665                                  ;ITEM   26
  666
  667  002372  025354                    EM26    ;HE ON WRT CHK
  668  002374  025567                    DH11    ;PC     RKCS    RKER    RKDS    RKDA:
  669                                             ;DRV#   CYL     SUR     SEC
  670  002376  026302                    DT11    ;$ERRPC $REG0   $REG1   $REG2
  671                                             ;$REG4  $REG5   $REG6   $REG7
  672  002400  000000                    0
  673
  674                                  ;ITEM   27
  675
  676  002402  025372                    EM27    ;WRT CHK EROR
  677  002404  026075                    DH24    ;PC     TRY#    RKCS    RKER    RKDS    RKDA:
  678                                             ;DRV#   CYL     SUR     SEC
  679  002406  026336                    DT24    ;$ERRPC $REG10  $REG0   $REG1   $REG2
  680                                             ;$REG4  $REG5   $REG6   $REG7
  681  002410  000000                    0
  682
  683                                  ;ITEM   30
  684
  685  002412  025407                    EM30    ;ERROR
  686  002414  025414                    DH1     ;PC     RKCS    RKER    RKDS    RKDA
  687  002416  026250                    DT1     ;$ERRPC $REG0   $REG1   $REG2   $REG3
  688  002420  000000                    0
  689
  690
  691
  692
  693
  694
```

```
                                        ;THIS IS THE HANDLER FOR UNEXPECTED TIME OUT, PRESSING CONTINUE WILL
                                        ;RESTART THE PROGRAM.
                                        
                                        
002422  011600                  BADTMO: MOV     (SP),R0 ;SAVE PC WHERE TIME OUT OCCURED
002424  005740                          TST     -(R0)
002426  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
002430  104401  002436                  TYPE    ,65$            ;;TYPE ASCIZ STRING
002434  000407                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/TIMOUT:PC=/
002454                          64$:
002454  010046                          MOV     R0,-(SP)        ;SET UP FOR TYPING OUT PC
002456  104402                          TYPOC                   ;GO TYPE OUT OCTAL PC
002460  000000                          HALT
                                
002462  000005                  START:  RESET                   ;CLEAR THE BUS
                                ,SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002464  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
002470  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
002472  022706  001140                  CMP     #SWR,R6 ;;DONE?
002476  001374                          BNE     .-6             ;;LOOP BACK IF NO
002500  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
002504  012737  016732  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
002512  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
002520  012737  017106  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
002526  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
002534  012737  022616  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002542  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
002550  012737  022724  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
002556  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
002564  005037  001204                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002570  112737  000001  001115          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002576  012737  002576  001106          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002604  012737  002604  001110          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
002612  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002616  012737  002652  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002624  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002632  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002640  022777  177777  176272          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002646  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
002650  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
002652  012716  002660          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002656  000002                          RTI
002660  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002666  012737  000174  001142          MOV     #DISPREG,DISPLAY
002674  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR
                                
002700  004737  020536                  JSR     PC,$TKINT       ;INITIALIZE THE TTY HANDLER
                                ,SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
```



```
002704  005227  177777                  INC     #-1             ;;FIRST TIME?
002710  001043                          BNE     67$             ;;BRANCH IF NO
002712  104401  002750                  TYPE    ,68$            ;;TYPE ASCIZ STRING
                                ,SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
002716  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
002722  001006                          BNE     69$             ;;BRANCH IF YES
002724  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
002732  001005                          BNE     70$             ;;BRANCH IF NO
002734  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
002736  000403                          BR      70$
002740  112737  000001  001134  69$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
002746                          70$:
002746  000424                          BR      67$             ;;GET OVER THE ASCIZ
                                ;;68$:  .ASCIZ  <CRLF>/RK11 DYNAMIC TEST/<15><12>/MAINDEC-11-DZRKL-D/<CRLF>
003020                          67$:
003020  105737  001216          START1: TSTB    FFUNC           ;FUNCTION PROGRAM SELECTED?
003024  001404                          BEQ     7$              ;NO
003026  105037  001216                  CLRB    FFUNC           ;YES, CLEAR THE FLAG
003032  000137  023106                  JMP     @#FUNBEG        ;GO TO 'FUNTION SELECTION PROGRAM'
003036  012700  001220          7$:     MOV     #XXDPMD,R0      ;CLEAR FLAGS FROM
003042  105020                  5$:     CLRB    (R0)+           ;'XXDPMD' TO 'DRIVAD'
003044  020027  001232                  CMP     R0,#DRIVAD+2
003050  001374                          BNE     5$
003052  012701  177770                  MOV     #-10,R1
003056  042720  000003          6$:     BIC     #3,(R0)+        ;CLEAR BIT 0'S IN 'DRIVE
003062  005201                          INC     R1              ;PRESENT' FLAGS.
003064  001374                          BNE     6$
                                
                                ;THE FOLLOWING CODE FINDS OUT THE PROGRAM CONTROL MODE:
                                ;PAPER TAPE (MANUAL), ACT11, RKDP CHAIN OR DUMP
                                
003066  122737  000002  000041          CMPB    #2,41           ;LOADED FROM AN RK05 ?
003074  001160                          BNE     ST2             ;BR IF NOT
003076  013737  000040  001220          MOV     40,XXDPMD       ;GET DEVICE INDICATOR AND DRIVE ADDRESS OF
                                                                ;LOADING RK05
003104  122737  000010  001220          CMPB    #10,XXDPMD      ;DRIVE ADDRESS 7 OR LESS ?
003112  101002                          BHI     2$              ;BR IF YES
003114  105037  001220                  CLRB    XXDPMD          ;DRIVE ZERO LOADED THE PROGRAM
003120  005737  000042          2$:     TST     42              ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
003124  001424                          BEQ     3$              ;BR IF NEITHER
003126  104401  003134                  TYPE    ,65$            ;;TYPE ASCIZ STRING
003132  000413                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/NOT TESTING DRIVE /
003162                          64$:
003162  005046                          CLR     -(SP)           ;CLEAR WORD ON STACK
003164  113716  001220                  MOVB    XXDPMD,(SP)     ;GET DRIVE ADDRESS
003170  104403                          TYPOS                   ;TYPE THE ADDRESS
003172     001                          .BYTE   1               ;ONLY 1 CHARACTER
003173     000                          .BYTE   0               ;SUPRESS LEADING ZEROS
003174  000520                          BR      ST2             ;GET NUMBER OF DRIVES
003176  005227  177777          3$:     INC     #-1             ;FIRST TIME THROUGH HERE ?
003202  001115                          BNE     ST2             ;BR IF NOT FIRST TIME
003204  104401  003212                  TYPE    ,67$            ;;TYPE ASCIZ STRING
003210  000411                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <15><12>/TO TEST DRIVE /
003234                          66$:
```

```
                                        CLR     -(SP)           ;CLEAR WORD ON THE STACK
003234  005046
003236  113716  001220                MOVB    XXDPMD,(SP)     ;GET DRIVE ADDRESS
003242  104403                        TYPOS                   ;TYPE THE DRIVE ADDRESS
003244     001                        .BYTE   1               ;ONLY 1 CHARACTER
003245     000                        .BYTE   0               ;SUPRESS LEADING ZEROS
003246  104401  003254                TYPE    ,69$            ;;TYPE ASCIZ STRING
003252  000431                        BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  / HALT PROGRAM, REMOVE RKDP PACK AND REPLACE IT/<15><12>
003336                          68$:
003336  104401  003344                TYPE    ,71$            ;;TYPE ASCIZ STRING
003342  000435                        BR      70$             ;;GET OVER THE ASCIZ
                                ;;71$:  .ASCIZ  /WITH A WORK PACK, CLEAR LOCATION 40, AND RESTART PROGRAM/
003436                          70$:

003436  012737  002422  000004  ST2:    MOV     #BADTMO,ERRVEC  ;SET TIMEOUT VECTOR FOR
                                                                ;UNEXPECTED TIME OUT

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

003444  104401  003452                TYPE    ,65$            ;;TYPE ASCIZ STRING
003450  000411                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/DRIVES PRESENT/
003474                          64$:
003474  105037  001224                CLRB    DRIVS           ;INITIALIZE NO. OF DRVS PRESENT
003500  005001                        CLR     R1
003502  012702  001232                MOV     #DRIV0,R2
003506  005003                        CLR     R3              ;INITIALIZE COUNT TO 0
003510  005737  001220          1$:   TST     XXDPMD          ;LOADED FROM AN RK05 ?
003514  001403                        BEQ     6$              ;BR IF NOT
003516  120337  001220                CMPB    R3,XXDPMD       ;CHECKING THE LOAD DRIVE ?
003522  001411                        BEQ     2$              ;BR IF YES
003524  010177  175734          6$:   MOV     R1,@RKDA        ;ADRES A DRIVE
003530  105777  175716                TSTB    @RKDS           ;IS IT PRESENT?
003534  100004                        BPL     2$              ;NO, BRANCH
003536  105237  001224                INCB    DRIVS           ;INCREMENT TOTAL # OF DRVS
003542  052712  000001                BIS     #1,(R2)         ;SET FLAG INDICATING THIS DRV PRSNT
003546  005722                  2$:   TST     (R2)+
003550  005203                        INC     R3              ;INCREMENT COUNT
003552  062701  020000                ADD     #20000,R1       ;ADRES THE NXT DRV
                                                                ;CHKD ALL 8 DRIVES?
003556  001354                        BNE     1$              ;IF NOT, GO CHK IF NEXT DRV PRSNT

003560  004737  024164                JSR     PC,SIZEF        ;FIND WHICH ARE FS
003564  105737  001224                TSTB    DRIVS           ;WERE ANY DRIVES FOUND?
003570  001010                        BNE     3$              ;YES, BRANCH
003572  104401  003600                TYPE    ,67$            ;;TYPE ASCIZ STRING
003576  000403                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  / NONE/
003606                          66$:
003606  000137  015172                JMP     $EOP            ;IF NONE WERE FOUND, GO
                                                                ;TO THE END OF PROGRAM
003612  005002                  3$:   CLR     R2              ;DRIVE NUMBER
003614  012700  001232                MOV     #DRIV0,R0       ;TABLE OF AVAIL DRIVES
003620  105710                  5$:   TSTB    (R0)            ;DRIVE HERE?
```



```
003622  001414                        BEQ     4$              ;NO
003624  104401                        TYPE
003626  001213                        $CRLF
003630  010246                        MOV     R2,-(SP)        ;PUSH NO ON THE STACK
003632  104403                        TYPOS                   ;TO TYPE OCTAL NO.
003634     001                        .BYTE   1               ;TYPE 1 DIGIT, SUPRESS LDG 0'S
003635     000                        .BYTE   0
003636  032710  000002                BIT     #2,(R0)         ;IS IT RK05F?
003642  001404                        BEQ     4$              ;NO
003644  104401  003652                TYPE    ,69$            ;;TYPE ASCIZ STRING
003650  000401                        BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  /F/
003654                          68$:
003654  005202                  4$:   INC     R2              ;POINT TO NEXT DRIVE #
003656  005720                        TST     (R0)+           ;NEXT DRIVE IN TABLE
003660  020027  001251                CMP     R0,#DRIV7+1     ;ALL DONE?
003664  002755                        BLT     5$              ;NO, CHECK REST

                                ;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.

003666  012737  001232  001226  ST3:    MOV     #DRIV0,DRVPTR
003674  105037  001223                CLRB    DRVDON
003700  005037  001230                CLR     DRIVAD
003704  105037  001102          NXTDRV: CLRB    $TSTNM          ;RESET TEST NUMBER TO 1
003710  005037  001112                CLR     $ERTTL          ;CLEAR ERROR COUNT FOR THIS DRIVE
003714  013701  001226                MOV     DRVPTR,R1
003720  032721  000001          1$:   BIT     #1,(R1)+        ;IS THIS DRIVE PRESENT?
003724  001005                        BNE     2$              ;YES, BRANCH
003726  020127  001252                CMP     R1,#DRIV7+2     ;CHECKED THE WHOLE LIST?
003732  001372                        BNE     1$              ;NO
003734  000137  015172                JMP     $EOP            ;YES, EXIT
003740  010137  001226          2$:   MOV     R1,DRVPTR       ;NO, GO AHEAD
003744  014104                        MOV     -(R1),R4        ;GET DRIVE NO. TO BE TESTED
003746  005037  002116                CLR     FDRVE1
003752  005037  002114                CLR     FDRIVE          ;SHOWS F IF -1
003756  032704  000002                BIT     #2,R4           ;RK-05F?
003762  001410                        BEQ     7$              ;NO
003764  005237  002116                INC     FDRVE1          ;SHOWS F
003770  032704  020000                BIT     #20000,R4       ;EVEN DRIVE?
003774  001003                        BNE     7$              ;NO
003776  012737  177777  002114        MOV     #-1,FDRIVE      ;RK05F AND EVEN
004004  042704  000003          7$:   BIC     #3,R4
004010  010437  001230                MOV     R4,DRIVAD       ;SET UP DRIVE ADRES
004014  104401  002064                TYPE    ,MSG14
004020  000241                        CLC
004022  006104                        ROL     R4              ;TYPE OUT THE DRIVE NO.
004024  006104                        ROL     R4
004026  006104                        ROL     R4
004030  006104                        ROL     R4
004032  010446                        MOV     R4,-(SP)
004034  104403                        TYPOS
004036     001                        .BYTE   1
004037     000                        .BYTE   0
```

```
                                TST     FDRVE1          ;RK=05F?
                                BEQ     6$              ;NO
                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  /F/
                        64$:
                        6$:
                        ;IF SW 8 IS SET THEN FIND OUT WHICH TEST NUMBER IS
                        ;SELECTED AND JUMP TO THAT TEST.

                                CLRB    LUPSW           ;CLEAR FLAG INDICATING SW8 SET
                                BIT     #SW8,@SWR       ;SW 8 SET?
                                BEQ     TST1            ;NO, BRANCH
                        5$:
                                TYPE    ,67$            ;;TYPE ASCIZ STRING
                                BR      66$             ;;GET OVER THE ASCIZ
                        ;;67$:  .ASCIZ  <15><12>/OCTAL TEST#?/
                        66$:
                                RDOCT
                                MOV     (SP)+,R0
                                BEQ     5$
                                CMP     R0,#11          ;CHECK TYPED IN TEST #
                                BGT     5$              ;IS LEGAL, IF NOT ASK
                                MOVB    R0,$TSTNM
                                DEC     R0              ;FORM POINTERS FOR THE TEST #
                                ASL     R0
                                MOV     PT1(R0),$LPADR  ;ADJUST POINTERS FOR SCOPE
                                MOV     $LPADR,$LPERR   ;LOOP, ETC.
                                INCB    LUPSW           ;SET FLAG INDICATING TEST #
                                                        ;SELECTED
                                JMP     @$LPADR         ;GO TO THE TEST SELECTED



                        ;ON RECOVERY FROM POWER FALIURE RETURN HERE

                        PWRFL:  CLR     R0
                                CLR     R1
                        1$:     INC     R1
                                BNE     1$
                                INCB    R0
                                BNE     1$


                        ;;*************************************************************
                        ;*TEST 1         CHECK INNER LIMIT SWITCH & ELECTROMECHANICAL INTEGRITY
                                ;*THIS TEST PERFORMS 200(8) PURE SEEKS FROM CYLINDER 0
                                ;*TO CYLINDER 312 AND BACK TO 0.  IT IS SUPPOSED TO
                                ;*CHECK THE INNER LIMIT SWITCH AND THE MECHANICAL
                                ;*INTEGRITY OF THE DRIVE.  NOTE THAT A VISUAL CHECK FOR
                                ;ANY MECHANICAL FAILURES WOULD BE VERY HELPFUL.
                        ;;*************************************************************
                        TST1:   SCOPE
```



```
                                CLR     R0              ;INITIALIZE COUNT
                                CLR     R1              ;INITIALIZE COUNT FOR # OF SEEKS
                                CLR     R2              ;CONTAINS SEEK ADRES
                                MOV     #20$,$LPERR     ;SET RETURN ADRES FOR LUPING
                                                        ;ON ERROR
                                MOV     #BUFR,R3        ;INITIALIZE POINTER TO THE TABLE
                                MOV     #-11,R4         ;ALLOW ONLY 9 CNTRL-RDY, SIN, OR R/W/S RDY ERORS
                                MOV     #-10,R5         ;ALLOW ONLY 8 DRU+DRE+ERR+DRY ERRORS
                                BR      2$

                        1$:     TST     R3              ;WAS THERE ANY ERROR?
                                BEQ     3$              ;NO, BRANCH
                        2$:     CLR     R3              ;CLR EROR FLAG
                        20$:    CON.RESET               ;GO DO CNTRL RESET, SUB ROUTINE
                                                        ;AT 'CNT,RST'
                                DRV.RESET               ;GO TO 'DRV,RST' & DO DRV RESET

                        3$:     MOV     DRIVAD,@RKDA    ;ADRES THE DRIVE
                                BIS     R2,@RKDA        ;SET SEEK ADRES
                                TSTB    @RKDS           ;DRIVE RDY?
                                BMI     21$             ;YES
                                JSR     PC,GT4RG        ;NO, GET RKCS, ER, DS, DA
                                ERROR   30              ;DRIVE RDY BIT IS NOT SET
                                                        ;IN RKDS
                                INC     R3              ;SET ERROR FLAG
                                INC     R5              ;ALLOW ONLY 5 ERORS, IF MORE
                                BEQ     10$             ;ABORT

                        21$:    MOV     #11,@RKCS       ;GO, SEEK
                        4$:     INC     R0              ;WAIT FOR CNTRL RDY
                                BNE     5$              ;WAITED LONG?
                                                        ;IF YES, ERROR
                                JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
                                ERROR   1               ;CNTRL RDY DIDN'T SET AFTER
                                                        ;SEEK WAS DONE TO CYLINDER
                                                        ;SHOWN IN RKDA, GO TO 'RK11 LOGIC TESTS'
                                INC     R3              ;SET ERROR FLAG
                                INC     R4              ;EXIT THIS TEST IF THERE R 5 OR MORE ERRORS
                                BEQ     10$
                                BR      6$

                        5$:     TSTB    @RKCS           ;DID CNTRL RDY SET?
                                BPL     4$              ;IF NOT WAIT FOR IT

                        6$:     CLR     R0              ;INITIALIZE COUNT
                                BIT     #100,@RKDS      ;R/W/S RDY SET?
                                BNE     7$              ;YES
                                INC     R0              ;WAIT FOR R/W/S RDY
                                BNE     6$+2
                                JSR     PC,GT4RG        ;GET RKCS, ER, DS, DA
                                ERROR   6               ;R/W/S RDY DID NOT SET WHEN SEEK
                                                        ;WAS DONE TO CYLINDER INDICATED IN RKDA
                                INC     R3              ;SET ERROR FLAG
                                INC     R4              ;IF MAXM EROR COUNT, ABORT
                                BEQ     10$

                        7$:     BIT     #1000,@RKDS     ;SIN ERROR?
                                BEQ     8$              ;NO, BRANCH
```

```
1031  004406  004737  016106          T1      JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
1032  004412  104002                          ERROR   2               ;SIN ERROR, ON DOING SEEK TO
1033                                                                  ;CYL AS SHOWN IN RKDA
1034  004414  005203                          INC     R3              ;SET ERROR FLAG
1035  004416  005204                          INC     R4              ;IF MAXM EROR COUNT REACHED,
1036  004420  001447                          BEQ     18$             ;ABORT THE TEST
1037  004422  005777  175026          8$:     TST     @RKER           ;DRE ERROR?
1038  004426  100006                          BPL     10$             ;NO, BRANCH
1039
1040  004430  004737  016106                  JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
1041  004434  104003                          ERROR   3               ;DRE ON DOING SEEK TO CYLINDER
1042                                                                  ;AS SHOWN IN RKDA
1043  004436  005203                          INC     R3              ;SET ERROR FLAG
1044  004440  005205                          INC     R5              ;IF MAXM EROR COUNT REACHED,
1045  004442  001767                          BEQ     8$              ;ABORT THE TEST
1046
1047  004444  005777  175006          10$:    TST     @RKCS           ;'ERR' BIT IN RKCS SET?
1048  004450  100006                          BPL     12$             ;NO, BRANCH
1049  004452  004737  016106                  JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
1050  004456  104004                          ERROR   4               ;'ERR' IN RKCS SET, ON DOING SEEK
1051                                                                  ;TO CYL AS SHOWN IN RKDA, NOTE
1052                                                                  ;WHICH BIT IN RKER SET?
1053  004460  005203                          INC     R3              ;SET ERROR FLAG
1054  004462  005205                          INC     R5              ;IF MAXM EROR COUNT REACHED,
1055  004464  001425                          BEQ     18$             ;ABORT THE TEST
1056
1057  004466  032777  002000  174756  12$:    BIT     #2000,@RKDS     ;DRU SET?
1058  004474  001406                          BEQ     15$             ;NO, BRANCH
1059  004476  004737  016106                  JSR     PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
1060  004502  104005                          ERROR   5               ;DRU SET, THIS IS AN IRRECOVERABLE
1061                                                                  ;ERROR.  HENCE PUT THE DRIVE ON
1062                                                                  ;LOAD, BACK TO RUN.  DRU ERROR
1063                                                                  ;SHOULD BE CLEARED, IF IT IS NOT
1064                                                                  ;1) THE HEAD POSITION TRANSDVCER LAMP
1065                                                                  ;IS INOPERATIVE
1066                                                                  ;2) OR ERASE OR WRT CURRENT PRESENT
1067                                                                  ;WITHOUT 'WRT GATE'
1068  004504  005203                          INC     R3              ;SET EROR FLAG
1069  004506  005205                          INC     R5              ;ALLOW ONLY 5 ERRORS
1070  004510  001413                          BEQ     18$             ;IF MORE THAN 5
1071                                                                  ;GO TO THE END OF THE PROGRAM
1072
1073  004512  005702                  15$:    TST     R2              ;WAS SEEKING TO 0 OR 312?
1074  004514  001402                          BEQ     16$             ;TO 0, BRANCH
1075                                                                  ;TO 312,
1076  004516  005002                          CLR     R2              ;SEEK NXT TIME TO 0
1077  004520  000647                          BR      1$              ;GO BAK & SK TO 0
1078
1079  004522  012702  014500          16$:    MOV     #14500,R2       ;SEEK NXT TIME TO 312
1080
1081  004526  005201                          INC     R1              ;DONE SEEKS 200 TIMES?
1082  004530  022701  000200                  CMP     #200,R1
1083
1084  004534  001241                          BNE     1$              ;IF NOT, GO BAK
1085  004536  000404                          BR      TST2            ;;EXIT
1086
```



```
1087
1088  004540  104401  001640          18$:    TYPE    ,MSG4
1089  004544  000137  015150                  JMP     TST12
1090
1091                                  ;;*****************************************************************
1092                                  ;*TEST 2        FORMAT THE DISK
1093                                          ;*THIS PROGRAM ASSUMES AN UNFORMATTED DISK AND ITS
1094                                          ;*FORMATTING IS DONE IN THIS TEST. A SECTOR IS FORMATTED
1095                                          ;*AT A TIME.  THE FIRST OWRD OF EVERY SECTOR IS WRITTEN
1096                                          ;*TO BE A PSUEDO-HEADER CONTAINING THE DRIVE #, CYLINDER
1097                                          ;**, SURFACE AND SECTOR #.  THE FOLLOWING IS CHECKED
1098                                          ;*1.  'SIN' IF 'SIN' OCCURS, A DRIVE RESET IS DONE
1099                                          ;*AND THE SAME SECTOR IS FORMATTED AGAIN.  THREE
1100                                          ;*RETRIES ARE DONE BEFORE AN ERROR MESSAGE IS PRINTED.
1101                                          ;*2.  'ERR' ON FINDING THAT THE 'ERR' BIT SET, RKER
1102                                          ;*SCANNED TO FIND OUT WHAT CAUSED IT AND THE
1103                                          ;*ERROR IS REPORTED.
1104                                  ;;*****************************************************************
1105  004550  000004                  TST2:   SCOPE
1106  004552  013737  001230  002120          MOV     DRIVAD,DRHOLD   ;SAVE DRIVE NUMBER
1107  004560  005737  002114                  TST     FDRIVE          ;SEE IF EVEN RK-05F DRIVE
1108  004564  001003                          BNE     11$             ;YES
1109  004566  005737  002116                  TST     FDRVE1          ;ODD RK-05F?
1110  004572  001125                          BNE     TST3            ;DO NOT FORMAT IF ODD RK-05F
1111  004574                          11$:
1112  004574  012702  177152                  MOV     #-626,R2        ;203 CYLINDERS, (406 TRAKS)
1113  004600  012703  177764                  MOV     #-14,R3         ;12 SECTORS
1114  004604  012701  177773                  MOV     #-5,R1          ;ALLOW ONLY 5 'SIN' ERRORS
1115  004610  012705  177773                  MOV     #-5,R5          ;ALLOW ONLY 5 'ERR'S
1116  004614  013704  001230                  MOV     DRIVAD,R4       ;STORE ADRES OF DRIVE.
1117  004620  104415                  4$:     CON.RESET
1118  004622  104416                          DRV.RESET               ;GO TO 'DR-RST' & DO DRIVE RESET
1119  004624  005000                  1$:     CLR     R0              ;KEEP COUNT OF 'SIN' ERORS
1120                                                                  ;ALLOW 3 RETRIES ON SIN
1121  004626  005777  174624                  TST     @RKCS           ;ERR?
1122  004632  100001                          BPL     3$              ;NO
1123
1124  004634  104415                          CON.RESET               ;GO TO 'CN-RST' & DO CONTROL RESET
1125
1126  004636  005046                  3$:     CLR     -(SP)
1127  004640  012746  004646                  MOV     #12$,-(SP)
1128  004644  000002                          RTI                     ;SET PRIORITY TO ZERO
1129  004646  010437  026362          12$:    MOV     R4,OUTBUF       ;WRITE THIS WORD
1130  004652  012777  026362  174602          MOV     #OUTBUF,@RKBA   ;FROM THIS ADRES
1131  004660  010477  174600                  MOV     R4,@RKDA        ;ON THIS DISK SECTOR
1132  004664  012777  177777  174566          MOV     #-1,@RKWC       ;WRITE 1 WORD
1133  004672  012737  004620  001110          MOV     #4$,$LPERR      ;SET RETURN ADDRESS FOR
1134                                                                  ;LUPING ON ERROR
1135
1136  004700  012777  002003  174550          MOV     #2003,@RKCS     ;GO WRT FNT
1137
1138  004706  104421                          CON.RDY                 ;WAIT FOR CONTROL READY
1139  004710  032777  001000  174534  5$:     BIT     #1000,@RKDS     ;WAS THERE A SIN?
1140  004716  001413                          BEQ     6$              ;NO, SKIP DOING DRV RESET
1141  004720  004737  016062                  JSR     PC,GT5RG        ;GO, GET RKCS, ER, DS, DA, CYLINDER
1142  004724  104007                          ERROR   7               ;SIN ERROR ON TRYING TO
```

```
                                                        ;WRT FMT ON CYLINDER AS
                                                        ;INDICATED IN RKDA.  3 RETRIES
                                                        ;ARE DONE
                                                        ;NOTE THAT BEFORE
004726  104415                  CON,RESET             ;RETRYING A DRIVE RESET WAS DONE
004730  104416                  DRV,RESET             ;GO TO 'DR-RST' & DO DRV RESET
004732  005200                  INC     R0            ;INCRMNT SIN COUNT
004734  022700  000003          CMP     #3,R0         ;ALLOW 3 RETRIES WERE THERE 3?
004740  001332                  BNE     1$+2          ;IF NOT, GO & RETRY

004742  005201                  INC     R1            ;ALLOW ONLY 12 SIN ERRORS
                                                        ;IF MORE THAN 5 EXIT THIS TEST
004744  001436                  BEQ     9$            ;IF MORE THAN 5 EXIT THIS TEST
004746  005777  174504      6$: TST     @RKCS         ;DID 'ERR' BIT SET IN RKCS?
004752  100005                  BPL     7$            ;NO, BRANCH
004754  004737  016062          JSR     PC,GT5RG      ;GO, GET RKCS, ER, DS,DA,CYL
004760  104010                  ERROR   10            ;'ERR' OCCURED WHILE DOING
                                                        ;WRT FMT ON SECTOR, CYLINDER
                                                        ;AS INDICATED IN RKDA.
004762  005205                  INC     R5            ;ALLOW ONLY 5 'ERR'S, IF
004764  001426                  BEQ     9$            ;MORE THAN 5 EXIT THIS TEST
004766  005204              7$: INC     R4            ;INCRMNT DISK ADRES TO NXT SCTR
004770  005203                  INC     R3            ;ALL 12 SECTORS DONE?
004772  001314                  BNE     1$            ;IF NOT, GO BAK & FMT NXT SCTR
                                                        ;IF YES
004774  012703  177764          MOV     #-14,R3       ;RESET COUNT FOR 12 SECTORS
005000  042704  000017          BIC     #17,R4        ;CLR OUT SEC BITS

005004  062704  000020      8$: ADD     #20,R4        ;ADRES THE NXT TRAK TO B FMTED
005010  005202                  INC     R2            ;ALL TRAKS FMTED?
005012  001304                  BNE     1$            ;IF NOT GO BAK B FMT NXT CYL, SUR 0
005014  005237  002114          INC     FDRIVE        ;EVEN RK05F?
005020  001004                  BNE     10$           ;NO
005022  062737  020000  001230  ADD     #20000,DRIVAD ;FORMAT ODD DRIVE OF F
005030  000661                  BR      11$
005032  013737  002120  001230 10$: MOV     DRHOLD,DRIVAD ;RESTORE DRIVE ADDR
005040  000402                  BR      TST3          ;;EXIT

005042  004737  016716      9$: JSR     PC,ABRT

                            ;;*************************************************************
                            ;*TEST 3        READ FORMAT OF THE DISK
                                    ;* IN THIS TEST, THE HEADERS FROM ALL THE SECTORS ARE READ
                                    ;* & CHECKED IF THEY ARE CORRECT.  THE FOLLOWING IS THE
                                    ;* TEST SEQUENCE.
                                    ;* 1.  READ 12 SECTORS (HDRS ONLY) AT A TIME
                                    ;* 2.  IF THERE IS A 'SIN' ERROR RETRY ONCE MORE, IF SAW AGAIN
                                    ;* REPORT ERROR & READ HEADER FROM NEXT CYLINDER
                                    ;* 3.  IF THERE IS 'ERR' IN RKCS, DO A CONTROL RESET, REPORT
                                    ;* ERROR & READ HEADER FROM NEXT CYLINDER.  IF THERE ARE
                                    ;* MORE THAN 5 ERRORS OF THIS KIND, THIS TEST WILL BE EXITED
                                    ;* 4.  THE 12 HEADERS ARE CHECKED.  IF THEY ARE CORRECT THE
                                    ;* NEXT CYLINDER IS READ.
                                    ;* IF THEY ARE NOT CORRECT, A RETRY IS DONE; IF AGAIN CORRECT
                                    ;* HEADERS ARE NOT RECIEVED, AN ERROR IS REPORTED.  THE
                                    ;* SECTOR #'S GIVING THE BAD HEADERS, & THE BAD HEADERS ARE
```



```
                                    ;* STORED.
                                    ;* 5.  IF INHIBIT TYPEOUT' SWITCH IS NOT SET, THE FIRST WORDS OF
                                    ;* THE 12 SECTORS (PSUEDO-HEADERS) ARE READ.  IN A PREVIOUS
                                    ;* TEST THE FIRST WORD OF EVERY SECTOR WAS WRITTEN
                                    ;* AS A SOFTWARE HEADER (CONSISTING OF DRIVE #, CYL#, SUR,SEC#)
                                    ;* THEN THE SECTOR # GIVING BAD HEADER, EXPECTED PSUEDO-HEADER,
                                    ;* & THE PS-HEADER RECIEVED ARE TYPED OUT.  THIS WOULD
                                    ;* WRONG, HEADER WAS READ WRONG, ETC.
                                    ;* 6.  THE NEXT CYLINDER IN LINE IS READ.  ORDER OF READING IS
                                    ;* CYL0,SUR0    CYL0,SUR1       CYL312,SUR1
                            ;;*************************************************************
005046  000004            TST3:   SCOPE

005050  012737  177773  001504      MOV     #-5,ERCNT1    ;ALLOW ONLY 5 ERRORS (OF BAD HEADER
                                                        ;KIND FROM 5 CYLINDERS)
005056  012737  177766  001506      MOV     #-12,ERCNT2   ;ALLOW ONLY 12 'ERR'S
005064  012737  177773  001510      MOV     #-5,ERCNT3    ;ALLOW ONLY 5 ERRORS
005072  013705  001230              MOV     DRIVAD,R5     ;SET DRIVE #,CYL ADRES=0
005076  012737  177152  001476      MOV     #-626,INDX2   ;313 CYLS (626 TRAKS) TO B READ
005104  104415              4$:     CON,RESET             ;GO DO CONTROL RESET
005106  104416                      DRV,RESET             ;GO DO DRIVE RESET

005110  005037  001254          1$: CLR     RETRY2        ;ALLOW 2 RETRIES IF HDRS READ WRONG
005114  005037  001252          2$: CLR     RETRY1        ;ALLOW 2 RETRIES FOR 'SINS'

005120  012777  026362  174334  3$: MOV     #OUTBUF,@RKBA ;RD HDRS INTO LOC STARTING AT THIS
005126  010577  174332              MOV     R5,@RKDA      ;FROM THIS DSK ADRES
005132  012777  177764  174320      MOV     #-14,@RKWC    ;12 HDRS TO BE READ
005140  012737  005104  001110      MOV     #4$,$LPERR    ;SET RETURN ADRES FOR LUPING ON ERROR

005146  012777  002005  174302      MOV     #2005,@RKCS   ;GO, RD FMT OF THIS CYLINDER

005154  104421                      CON,RDY               ;WAIT FOR CNTRL RDY TO SET

005156  032777  001000  174266  5$: BIT     #1000,@RKDS   ;'SIN' ERROR?
005164  001420                      BEQ     6$            ;NO, BRANCH
005166  004737  016140              JSR     PC,GETINF

005172  104011                      ERROR   11            ;'SIN' OCCURED WHEN DOING RD FMT
                                                        ;FROM CYL SHOWN IN RKDA.  IT
005174  104415                      CON,RESET             ;DO CNTRL RESET
005176  104416                      DRV,RESET             ;GO, DO DRIVE RESET
005200  005237  001252              INC     RETRY1        ;ALLOW ONLY 2 RETRIES FOR THIS  ERROR
005204  022737  000002  001252      CMP     #2,RETRY1     ;IF TRIED 2 TIMES REPORT
005212  001342                      BNE     3$            ;ERROR, OTHERWISE GO BAK & RETRY
                                                        ;WAS TRIED TWICE, BUT 'SIN'.
005214  005237  001510              INC     ERCNT3        ;ALLOW 5 ERRORS AT MOST
005220  001002                      BNE     6$
005222  000137  005532              JMP     16$

005226  005777  174224      6$: TST     @RKCS         ;'ERR' IN RKCS?
005232  100010                      BPL     7$            ;NO, BRANCH
005234  004737  016062              JSR     PC,GT5RG      ;GO, GET RKCS, ER,DS,DA,CYLNDR
005240  104012                      ERROR   12            ;'ERR' SET WHILE DOING RD FMT
                                                        ;FROM CYL SHOWN IN RKDA
005242  104415                      CON,RESET             ;GO DO CNTRL RESET
```

```
                                                        ;ALLOW ONLY 12 ERRORS OF THIS
005244  005237  001506            INC     ERCNT2        ;KIND, IF MORE THAN FIVE ERRORS
                                                        ;SKIP THIS TEST
005250  001532                    BEQ     TST4          ;;EXIT
005252  000520                    BR      14$           ;GO SET UP TO RD FMT FROM NXT
                                                        ;CYL IN LINE
                                                        ;CHECK THAT CORRECT HEADERS WERE RECVD.
                                                        ;SECTR # HAVING BAD HDR IS STORED ALONG
                                                        ;WITH BAD HDR
005254  004737  007204    7$:     JSR     PC,CHKHDRS    ;GO CHECK IF CORRECT HEADERS WERE READ
005260  005737  001500            TST     INDX3         ;WAS THERE A MISCOMPARISON?
005264  001513                    BEQ     14$           ;IF NOT, GO SET UP TO RD FMT
                                                        ;NXT CYL IN LINE
005266  012737  005114  001110    MOV     #2$,$LPERR
005274  104013                    ERROR   13            ;CORRECT HDRS WERE NOT RECVD
                                                        ;FROM SECTRS AS TYPED OUT.
                                                        ;THE SAME CYLINDER WAS READ TWICE
005276  005237  001254            INC     RETRY2        ;RETRY RD FMT ON SAME CYL AGAIN
005302  022737  000002  001254    CMP     #2,RETRY2     ;TRIED RDING SAME CYL TWICE
005310  001301                    BNE     2$            ;IF NOT, GO RD AGAIN
                                                        ;YES, REPORT ERROR
005312  005237  001504            INC     ERCNT1        ;ALLOW ONLY 5 ERRORS OF THE
                                                        ;ABOVE TYPE.  IF MORE THAN 12
                                                        ;EXIT THIS TEST
005316  001505                    BEQ     16$
005320                    20$:                          ;THE PSUEDO-HEADERS (FIRST WORD OF EVERY
                                                        ;SECTOR) FROM THIS CYLINDER (ABOVE,
                                                        ;THE CYLINDER THAT GAVE WRONG HEADERS)
                                                        ;WILL BE READ, NOW.  FOLLOWING WILL B TYPD OUT:
                                                        ;SEC#, EXPCTD PSUEDO-HDR, RECVD PHDR.
                                                        ;IF "INHIBIT TYPEOUT" SW IS SET THIS ENTIRE
                                                        ;READING & TYPING WILL BE SKIPPED
005320  032777  020000  173612    BIT     #20000,@SWR   ;INHIBIT TYPEOUT?
005326  001072                    BNE     14$           ;YES, SKIP THE FOLLOWING & GO
                                                        ;SET UP TO RD FMT NXT CYL IN LINE
005330  012701  177764            MOV     #-14,R1       ;READ FROM 12 SECTRS
005334  010577  174124            MOV     R5,@RKDA      ;FROM THIS DSK-ADRES
005340  012777  026362  174114    MOV     #OUTBUF,@RKBA ;INTO THIS BUS-ADRES
005346  012777  177777  174104  10$:  MOV   #-1,@RKWC    ;RD 1 WRD
005354  012777  000005  174074    MOV     #5,@RKCS      ;GO, RD
005362  104421                    CON.RDY               ;WAIT FOR CNTRL RDY
005364  005777  174066            TST     @RKCS         ;ANY EROR?
005370  100002                    BPL     15$           ;NO, PROCEED
005372  104415                    CON.RESET             ;CLEAR THE EROR
005374  000447                    BR      14$           ;EROR, SO COULDN'T READ PSUEDO-HDRS
005376  005201            15$:    INC     R1            ;READ FROM ALL 12 SECS
005400  001362                    BNE     10$           ;IF NOT GO RD THE NXT ONE
                                                        ;TYPE OUT PSUEDO-HDRS CORRESPONDING TO
                                                        ;THE SECTORS WHICH GAVE BAD HEADERS
```



```
                                                        ;TYPE: SEC #, EXPC P-HDR, RECVD P-HDR
005402  104401                    TYPE                  ;TYPE OUT
005404  001771                    MSG11
005406  012701  001266            MOV     #BUFR,R1      ;SEC #'S ARE STORED HERE
005412  104401            11$:    TYPE                  ;TYPE CR, LF
005414  001213                    $CRLF
005416  011102                    MOV     (R1),R2
005420  012703  026362            MOV     #OUTBUF,R3    ;PSUEDO-HEADERS WHICH WERE
                                                        ;READ ARE STORED HERE
005424  005702            12$:    TST     R2            ;IS THIS SEC # CORRESPONDING TO THE
005426  001403                    BEQ     13$           ;ONE IN ERROR
005430  005302                    DEC     R2            ;R2 CONTAINS THE SEC #
005432  005723                    TST     (R3)+
005434  000773                    BR      12$
005436  011146            13$:    MOV     (R1),-(SP)    ;GO TYPEOUT SEC # GIVING
005440  104403                    TYPOS                 ;MISCOMPARISON OF HEADERS
005442     002                    .BYTE   2
005443     000                    .BYTE   0             ;SUPRES LDG 0'S
005444  104401                    TYPE                  ;TYPE 2 BLANKS
005446  002105                    BLNKS5
005450  010546                    MOV     R5,-(SP)      ;GO TYPE EXPCTD PSUEDO HEADER
005452  051116                    BIS     (R1),(SP)
005454  104402                    TYPOC
005456  104401                    TYPE                  ;TYPE 2 BLNKS
005460  002103                    BLNKS7
005462  011346                    MOV     (R3),-(SP)    ;GO TYPE PSUEDO-HEADER RECVD
005464  104402                    TYPOC
005466  005721                    TST     (R1)+         ;TYPED OUT ALL SEC #'S IN ERROR.
005470  021127  177777            CMP     (R1),#177777
005474  001346                    BNE     11$           ;IF NOT GO BAK & TYPE NXT
005476  104401  001733            TYPE    ,MSG7         ;TYPE OUT PC
005502  012746  005320            MOV     #20$,-(SP)
005506  104402                    TYPOC
005510  104401  001213            TYPE    ,$CRLF
                                                        ;TYPE ROUTINE ENDS HERE
                                                        ;FIND OUT NXT TRAK TO B READ
                                                        ;FORMATTED
005514  062705  000020    14$:    ADD     #20,R5        ;SET ADRES FOR SUR 0, NXT CYL IN LINE
005520  005237  001476            INC     INDX2         ;READ ALL 313 CYLINDERS (626 TRAKS)?
005524  001404                    BEQ     TST4          ;;EXIT
005526  000137  005110            JMP     1$            ;IF NOT, GO BAK & READ NXT
005532  004737  016716    16$:    JSR     PC,ABRT
```

```
1367
1368                                    ;;*****************************************************************
1369                                    ;*TEST 4         SEEK PATTERNS: 0-312-0-311-...,USING IMPLIED SEEK
1370
1371                                            ;****TEST 2 (WRITING PSUEDO-HEADERS) SHOULD HAVE BEEN DONE BEFORE****
1372                                            ;**** DOING THIS TEST****
1373                                            ;*THIS TEST PERFORMS SEEKS (IMPLIED SEEKS USING 'READS') IN THE
1374                                            ;*FOLLOWING PATTERN.
1375                                            ;*0-312-0-311-0-310-.....0-1-0-0
1376
1377                                            ;*THE FIRST WORD OF EVERY SECTOR IS A PSEUDO-HEADER (WRITTEN IN
1378                                            ;*A PREVIOUS TEST) CONSISTING OF DRIVE NO, CYLINDER NO., SURFACE
1379                                            ;*AND SECTOR NO.  AN IMPLIED SEEK IS DONE BY ISSUING A 'READ' FOR
1380                                            ;*THE PSEUDO-HEADER OF SECTOR 0, SURFACE 0.
1381
1382                                            ;*IF A 'SIN' OCCURS TWO TRIES ARE DONE BEFORE ABORTING.  IF A 'SKE'
1383                                            ;*OCCURS IT COULD MEAN THAT 1) EITHER THE HEADERS WAS REAL 'RONG
1384                                            ;*OR 2) THE HEADS GOT POSITIONED ON THE WRONG CYLINDER.  IN
1385                                            ;*ORDER TO PROVIDE A FURTHER INSIGHT INTO THE PROBLEM, THE FOLLOWING
1386                                            ;*IS DONE:
1387                                            ;*THE HEADERS ARE READ FROM THE CYLINDER THAT GAVE 'SKE'.  IF THE HEADERS
1388                                            ;*ARE CORRECT IT IS SO REPORTED.  IF THE HEADERS ARE INCORECT, THEN THE
1389                                            ;*EXPECTED HEADERS AND THE RECEIVED ONES ARE REPORTED.  ONE MORE TRY IS
1390                                            ;*DONE (THE IMPLIED SEEK IS TRIED AGAIN BETWEEN THE CYLINDERS THAT GAVE RISE
1391                                            ;*TO 'SKE')
1392
1393                                            ;*THE FOLLOWING ACTION IS TAKEN WHEN THERE IS NO 'SKE' OR 'SIN' BUT STILL THE
1394                                            ;*PSEUDO-HEADER IS READ WRONG:
1395                                            ;*FIRST THE HEADERS ARE READ FROM THAT CYLINDER AND CHECKED.  IF THEY ARE
1396                                            ;**CORRECT, IT IS SO REPORTED.  IF THEY ARE WRONG THEN THE EXPECTED AND RECEIVED
1397                                            ;*HEADERS ARE REPORTED.  THEN THE PSEUDO-HEADER FROM SECTOR 1 IS READ AND REPORT
1398                                            ;*ONE MORE TRY IS DONE BY REPEATING THE WHOLE PROCESS. (IMPLIED SEEK
1399                                            ;*BETWEEN THE TWO CYLINDERS AND READING PSEUDO-HEADER FROM SECTOR 0 OF THE DESTI
1400                                            ;*CYLINDER).
1401
1402                                            ;*UP TO 12 ERRORS OF EACH KIND (SIN, SKE, BAD PSEUDO-HEADER) ARE ALLOWED.
1403                                            ;*IF ANY ERROR OCCURS MORE THAN 12 TIMES THE TEST IS ABORTED.
1404                                    ;;*****************************************************************
1405  005536  000004                   TST4:   SCOPE
1406  005540  104415                           CON.RESET                       ;GO DO CONTROL RESET
1407  005542  104416                           DRV.RESET                       ;GO DO DRIVE RESET
1408
1409  005544  005004                           CLR     R4                      ;FLAG, CLR IF DOING IMPLIED
1410                                                                            ;SEEK IN FROM 0 TO 'INADR'
1411                                                                            ;=1, IF GOING FROM 'INADR'
1412                                                                            ;OUT TO CYL 0
1413  005546  012737  177465  001476            MOV     #-313,INDX2             ;313 SEEK PATTERNS
1414  005554  012700  177764                   MOV     #-14,R0
1415  005560  010037  001504                   MOV     R0,ERCNT1               ;ALLOW ONLY 12 ERRORS
1416  005564  010037  001506                   MOV     R0,ERCNT2               ;OF THESE KINDS
1417  005570  010037  001510                   MOV     R0,ERCNT3
1418
1419  005574  012737  014500  001260            MOV     #14500,INADR            ;'INADR' CONTAINS THE INER
1420                                                                            ;CYL TO WHICH IMPLIED SEEK WILL
1421                                                                            ;BE DONE
1422
```



```
1423  005602  005704                   1$:     TST     R4                      ;GOING IN OR OUT?
1424  005604  001005                           BNE     2$                      ;GOING OUT, BRANCH
1425  005606  013705  001260                   MOV     INADR,R5                ;SET CYL ADRES BITS FOR GOING IN
1426  005612  053705  001230                   BIS     DRIVAD,R5               ;FORM DISK ADRES FOR INNER
1427  005616  000402                           BR      3$                      ;CYLINDER
1428  005620  013705  001230           2$:     MOV     DRIVAD,R5               ;FORM DISK ADRES FOR OUTER
1429                                                                            ;CYLINDER = 0
1430                                                                            ;ALLOW 2 TRIES WHEN
1431  005624  012737  177776  001254  3$:     MOV     #-2,RETRY2              ;ERRORS OCCUR
1432  005632  012737  177776  001252  13$:    MOV     #-2,RETRY1
1433  005640  012737  177777  001256  4$:     MOV     #-1,RETRY3
1434  005646  000404                           BR      5$
1435  005650  104415                   6$:     CON.RESET
1436  005652  104416                           DRV.RESET
1437  005654  004737  006452                   JSR     PC,$BR1                 ;REPOSITION HEADS TO PRE-ERROR CYL
1438
1439  005660  012777  177777  173572  5$:     MOV     #-1,@RKWC               ;READ 1 WORD
1440  005666  010577  173572                   MOV     R5,@RKDA                ;FROM THIS CYLINDER, SEC 0
1441  005672  012777  026362  173562           MOV     #OUTBUF,@RKBA           ;INTO THIS BUS ADRES
1442  005700  012737  005650  001110           MOV     #6$,$LPERR              ;SET RETURN ADRES FOR LUPING
1443                                                                            ;ON 'ERROR'
1444
1445  005706  012777  000005  173542           MOV     #5,@RKCS                ;GO, READ
1446
1447  005714  104421                           CON.RDY                         ;WAIT FOR CNTRL RDY
1448
1449  005716  032777  001000  173526           BIT     #1000,@RKDS             ;SIN?
1450  005724  001434                           BEQ     8$                      ;NO, BRANCH
1451                                                                            ;YES, THERE WAS A SIN
1452  005726  004737  016322                   JSR     PC,ERR1                 ;GO GET, CYLS BETW'N WHICH SK WAS TRIED
1453  005732  017737  173514  001170           MOV     @RKDS,$REG3
1454  005740  017737  173510  001166           MOV     @RKER,$REG2
1455  005746  104420  001602                   TYPMSG  ,MSG1
1456  005752  013737  001256  001172           MOV     RETRY3,$REG4            ;SAVE TRY # ON 'SIN'
1457  005760  062737  000002  001172           ADD     #2,$REG4
1458  005766  104014                           ERROR   14                      ;AN IMPLIED SEEK WAS TRIED
1459                                                                            ;FROM 'CYLA' TO 'CYL8' (INDICATED
1460                                                                            ;IN EROR MESAGE), 'SIN' OCCURRED.
1461                                                                            ;2 TRIES ARE DONE BEFORE
1462                                                                            ;ABORTING
1463  005770  005737  001256                   TST     RETRY3                  ;DONE RETRIES
1464  005774  001403                           BEQ     7$                      ;YES, BRANCH
1465  005776  005237  001256                   INC     RETRY3                  ;GO DO 2ND TRY
1466  006002  000722                           BR      6$
1467
1468  006004  005237  001504           7$:     INC     ERCNT1                  ;ALLOW LESS THAN 12 ERORS OF THIS TYPE
1469  006010  001103                           BNE     19$                     ;IF MORE SKIP THIS TEST
1470  006012  000137  006540                   JMP     EXT4                    ;EXIT THIS TEST
1471
1472  006016  032777  010000  173430  8$:     BIT     #10000,@RKER            ;SKE?
1473  006024  001506                           BEQ     20$
1474  006026  004737  016322           15$:    JSR     PC,ERR1                 ;GO GET 2 CYL NOS. BETWEEN WHICH
1475  006032  017737  173416  001166           MOV     @RKER,$REG2             ;IMPLIED SEEK WAS DONE
1476  006040  017737  173406  001170           MOV     @RKDS,$REG3
1477  006046  013737  001252  001172           MOV     RETRY1,$REG4            ;GET TRY # ON 'SKE'
1478  006054  062737  000003  001172           ADD     #3,$REG4
```

```
1479  006062  104420  001610                  TYPMSG  ,MSG2           ;GO PRINT 'SKE'
1480  006066  104014                          ERROR   14              ;IMPLIED SEEK WAS TRIED FROM
1481                                                                  ;'CYLA' TO 'CYLB' (INDICATED
1482                                                                  ;IN EROR MESAGE); 'SKE' OCCURRED,
1483                                                                  ;2 TRIES ARE DONE.
1484  006070  104415                          CON,RESET               ;DO CONTROL RESET
1485
1486  006072  004737  006476          9$:     JSR     PC,SBR2         ;GO READ 12 HEADERS FROM
1487                                                                  ;THIS CYLINDER & COMPARE
1488                                                                  ;THEM.  NOTE R5 CONTAINS THE
1489                                                                  ;DISK ADRES THAT WILL BE USED.
1490
1491  006076  012777  000015  173352          MOV     #15,@RKCS       ;GO DO DRIVE RESET
1492                                                                  ;WHILE THE DRIVE IS DOING RESET
1493                                                                  ;THE HDRS THAT WERE READ
1494                                                                  ;ABOVE ARE CHECKED, PRINTED
1495
1496  006104  005737  001500                  TST     INDX3           ;WAS THERE A MISCOMPARISON
1497                                                                  ;IN ANY HEADER?
1498  006110  001006                          BNE     10$             ;IF INDX3>0, THERE WAS.
1499                                                                  ;NO, THERE WASN'T. HDRS OK
1500  006112  005237  001252                  INC     RETRY1          ;ONLY 2 TRIES FOR SKE
1501  006116  001414                          BEQ     12$             ;BRANCH IF THIS WAS A 2ND TRY
1502  006120  104420  001657                  TYPMSG  ,MSG5           ;TYPE OUT THAT HDRS WERE READ
1503                                                                  ;CORRECTLY.  THIS WAS TRY # 1
1504
1505  006124  000405                          BR      11$
1506
1507                                                                  ;HDRS WERE READ INCORRECT.
1508  006126  005237  001252          10$:    INC     RETRY1          ;ALLOW 2 TRIES FOR SKE
1509  006132  001411                          BEQ     14$             ;BRANCH, IF THIS WAS 2ND TRY
1510
1511                                                                  ;THERE WAS SKE ON DOING IMPLIED
1512  006134  104417  000015                  MESAGE  ,15             ;SEEK TO 'CYL B', THEN HDRS WERE
1513                                                                  ;READ FROM CYL B, WRONG HDRS
1514                                                                  ;RECVD
1515
1516  006140  104423                  11$:    RESDON                  ;WAIT FOR PREVIOUS DRIVE RESET
1517                                                                  ;TO BE DONE
1518  006142  004737  006452                  JSR     PC,SBR1         ;GO, REPOSITION HEADS
1519  006146  000634                          BR      4$
1520
1521                                                                  ;2ND TRY, SKE THIS TIME ALSO, BUT
1522                                                                  ;READ HDRS CORRECTLY FROM
1523                                                                  ;CYLINDER THAT GAVE SKE
1524                                                                  ;NOTE THIS WAS THE 2ND TRY
1525  006150  104420  001657          12$:    TYPMSG  ,MSG5           ;TYPE OUT THAT HDRS WERE
1526                                                                  ;READ CORRECTLY.
1527
1528  006154  000402                          BR      16$
1529
1530                                                                  ;2ND TRY, SKE THIS TIME ALSO,
1531  006156  104417  000015          14$:    MESAGE  ,15             ;READ HDRS FROM CYL THAT
1532                                                                  ;GAVE SKE, THEY WERE INCORRECT.
1533
1534  006162  104423                  16$:    RESDON                  ;WAIT FOR PREVIOUS DRIVE RESET
```



```
1535                                                                  ;TO BE DONE
1536
1537  006164  005237  001506                  INC     ERCNT2          ;ALLOW ONLY LESS THAN 10 ERRORS OF
1538                                                                  ;THIS TYPE (SKE)
1539  006170  001002                          BNE     17$
1540  006172  000137  006540                  JMP     EXT4            ;EXIT THIS TEST IF MORE
1541
1542  006176  005704                  17$:    TST     R4              ;WENT LAST TIME IN OR OUT?
1543  006200  001007                          BNE     19$             ;OUT
1544                                                                  ;IN
1545  006202  005703                          TST     R3              ;WERE HDRS CORRECT?
1546  006204  001005                          BNE     19$             ;NO
1547                                                                  ;YES
1548
1549  006206  005204                  18$:    INC     R4
1550  006210  004737  006452                  JSR     PC,SBR1         ;GO POSITION HEADS BAK ON INNER
1551                                                                  ;CYL
1552  006214  000137  005602                  JMP     1$              ;GO BAK & SEEK OUT NOW
1553
1554  006220  005237  001476          19$:    INC     INDX2           ;ALL SEEK PATTERNS DONE?
1555  006224  001547                          BEQ     TST5            ;;EXIT
1556
1557  006226  162737  000040  001260          SUB     #40,INADR       ;SET ADDRESS FOR THE NXT
1558                                                                  ;INNER CYLINDER
1559  006234  005004                          CLR     R4              ;INDICATE THAT NOW SEEK IS GOING
1560                                                                  ;TO BE IN
1561  006236  000137  005602                  JMP     1$              ;GO BAK & SEEK IN TO 'INADR'
1562
1563  006242                          20$:                            ;IF THERE WAS NO SIN OR SKE
1564                                                                  ;ENTER HERE
1565
1566  006242  012737  005624  001110          MOV     #3$,$LPERR      ;SET RETURN ADRES FOR LUPING
1567                                                                  ;ON ERROR
1568  006250  020537  026362                  CMP     R5,OUTBUF       ;CORRECT PSUEDO-HEADER READ?
1569  006254  001471                          BEQ     24$             ;YES, BRANCH
1570  006256  013737  001254  001172          MOV     RETRY2,$REG4    ;GET TRY #
1571  006264  062737  000003  001172          ADD     #3,$REG4
1572  006272  004737  016322                  JSR     PC,ERR1         ;GO GET CYL #'S BETW'N
1573                                                                  ;WHICH IMPLIED SEEK (READ)
1574                                                                  ;WAS DONE
1575  006276  010537  001166                  MOV     R5,$REG2        ;GET EXPCTD PSUEDO-HDR
1576  006302  013737  026362  001170          MOV     OUTBUF,$REG3    ;GET PSUEDO-HDR RECVD
1577  006310  104016                          ERROR   16              ;IMPLIED SEEK FROM CYLA TO CYLB WAS DONE.
1578                                                                  ;READ PSEUDO-HEADER OF SEC 0,
1579                                                                  ;CYLB (IN EROR MESAGE), BUT
1580                                                                  ;THE WRONG PSEUDO-HEADER WAS
1581                                                                  ;RECEIVED
1582  006312  005237  001254                  INC     RETRY2
1583  006316  001402                          BEQ     21$
1584  006320  000137  005632                  JMP     13$
1585
1586
1587  006324  004737  006476          21$:    JSR     PC,SBR2         ;GO READ HEADERS (12) FROM
1588                                                                  ;THIS CYLINDER, & CHECK THEM.
1589                                                                  ;IF MISCOMPARISON INDX3 WILL
1590                                                                  ;BE > 0.
```

```
1591  006330  005737  001500          T4      TST     INDX3
1592  006334  001003                  BNE     22$
1593  006336  104420  001657          TYPMSG  ,MSG5           ;WRONG PSUEDO-HDR WAS READ
1594                                                           ;BUT WHEN HDRS WERE READ
1595                                                           ;FROM THE SAME CYLINDER, THEY
1596  006342  000402                  BR      23$             ;WERE CORRECT
1597
1598  006344  104417  000015          22$:    MESAGE  ,15     ;WRONG PSUEDO-HDR WAS READ
1599                                                           ;FROM 'CYLB' (IN ERROR MESAGE),
1600                                                           ;THEN HEADERS WERE READ FROM THE
1601                                                           ;SAME CYLINDER.  THEY WERE ALSO
1602                                                           ;WRONG.
1603  006350  010500                  23$:    MOV     R5,R0   ;NOW READ THE PSUEDO-HEADER
1604  006352  005200                  INC     R0              ;FROM THE NEXT SECTOR (1)
1605  006354  010077  173104          MOV     R0,@RKDA        ;SAME CYLINDER
1606  006360  012777  026362  173074  MOV     #OUTBUF,@RKBA
1607  006366  012777  177777  173064  MOV     #-1,@RKWC
1608  006374  012777  000005  173054  MOV     #5,@RKCS
1609  006402  104421                  CON.RDY
1610  006404  010537  001162          MOV     R5,$REG0
1611  006410  004737  016360          JSR     PC,GCYL         ;GO GET CYL # & STORE IT IN $REG0
1612  006414  010037  001164          MOV     R0,$REG1        ;GET EXPCT PSUEDO-HDR FROM SEC 1
1613  006420  013737  026362  001166  MOV     OUTBUF,$REG2
1614  006426  104417  000017          MESAGE  ,17             ;PSUEDO-HEADER FROM SEC 1, CYLB
1615                                                           ;(IN MESAGE) WAS READ.  THE EXPCTD
1616                                                           ;& RECVD DATA WORDS ARE REPORTED.
1617  006432  005237  001510          INC     ERCNT3          ;ALLOW ONLY LESS THAN 10 ERRORS
1618                                                           ;OF THIS TYPE (WRONG PS-HDRS)
1619  006436  001440                  BEQ     EXT4
1620
1621  006440  005704                  24$:    TST     R4      ;SEEKED IN OR OUT LAST TIME?
1622  006442  001266                  BNE     19$             ;IF OUT, GO SEEK NXT INNER CYL
1623                                                           ;IF IN, GO SEEK BAK TO 0
1624  006444  005204                  INC     R4              ;INDICATE THAT SEEK OUT (0)
1625  006446  000137  005602          JMP     1$              ;WILL BE DONE NOW
1626
1627
1628                                          ;THIS ROUTINE IS USED IN THIS TEST ONLY.
1629                                          ;R4=0 INDICATES SEEK BEING DONE FROM
1630                                          ;CYL 0 TO INNER CYL,
1631                                          ;R4=1 INDICATES SEEK BEING DONE FROM
1632                                          ;INNER CYL TO 0, THIS ROUTINE POSITIONS
1633                                          ;THE HEADS ON 'INADR' CYL IF R4=1
1634
1635  006452  005704                  SBR1:   TST     R4
1636  006454  001407                  BEQ     1$
1637  006456  013777  001260  173000  MOV     INADR,@RKDA
1638  006464  012777  000011  172764  MOV     #11,@RKCS
1639  006472  104422                  TST.RWS
1640  006474  000207                  1$:     RTS     PC
1641
1642                                          ;THIS ROUTINE IS USED IN THIS TEST
1643                                          ;ONLY. IT READS 12 HEADERS FROM CYLINDER
1644                                          ;WHOSE ADRES IS IN R5. THEN IT CHECKS
1645                                          ;IF THE EXPECTED HEADER IS RECEIVED.
1646                                          ;IF IT IS NOT, INDX3 IS INCREMENTED INDICATING
```



```
1647                                          ;THE ERROR
1648
1649  006476  012700  177764          SBR2:   MOV     #-14,R0
1650  006502  012701  026362          MOV     #OUTBUF,R1
1651  006506  010077  172746          MOV     R0,@RKWC        ;READ 12 HDRS
1652  006512  010177  172744          MOV     R1,@RKBA        ;INTO THIS ADRES
1653  006516  010577  172742          MOV     R5,@RKDA        ;FROM THIS CYLINDER
1654  006522  012777  002005  172726  MOV     #2005,@RKCS     ;RD FMT, GO
1655  006530  104421                  CON.RDY
1656
1657  006532  004737  007204          JSR     PC,CHKHDRS      ;GO CHECK IF CORRECT HEADERS WERE READ
1658
1659  006536  000207                  RTS     PC              ;EXIT
1660
1661  006540  004737  016716          EXT4:   JSR     PC,ABRT
1662
1663                                  ;;*****************************************************************
1664                                  ;*TEST 5         PERFORM CONVERGING-DIVERGING (IMPLIED) SEEKS
1665                                  ;*THIS TEST PERFORMS A CONVERGING-DIVERGING SEEK PATTERN
1666                                  ;*USING IMPLIED SEEK (READ FORMAT).  THE SEEK SEQUENCE IS:
1667                                  ;*0-312-1-311-2-310-3-307-------310-2-311-1-312
1668                                  ;ALL READ FORMATS ARE DONE FROM SURFACE 0.
1669                                  ;*THE CYLINDER ADDRESSES, BETWEEN WHICH THE IMPLIED SEEK IS
1670                                  ;*PERFRORMED, ARE CONTAINED IN 'OUTADR' & 'INADR'.  IF 'SIN' OCCURS
1671                                  ;*AN ERROR IS REPORTED AND A RETRY IS DONE.  ON READING INCORRECT
1672                                  ;*HEADERS AN ERROR IS REPORTED AND A RETRY IS DONE.  NOTE THAT IF
1673                                  ;*ALL THE 12 HEADERS ARE INCORRECT, IT COULD MEAN THAT THE HEADS
1674                                  ;*COULD NOT POSITION CORRECTLY.  THIS WOULD BE CONFIRMED IF IN
1675                                  ;*PREVIOUS TESTS BAD HEADERS WERE NOT RECIEVED FROM THE SAME
1676                                  ;*CYLINDER.  IF THAT CYLINDER GAVE BAD HEADERS IN ALL THE PREVIOUS
1677                                  ;*TESTS THE PROBLEM COULD BE DIFFERENT.
1678                                  ;*MAXIMUM 12 ERRORS OF ANY KIND ARE ALLOWED.
1679                                  ;*IF MORE THAN 12 ERRORS OCCUR THE TEST IS ABORTED.
1680                                  ;;*****************************************************************
1681  006544  000004                  TST5:   SCOPE
1682  006546  104415                  CON,RESET               ;GO,DO CONTROL RESET
1683  006550  104416                  DRV,RESET               ;GO,DO DRIVE RESET
1684
1685  006552  005004                  CLR     R4              ;(R4)=0 SEEKING FROM 'OUTADR' TO 'INADR'
1686                                                           ;(R4)=1 SEEKING FROM 'INADR' TO 'OUTADR'
1687
1688  006554  012737  177466  001476  MOV     #-312,INDX2     ;SET COUNT FOR DOING 312 TIMES
1689  006562  012700  177764          MOV     #-14,R0
1690  006566  010037  001504          MOV     R0,ERCNT1       ;ALLOW ONLY 12 ERRORS
1691  006572  010037  001506          MOV     R0,ERCNT2
1692
1693  006576  005037  001262          CLR     OUTADR          ;INITIALIZE 'OUTADR' TO 0
1694  006602  012737  014500  001260  MOV     #14500,INADR    ;INITIALIZE 'INADR' TO 312
1695
1696  006610  005704                  18$:    TST     R4      ;GOING IN OR OUT?
1697  006612  001005                  BNE     2$              ;GOING OUT,BRANCH
1698  006614  013705  001260          MOV     INADR,R5        ;SET CYL ADRES BITS FOR GOING IN
1699  006620  053705  001230          BIS     DRIVAD,R5       ;FORM DISK ADRES FOR INNER CYLINDER
1700  006624  000404                  BR      3$
1701
1702  006626  013705  001262          2$:     MOV     OUTADR,R5       ;SET CYL ADRES BITS FOR GOING OUT
```

```
1703  006632  053705  001230                  BIS     DRIVAD,R5       ;FORM DISK ADRES FOR GOING OUT
1704
1705  006636  005037  001254          3$:     CLR     RETRY2          ;ALLOW 2 RETRIES
1706  006642  012737  177777  001252  4$:     MOV     #-1,RETRY1      ;WHEN ERRORS OCCUR
1707  006650  000404                  BR      7$
1708  006652  104415                  5$:     CON,RESET
1709  006654  104416                  DRV,RESET
1710  006656  004737  007150                  JSR     PC,$BR3         ;GO REPOSITION HEADS
1711
1712  006662  012777  177764  172570  7$:     MOV     #-14,@RKWC      ;READ ALL HDRS FROM THIS CYLINDER
1713  006670  010577  172570                  MOV     R5,@RKDA        ;FROM THIS CYL, SEC 0
1714  006674  012777  026362  172560          MOV     #OUTBUF,@RKBA   ;INTO THIS BUS ADRES
1715  006702  012737  006652  001110          MOV     #5$,$LPERR      ;SET RETURN ADRES FOR LOOPING ON ERROR
1716
1717  006710  012777  002005  172540          MOV     #2005,@RKCS     ;READ FORMAT,GO
1718
1719  006716  104421                          CON,RDY                 ;WAIT FOR CONTRL RDY
1720
1721  006720  032777  001000  172524          BIT     #1000,@RKDS     ;SIN?
1722  006726  001443                          BEQ     8$              ;NO, BRANCH
1723  006730  017737  172520  001166          MOV     @RKER,$REG2     ;SAVE RKER
1724  006736  017737  172510  001170          MOV     @RKDS,$REG3     ;SAVE RKDS
1725  006744  013737  001252  001172          MOV     RETRY1,$REG4    ;GET RETRY #
1726  006752  062737  000002  001172          ADD     #2,$REG4
1727  006760  004737  016244                  JSR     PC,ERR2         ;GET CYL #'S BELOW 'N WHICH
1728                                                                  ;SEEK WAS TRIED
1729  006764  104420  001602                  TYPMSG  ,MSG1           ;TYPE 'SIN'
1730  006770  104014                          ERROR   14
1731                                                                  ;'SIN' OCCURRED ON DOING IMPLIED
1732                                                                  ;SEEK FROM 'CYLA' TO 'CYLB' (IN
1733                                                                  ;EROR MESAGE).
1734  006772  005737  001252                  TST     RETRY1          ;DONE 2 TRIES?
1735  006776  001403                          BEQ     6$              ;YES, BRANCH
1736  007000  005237  001252                  INC     RETRY1          ;NO, RETRY
1737  007004  000722                          BR      5$
1738  007006  104415                  6$:     CON,RESET
1739  007010  104416                          DRV,RESET
1740
1741
1742  007012  005237  001504                  INC     ERCNT1          ;ALLOW LESS THAN 12 ERORS OF THE
1743  007016  001527                          BEQ     EXT5            ;ABOVE KIND
1744                                                                  ;IF MORE SKIP THIS TEST
1745                                                                  ;SIN OCCURED WHEN GOING TO CYL (IN
1746                                                                  ;R5).  A DRVE RESET HAS BEEN DONE,
1747                                                                  ;NOW TRY POSITIONING HEADS ON
1748                                                                  ;THAT CYL.
1749  007020  010577  172440                  MOV     R5,@RKDA        ;SET CYL ADRES
1750  007024  012777  000011  172424          MOV     #11,@RKCS       ;SEEK,GO
1751  007032  104422                          TST,RWS
1752  007034  000424                          BR      11$
1753                                                                  ;IF NO SIN, ENTER HERE
1754  007036  004737  007204          8$:     JSR     PC,CHKHDRS      ;GO CHECK IF CORRECT HEADERS WERE READ
1755
1756  007042  012737  006642  001110          MOV     #4$,$LPERR      ;SET LUP ADDRESS
1757  007050  005737  001500                  TST     INDX3           ;WAS THERE A BAD HDR?
1758  007054  001414                          BEQ     11$             ;NO, BRANCH
```



```
1759  007056  104020                          ERROR   20              ;WRONG HEADERS WERE READ FROM
1760                                                                  ;'SEC #'S, ON DOING AN IMPLIED
1761                                                                  ;SEEK (RDFMT) FROM 'CYLA' TO 'CYLB'
1762  007060  005237  001254                  INC     RETRY2          ;ALLOW 2 TRIES
1763  007064  022737  000002  001254          CMP     #2,RETRY2
1764  007072  001263                          BNE     4$              ;GO TRY 2ND TIME
1765  007074  005237  001506                  INC     ERCNT2          ;ALLOW ONLY 12 ERRORS
1766  007100  001002                          BNE     11$
1767  007102  000137  007276                  JMP     EXT5            ;IF MORE, EXIT THIS TEST
1768
1769  007106  005704                  11$:    TST     R4              ;GOING WHICH WAY?
1770  007110  001006                          BNE     12$             ;'INADR' TO 'OUTADR', BRANCH
1771                                                                  ;'OUTADR' TO 'INADR'
1772  007112  005204                          INC     R4              ;INDICATE THAT NXT TIME GOING
1773                                                                  ;FROM 'INADR' TO 'OUTADR'
1774  007114  062737  000040  001262          ADD     #40,OUTADR      ;INCREMENT CYCLINDER ADRES
1775  007122  000137  006610                  JMP     1$              ;GO BAK & DO IMPLIED SEEK
1776                                                                  ;FROM 'INADR' TO 'OUTADR'
1777
1778  007126  005004                  12$:    CLR     R4              ;INDICATE THAT NXT TIME GOING
1779                                                                  ;FROM 'OUTADR' TO 'INADR'
1780  007130  162737  000040  001260          SUB     #40,INADR       ;DECREMENT CYLINDER ADRES
1781  007136  005237  001476                  INC     INDX2           ;DONE ALL 312 FORWARD-BACKWARD
1782                                                                  ;SEEK PATTERNS
1783  007142  001457                          BEQ     TST6            ;;IF YES, EXIT
1784
1785  007144  000137  006610                  JMP     1$              ;IF NOT, GO BAK & DO IMPLIED
1786                                                                  ;SEEK FROM 'OUTADR' TO 'INADR'
1787
1788                                  ;THIS SUBROUTINE IS ENTERED AFTER A 'SIN' ERROR OCCURED ON GOING FROM
1789                                  ;'CYLA' TO 'CYLB' AS INDICATED IN THE ERROR MESSAGE, BEFORE RETRYING THE
1790                                  ;HEADS HAVE TO BE POSITIONED BACK TO 'CYLA', NOTE THAT A DRIVE RESET
1791                                  ;WAS DONE TO CLEAR SIN.
1792                                  ;R4=0, INDICATES SEEK IS BEING DONE FROM 'OUTADR' CYLINDER TO 'INADR'CYLINDER.
1793                                  ;R4=1, INDICATES THAT SEEK IS BEING DONE FROM 'INADR' TO 'OUTADR'.
1794  007150  005704                  $BR3:   TST     R4              ;GOING WHICH WAY?
1795  007152  001404                          BEQ     1$              ;IF FROM 'OUTADR' TO 'INADR', BRANCH.
1796  007154  013777  001260  172302          MOV     INADR,@RKDA
1797  007162  000403                          BR      2$
1798  007164  013777  001262  172272  1$:     MOV     OUTADR,@RKDA
1799  007172  012777  000011  172256  2$:     MOV     #11,@RKCS
1800  007200  104422                          TST,RWS
1801  007202  000207                          RTS     PC
1802
1803
1804                                  ;CHKHDRS
1805                                  ;THIS ROUTINE CHECKS THAT THE HEADERS READ PREVIOUSLY WERE CORRECT.
1806                                  ;IF NOT, THE BAD HEADERS AND THE SECTOR #'S FROM WHICH THEY WERE READ
1807                                  ;ARE STORED.
1808                                  ;ON ENTRY:
1809                                  ;R5 CONTAINS DISK ADDRESS FROM WHERE HEADERS WERE READ.
1810                                  ;OUTBUF - 12 HEADRERS READ PREVIOUSLY ARE STORED STARTING 'OUTBUF'.
1811                                  ;ON EXIT:
1812                                  ;INDX3=0, IF THE HEADERS WERE CORRECT
1813                                  ;INDX3=1, IF THE HEADERS WERE INCORRECT
1814                                  ;BUFR - SECTOR #'S GIVING BAD HEADRERS 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.
007204  005000          CHKHDRS: CLR     R0              ;INITIALIZE FOR 14 HDRS
007206  012701  026362          MOV     #OUTBUF,R1      ;INITIZLIZE PTR TO HDRS RECVD
007212  012702  001266          MOV     #BUFR,R2        ;INITIALIZE PTR TO SECTOR TABLE
007216  012703  001320          MOV     #BUFR1,R3       ;INITIALIZE PTR TO BAD HDR TABLE
007222  010537  001120          MOV     R5,@GDADR
007226  042737  160037  001120  BIC     #160037,@GDADR  ;GET EXPCTD HEADER
007234  005037  001500          CLR     INDX3           ;CLR FLG INDICATING BAD HDRS
007240  023711  001120  9$:     CMP     @GDADR,(R1)     ;HEADER OK?
007244  001406                  BEQ     10$             ;YES,BRANCH
007246  011123                  MOV     (R1),(R3)+      ;SAVE BAD HDR
007250  010022                  MOV     R0,(R2)+        ;SAVE BAD SECTR #
007252  012712  177777          MOV     #177777,(R2)    ;PUT TERMINATR ON SECTR TABLE
007256  005237  001500          INC     INDX3           ;SET FLG INDICATING BAD HDR
007262  005721          10$:    TST     (R1)+           ;INCRMNT PTR TO NXT HDR
007264  005200                  INC     R0              ;ALL HDRS CHKD?
007266  022700  000014          CMP     #14,R0
007272  001362                  BNE     9$              ;IF NOT, LUP BAK
007274  000207                  RTS     PC
007276  004737  016716  EXT5:   JSR     PC,ABRT

                                ;;**************************************************************
                                ;*TEST 6          WRITE PATTERNS ON THE DISK
                                ;*IN THIS TEST DIFFERENT PATTERNS ARE WRITTEN ON THE ENTIRE DISK,  768
                                ;*WORDS (3 SECTORS) ARE WRITTEN AT A TIME,  TWO 768 WORDS LONG
                                ;*BUFFERS HAVE BEEN USED IN THIS TEST.  WHILE ONE BUFFER IS
                                ;*USED TO WRITE ON THE DISK, THE OTHER BUFFER GETS FILLED UP WITH
                                ;*PATTERNS TO BE USED NEXT!  THIS OVERLAPPING IS DONE TO SAVE TIME.

                                ;*THREE PATTERN-GENERATOR SUB-ROUTINES ARE USED:
                                ;*1,  PTGEN0    2,  PTGEN1      3,  PTGEN2      4,  PTGEN3
                                ;*THESE THREE ROUTINES ARE CALLED IN A CYCLIC ORDER:
                                ;*      PTGEN0-PTGEN1-PTGEN2-PTGEN3-PTGEN0-PTGEN1.......
                                ;*THE FOLLOWING ORDER IS MAINTAINED IN WRITING BLOCKS (EACH
                                ;*3 SECTORS LONG) USING ONE OF THE FOUR PATTERN GENERATORS:

                                ;*CYL   SECTORS       CYL   SECTORS       CYL   SECTORS       CYL   SECTORS
                                ;*   SUR       ROUTINE   SUR       ROUTINE   SUR       ROUTINE   SUR       ROUTI
                                ;* 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 ADDRES OF PAT-GENERATOR 0
                                ;*LOCATION 'PAT3' TO 'PTGEN0' THE STARTING ADDRESS OF PAT-GENERATOR 0
                                ;;**************************************************************
007302  000004          TST6:   SCOPE
007304  012737  177764  001504  MOV     #-14,ERCNT1
007312  012737  177764  001506  MOV     #-14,ERCNT2
007320  012737  177152  001474  MOV     #-626,INDX1     ;SET COUNT FOR 313X2 TRACKS
007326  005037  001410          CLR     BUFLG0          ;CLR FLAG FOR BUFR 0
007332  005037  001412          CLR     BUFLG1          ;CLR FLAG FOR BUFR 1
                                                        ;BIT 7 OF ABOVE FLAGS WHEN SET
                                                        ;INDICATES, THAT BUFR TO BE USED
                                                        ;FOR WRITING ON DSK
007336  013737  001230  001432  MOV     DRIVAD,DSKADR   ;GET DRIVE #, DISK ADRES
007344  012737  001414  001424  MOV     #PAT0,PRSPAT    ;INITIALIZE PTR TO THE FIRST
                                                        ;PATRN GENERATOR
007352  004737  007770          JSR     PC,GETBUF
007356  017737  172042  001430  MOV     @PRSPAT,PGSUBR
007364  004777  172040          JSR     PC,@PGSUBR      ;GO GENERATE PATRNS FOR
                                                        ;3 SECTOR (400X3)
007370  005005          18:     CLR     R5              ;INITIALIZE COUNT FOR 4 BLOCKS
007372  005737  001410  28:     TST     BUFLG0          ;FIND OUT WHICH BUFR TO USE
007376  100407                  BMI     3$              ;FOR WRITING ON DSK
007400  013737  001406  001434  MOV     PBUF1,BUSADR    ;USE 'IOBUF1' FOR TRANSFER
                                                        ;OR THE ONE INDICATED BY THE USER
007406  052737  000200  001412  BIS     #BIT7,BUFLG1    ;SET FLAG TO INDICATE THAT
                                                        ;WRTING ON DSK WILL B DONE FROM
                                                        ;THIS BUFR (BUF 1)
007414  000406                  BR      13$
007416  013737  001404  001434  3$:     MOV     PBUF0,BUSADR    ;USE 'IOBUF0' FOR TRANSFER
                                                        ;OR THE ONE INDICATED BY THE USER
007424  052737  000200  001410  BIS     #BIT7,BUFLG0    ;INDICATE THAT 'IOBUF0' WILL
                                                        ;B USED FOR WRTING ON DISK
007432  012737  007440  001110  13$:   MOV     #40,@LPERR
007440  013777  001432  172016  40$:   MOV     DSKADR,@RKDA    ;SET RKDA
007446  013777  001434  172006          MOV     BUSADR,@RKBA
007454  012777  176400  171776          MOV     #-1400,@RKWC
007462  012777  000003  171766          MOV     #3,@RKCS        ;WRITE THE 4 SECTORS ON
                                                        ;DISK
007470  013737  001424  001426          MOV     PRSPAT,NXTPAT   ;WHILE THE PATRNS R BEING WRITTEN
007476  023727  001424  001422          CMP     PRSPAT,#PAT3    ;GO GENERATE THE NXT PATRNS
007504  001004                  BNE     5$              ;TO B WRITTEN
007506  012737  001414  001426          MOV     #PAT0,NXTPAT    ;KEEP GENERATING PATRNS IN THIS
007514  000403                  BR      6$              ;WAY "PAT0"-"PAT1"-"PAT2"-"PAT3"-"PAT0"-
007516  062737  000002  001426  5$:     ADD     #2,NXTPAT
007524  004737  007770          6$:     JSR     PC,GETBUF
007530  017737  171672  001430          MOV     @NXTPAT,PGSUBR
007536  004777  171666          JSR     PC,@PGSUBR      ;GO GENERATE THESE PATRNS.
                                                        ;(3 X 400) WORDS
007542  104421                  CON,RDY
```

```
1927  007544  032777  140000  171704          BIT     #140000,@RKCS   ;ANY ERROR?
1928  007552  001411                          BEQ     12$             ;GET RKCS,ER,DS,DA
1929  007554  004737  016106                  JSR     PC,GT4RG
1930  007560  104021                          ERROR   21              ;ERROR ON DOING WRITE
1931  007562  104415                          CON,RESET               ;CLEAR IT
1932  007564  005237  001504                  INC     ERCNT1          ;ALLOW 12 ERRORS AT MOST
1933  007570  001002                          BNE     12$
1934  007572  000137  010356                  JMP     EXT6            ;IF MORE, EXIT
1935  007576  032777  001000  171646  12$:    BIT     #BIT9,@RKDS     ;SIN, ON DOING WRITE?
1936  007604  001412                          BEQ     7$
1937  007606  004737  016106                  JSR     PC,GT4RG
1938  007612  104022                          ERROR   22              ;SIN ERROR ON DOING WRITE
1939  007614  104415                          CON,RESET
1940  007616  104416                          DRV,RESET
1941  007620  005237  001506                  INC     ERCNT2          ;ALLOW 12 ERRORS AT MOST
1942  007624  001002                          BNE     7$
1943  007626  000137  010356                  JMP     EXT6
1944                                                                  ;FIGURE OUT WHICH BUFFER IS
1945                                                                  ;AVAILABLE FOR USE
1946  007632  105737  001410          7$:     TSTB    BUFLG0          ;WAS PREVIOUS DSK-WRITE DONE
1947  007636  100003                          BPL     8$              ;USING BUFR 0?
1948  007640  005037  001410                  CLR     BUFLG0          ;YES, CLR FLAG INDICATING IT'S
1949                                                                  ;AVAILABLE NOW
1950  007644  000402                          BR      9$
1951  007646  005037  001412          8$:     CLR     BUFLG1          ;CLR FLAG INDICATING BUFR1
1952                                                                  ;IS AVAILABLE NOW
1953  007652  013737  001426  001424  9$:     MOV     NXTPAT,PRSPAT   ;'PRSPAT'S PATRNS WILL BE USED
1954                                                                  ;ON NEXT WRITE
1955  007660  010500                          MOV     R5,R0           ;FORM SEC # TO BE USED NXT TIME
1956  007662  116000  010352                  MOVB    SECPTR(R0),R0   ;GET SEC #
1957  007666  042737  000017  001432  10$:    BIC     #17,DSKADR      ;MASK SECTOR BITS FROM DSK-ADRES
1958  007674  050037  001432                  BIS     R0,DSKADR       ;FORM NXT DSK-ADRES
1959  007700  005205                          INC     R5              ;DONE WITH 12 SECTRS (3 BLOCKS)?
1960  007702  022705  000004                  CMP     #4,R5
1961  007706  001231                          BNE     2$              ;ON THIS SURFACE? IF NOT, GO
1962                                                                  ;DO NXT 4 SECTRS
1963  007710  032737  000001  001474          BIT     #BIT0,INDX1     ;WHICH SURFACE WAS DONE, 0 OR 1?
1964  007716  001415                          BEQ     11$             ;IF 0, GO DO SURFACE 1
1965  007720  005237  001474                  INC     INDX1           ;COUNT # OF TRACKS
1966  007724  001002                          BNE     .+6
1967  007726  000137  010362                  JMP     TST7            ;EXIT IF DONE
1968  007732  042737  000020  001432          BIC     #20,DSKADR      ;GO TO SUR 0, NEXT CYLINDER
1969  007740  062737  000040  001432          ADD     #40,DSKADR
1970  007746  000137  007370                  JMP     1$              ;GO BACK AND DO WRITE
1971  007752  005237  001474  11$:    INC     INDX1           ;COUNT # OF TRACKS
1972  007756  052737  000020  001432          BIS     #20,DSKADR      ;SET BIT FOR SUR 1
1973  007764  000137  007370                  JMP     1$              ;GO, WRITE PATRNS ON SURFACE 1
1974                                  ;*GET BUF#
1975                                  ;*THIS ROUTINE FINDS OUT  WHICH BUFFER (IOBUF0, IOBUF1) OR ONE OF
1976                                  ;*THE TWO BUFFERS SELECTED BY THE USER) TO USE
1977                                  ;*FOR GENERATING PATTERNS. THEN IT SETS UP A FLAG INDICATING THAT
1978                                  ;*BUFFER HAS TO BE FILLED UP. THE STARTING ADDRESS OF THE
1979                                  ;*BUFFER IS STORED IN 'BUF #'.
1980
1981  007770  005737  001410          GETBUF: TST     BUFLG0          ;BUFR 0 AVAILABLE FOR USE?
1982  007774  100007                          BPL     1$              ;YES, BRANCH
```



```
1983  007776  052737  100000  001412          BIS     #BIT15,BUFLG1   ;NO, USE BUFR 1, INDICATE SO.
1984  010004  013737  001406  001446          MOV     PBUF1,BUFNO     ;SAVE STARTING ADRES OF BUFR1
1985  010012  000207                          RTS     PC
1986  010014  052737  100000  001410  1$:     BIS     #BIT15,BUFLG0   ;INDICATED, USING BUF 0
1987  010022  013737  001404  001446          MOV     PBUF0,BUFNO     ;SAVE STARTING ADRES OF BUFR 0
1988  010030  000207                          RTS     PC              ;RETURN
1989                                  ;*PTGEN0
1990                                  ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS LONG
1991                                  ;*BUFFER CONTAINING THE FOLLOWING PATTERNS
1992                                  ;*FIRST BLOCK OF 256 WORDS:     125252
1993                                  ;*SECOND BLOCK OF 256 WORDS:    052525 (COMPLEMENT OF ABOVE)
1994                                  ;*THIRD BLOCK OF 256 WORDS:     010421
1995                                  ;*YOU CAN USE ANY OTHER PATTERN/S (& ITS COMPLEMENT)
1996                                  ;*MAKING THE CHANGES IN THE 2 LOCATIONS SHOWN BELOW.
1997
1998  010032  013700  001446          PTGEN0: MOV     BUFNO,R0                ;GET STARTING ADRES OF BUFR
1999  010036  013701  010110                  MOV     PTRN01,R1       ;GET PATRN TO BE GENERATED
2000  010042  012702  177400                  MOV     #-400,R2        ;IN THE FIRST 400 WORD BLOCK
2001
2002  010046  010120                  1$:     MOV     R1,(R0)+        ;GENERATE THE FIRST BLOCK
2003  010050  005202                          INC     R2              ;WITH 'PAT01' PATRN
2004  010052  001375                          BNE     1$              ;ALL DONE?
2005
2006  010054  012702  177400                  MOV     #-400,R2
2007  010060  005101                          COM     R1              ;COMPLEMENT 'PAT01' PATAN
2008  010062  010120                  2$:     MOV     R1,(R0)+        ;GENERATE 2ND BLOCK WITH
2009  010064  005202                          INC     R2              ;'PAT01'S COMPLEMENT PATRN
2010  010066  001375                          BNE     2$              ;ALL DONE?
2011  010070  012702  177400                  MOV     #-400,R2
2012  010074  013701  010112                  MOV     PTRN02,R1       ;GET PATRN TO BE GENERATED
2013                                                                  ;FOR 3RD BLOCK
2014  010100  010120                  3$:     MOV     R1,(R0)+        ;GENERATE 3RD BLOCK USING
2015                                                                  ;'PAT02' PATRN
2016  010102  005202                          INC     R2
2017  010104  001375                          BNE     3$              ;ALL DONE?
2018
2019  010106  000207                          RTS     PC              ;RETURN
2020
2021  010110  125252          PTRN01: 125252                  ;CHANGE THESE LOCATIONS IF
2022  010112  010421          PTRN02: 010421                  ;YOU WANT ANY OTHER PATTERNS
2023
2024                                  ;*PTGEN1
2025                                  ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS
2026                                  ;*LONG BUFFER CONTAINING THE FOLLOWING PATTERNS:
2027
2028                                  ;*FIRST BLOCK-256 WORDS 000001  FILL 1'S
2029                                  ;*                      000003
2030                                  ;*                      .
2031                                  ;*                      177777
2032
2033                                  ;*SECOND BLOCK          177776  FILL 0'S
2034                                  ;*                      177774
2035                                  ;*                      .
2036                                  ;*                      000000
2037
2038                                  ;*THIRD BLOCK           000001  FLOAT A 1
```

```
                                        ;*                      000020
                                        ;*                      :
                                        ;*                      100000
                                        ;*'BUFNO' CONTAINS THE STARTING ADDRESS OF THE
                                        ;*BUFFER.
010114  012703  000001          PTGEN1: MOV     #1,R3           ;INITIALIZE PATRNS
010120  012704  177776                  MOV     #177776,R4
010124  013700  001446                  MOV     BUFNO,R0                ;GET STARTING ADRES OF BUFR
010130  012702  177760                  MOV     #-20,R2         ;SET COUNT
010134  010301          1$:     MOV     R3,R1           ;INITIALIZE PATRN
010136  010120          2$:     MOV     R1,(R0)+        ;GENERATE THE FIRST
010140  000261                  SEC                     ;BLOCK USING "FILL 1'S"
010142  006101                  ROL     R1
010144  103374                  BCC     2$
010146  005202                  INC     R2
010150  001371                  BNE     1$              ;ALL DONE?
010152  012702  177760                  MOV     #-20,R2
010156  010401          3$:     MOV     R4,R1           ;INITIALIZE PATRN
010160  010120          4$:     MOV     R1,(R0)+        ;GENERATE 2ND BLOCK
010162  000241                  CLC                     ;USING "FILL 0'S"
010164  006101                  ROL     R1
010166  103774                  BCS     4$
010170  005202                  INC     R2
010172  001371                  BNE     3$              ;DONE?
010174  012702  177760                  MOV     #-20,R2         ;SET COUNT
010200  010301          5$:     MOV     R3,R1           ;INITIALIZE PATRN
010202  010120          6$:     MOV     R1,(R0)+        ;GENERATE THE 3RD BLOCK
010204  006301                  ASL     R1              ;USING "FLOAT A 1"
010206  103375                  BCC     6$
010210  005202                  INC     R2
010212  001372                  BNE     5$              ;DONE?
010214  000207                  RTS     PC              ;RETURN
                                        ;*PTGEN2
                                        ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS LONG
                                        ;*BUFFER CONTAINING THE FOLLOWING PATTERNS:
                                        ;*FIRST BLOCK=256 WORDS 000000  COUNT PATRN=LOWER BYTE
                                        ;*                      000001  0-377
                                        ;*                      000002
                                        ;*                      :
                                        ;*                      000377
                                        ;*SECOND BLOCK          000000  COUNT PATRN=HIGHER BYTE
                                        ;*                      000400  0-377
                                        ;*                      001000
                                        ;*                      :
                                        ;*                      177400
                                        ;*THIRD BLOCK           000000  COUNT PATRN=HIGHER & LOWER BYTE
                                        ;*                      000401  0-377, 0-377
                                        ;*                      :
```



```
                                        ;*                      177777
                                        ;*'BUFNO' CONTAINS THE STARTING ADDRESS OF THE BUFFER.
010216  005001                  PTGEN2: CLR     R1              ;INITIALIZE PATRN
010220  013700  001446                  MOV     BUFNO,R0
010224  010120          1$:     MOV     R1,(R0)+        ;GENERATE 1ST BLOCK USING
010226  105201                  INCB    R1              ;USING 'COUNT UP LOWER BYTE'
010230  001375                  BNE     1$              ;DONE?
010232  005001                  CLR     R1
010234  010120          2$:     MOV     R1,(R0)+        ;GENERATE 2ND BLOCK
010236  062701  000400                  ADD     #400,R1         ;USING 'COUNT UP HIGHER BYTE'
010242  103374                  BCC     2$              ;DONE?
010244  005001                  CLR     R1
010246  010120          3$:     MOV     R1,(R0)+        ;GENERATE 3RD BLOCK USING
010250  062701  000401                  ADD     #401,R1         ;'COUNT UP HIGHER & LOWER BYTE'
010254  103374                  BCC     3$              ;ALL DONE?
010256  000207                  RTS     PC              ;RETURN
                                        ;PTGEN3
                                        ;*THIS ROUTINE GENERATES A 768 (3X256) WORDS LONG
                                        ;*BUFFER CONTAINING THE FOLLOWING PATTERNS:
                                        ;*FIRST BLOCK OF 256 WORDS:     167356 (COMPLEMENT OF 010421)
                                        ;*SECOND BLOCK          177776  FLOAT A 0
                                        ;*                      177775
                                        ;*                      :
                                        ;*                      077777
                                        ;*THIRD BLOCK           000377  COUNT UP HIGHER BYTE 0-377
                                        ;*                      000776  COUNT DOWN LOWER BYTE 377-0
                                        ;*                      :
                                        ;*                      177400
                                        ;*'BUFNO' CONTAINS THE STARTING ADDRESS OF THE BUFFER.
010260  013700  001446          PTGEN3: MOV     BUFNO,R0
010264  012702  177400                  MOV     #-400,R2
010270  013701  010112                  MOV     PTRN02,R1       ;GET PATTERN
010274  005101                  COM     R1              ;COMPLEMENT 'PAT02' PATRN
010276  010120          4$:     MOV     R1,(R0)+        ;GENERATE 1ST BLOCK
010300  005202                  INC     R2
010302  001375                  BNE     4$              ;ALL DONE?
                                                        ;2ND BLOCK
010304  012702  177760                  MOV     #-20,R2
010310  000261          7$:     SEC
010312  012701  177776                  MOV     #177776,R1
010316  010120          8$:     MOV     R1,(R0)+
010320  006101                  ROL     R1
010322  103775                  BCS     8$
010324  005202                  INC     R2
```

```
2151  010326  001370                    BNE     7$              ;ALL DONE?
2152
2153                                    ;3RD BLOCK
2154  010330  012701  000377            MOV     #377,R1
2155  010334  010102                    MOV     R1,R2           ;GENERATE 3RD BLOCK USING
2156  010336  010120            9$:     MOV     R1,(R0)+        ;'COUNT DOWN LOWER BYTE'
2157  010340  060201                    ADD     R2,R1           ;'COUNT UP HIGHER BYTE'
2158  010342  022701  177777            CMP     #-1,R1
2159  010346  001373                    BNE     9$              ;ALL DONE?
2160  010350  000207                    RTS     PC              ;RETURN
2161
2162                                    ;SECTOR POINTER TABLE. DATA TRANSFERS ARE DONE IN BLOCKS (3 SECTORS
2163                                    ;EACH) IN THE CYCLIC ORDER:  0-2, 6-10, 3-5, 11-13, 0-2, AND SO ON.
2164  010352      006           SECPTR: .BYTE   6
2165  010353      003                   .BYTE   3
2166  010354      011                   .BYTE   11
2167  010355      000                   .BYTE   0
2168
2169  010356  004737  016716    EXT6:   JSR     PC,ABRT
2170
2171
2172                                    ;;*****************************************************************
2173                                    ;*TEST 7         READ, SOFTWARE COMPARE, WRITE CHECK OF THE PATTERNS
2174
2175                                            ;****TEST 6 (WRITING PATTERNS ON DISK) SHOULD BE DONE BEFORE DOING****
2176                                            ;****THIS TEST.****
2177
2178                                            ;*IN THIS TEST THE PATTERNS THAT WERE WRITTEN BEFORE (IN THE
2179                                            ;*PREVIOUS TEST) ARE READ BACK AND SOFTWARE COMPARED.
2180                                            ;*WHILE THE SOFTWARE COMPARISON IS TAKING PLACE, AN OVERLAPPING
2181                                            ;*'WRITE CHECK' IS DONE FOR THE SAME BLOCK.  THE READING
2182                                            ;*BACK OF EACH BLOCK (4 SECTORS) IS DONE IN THE SAME
2183                                            ;*MANNER AS DESCRIBED IN THE BEGINNING OF PREVIOUS TEST.
2184                                            ;*OVERLAPPING OPERATIONS AND STAGGERING OF BLOCKS ARE DONE IN
2185                                            ;*A SIMILIAR MANNER.
2186                                            ;*THE FOLLOWING IS A DESCRIPTION OF HOW ERRORS ARE HANDLED.
2187                                            ;*IF A 'SIN' OR 'HE' OCCURS IT IS REPORTED AND THAT BLOCK
2188                                            ;*IS SKIPPED.  IN THIS STAGE OF TESTING THESE ERRORS SHOULD NOT
2189                                            ;*NORMALLY OCCUR.
2190                                            ;*IF A CHECKSUM ERROR OCCURS, CONTROL IS TRANSFERRED TO
2191                                            ;*'CSERROR'.  FIRST THE CSE IS REPORTED.  THE SECTOR GIVING
2192                                            ;*CSE IS READ AGAIN.  IN ALL 3 TRIES ARE DONE.  IF STILL
2193                                            ;*THE ERROR PRESISTS THAT SECTOR IS ABORTED AND THE REST
2194                                            ;*OF THE SECTORS ARE READ AND CHECKED FOR CSE.  IF
2195                                            ;*AGAIN SOME OTHER SECTOR GIVES CSE, REPORTING AND RETRIES
2196                                            ;*ARE DONE AGAIN.
2197                                            ;*NEXT SOFTWARE COMPARISON IS DONE OF THE DATA THAT WAS
2198                                            ;*READ BACK.  ON GETTING A DATA MISCOMPARISON
2199                                            ;*THE RELEVANT INFO(GOOD DATA, BAD DATA, ADDRESS ETC.) IS
2200                                            ;*STORED.  AFTER THE WHOLE BLOCK HAS BEEN CHECKED, THE DATA
2201                                            ;*ERROR/S IS/ARE REPORTED.  THEN THE BLOCK IS READ AGAIN.  IN ALL
2202                                            ;*THREE TRIES ARE DONE.
2203                                            ;*WHILE THE DATA COMPARISON (SOFTWARE) IS GOING ON THE 'WRITE
2204                                            ;*CHECK' IS ALSO IN PROGRESS.  IF A WRITE CHECK ERROR OCCURS THE
2205                                            ;*CONTROL IS TRANSFERRED TO 'WCEROR'.  WRITE CHECK OF THE SECTOR
2206                                            ;*THAT GAVE WCE IS DONE AGAIN.  IN ALL THREE TRIES ARE DONE.
```



```
2207                                            ;*NOTE THAT EVERY WCE IS REPORTED. IF ALL THE 4 SECTOR OF
2208                                            ;*A BLOCK GAVE WCE'S, ALL 4 WILL BE REPORTED AND RETRIED.
2209
2210                                            ;*DEPENDING ON THE NATURE OF THE PROBLEM, THE ABOVE ERRORS
2211                                            ;*CAN OCCUR IN ANY COMBINATION.  IT IS RECOMMENDED THAT ALL
2212                                            ;*THE ERROR MESSAGE BE CAREFULLY CO-RELATED AND EXAMINED.  IT
2213                                            ;*WILL PROVIDE A DEEP INSIGHT INTO THE PROBLEM.
2214                                    ;;*****************************************************************
2215  010362  000004            TST7:   SCOPE
2216  010364  012737  177764  001510    MOV     #-14,ERCNT3     ;ALLOW 12 ERRORS AT THE MOST
2217  010372  012737  177773  001512    MOV     #-5,ERCNT4      ;ALLOW 5 ERRORS AT THE MOST
2218  010400  012737  177742  001514    MOV     #-36,ERCNT5     ;ALLOW 10 ERRORS AT THE MOST
2219  010406  012737  177742  001516    MOV     #-36,ERCNT6     ;ALLOW 10 ERRORS AT THE MOST
2220  010414  012737  177764  001520    MOV     #-14,ERCNT7     ;ALLOW 12 ERRORS AT THE MOST
2221  010422  012737  177742  001522    MOV     #-36,ERCNT8     ;ALLOW 10 ERRORS AT THE MOST
2222  010430  012737  177152  001474    MOV     #-626,INDX1     ;SET COUNT FOR 626 TRACKS
2223  010436  005037  001476            CLR     INDX2           ;CLR COUNT FOR 4 BLOCKS ON A TRACK
2224
2225  010442  012737  001414  001424    MOV     #PAT0,PRSPAT    ;INITLZE PTR TO PATRN GENRTR
2226  010450  013737  001230  001450    MOV     DRIVAD,ADRES    ;INITLZE DRV#,ADRES
2227
2228
2229  010456  005037  001504    BEGIN:  CLR     ERCNT1          ;IF > 0, MEANS THAT RETRIES
2230  010462  005037  001506            CLR     ERCNT2          ;DONE AFTER CSE OR CSE CHKD
2231  010466  012737  177775  001252    MOV     #-3,RETRY1      ;RETRY COUNT FOR CSE
2232  010474  012737  177776  001254    MOV     #-2,RETRY2      ;RETRY COUNT FOR SFTWRE MISCMP'N
2233  010502  012737  000003  001256    MOV     #3,RETRY3       ;RETRY COUNT FOR WCE
2234
2235  010510  013737  001450  001440    MOV     ADRES,WDSKAD    ;DISK ADRES TO WRT CHK WITH
2236  010516  013737  001406  001442    MOV     PBUF1,WBUSAD    ;USE THIS BUFR 1 TO WRT CHK
2237                                            ;OR THE BUFR INDICATED BY THE USER
2238  010524  012737  176400  001444    MOV     #-1400,WWRDCN   ;WRT CHK 1 BLOCK=3SECS=1400 WRDS
2239
2240  010532  013737  001450  001432  READ:   MOV     ADRES,DSKADR    ;DISK ADRES TO READ FROM
2241  010540  012737  176400  001436    MOV     #-1400,WRDCNT   ;1 BLOCK = 3 SECTORS = 1400 WRDS
2242  010546  013737  001404  001434    MOV     PBUF0,BUSADR    ;USE 'IOBUF0' TO READ INTO
2243                                            ;OR THE BUFR INDICATED BY THE USER
2244
2245  010554  000404                    BR      RDAGAIN
2246
2247  010556  104415            LUPSIN: CON,RESET
2248  010560  104416                    DRV,RESET
2249  010562  000401                    BR      RDAGAIN
2250
2251  010564  104415            LUPHE:  CON,RESET
2252
2253  010566  013777  001432  170670  RDAGAIN:        MOV     DSKADR,@RKDA    ;READ FROM THIS DSK-ADRES
2254  010574  013777  001436  170656          MOV     WRDCNT,@RKWC    ;THIS # OF WORDS
2255  010602  013777  001434  170652          MOV     BUSADR,@RKBA    ;INTO THIS BUFR
2256
2257  010610  012777  000405  170640          MOV     #405,@RKCS      ;READ,SSE,GO
2258
2259
2260  010616  013737  001406  001446          MOV     PBUF1,BUFNO     ;SET UP STARTING ADRES
2261  010624  017737  170574  001430          MOV     @PRSPAT,PGSUBR
2262  010632  004777  170572                  JSR     PC,@PGSUBR      ;GO GENERATE A BUFFER
```

```
                                                        ;OF 1400 WORDS USING THIS
                                                        ;PATRN GENRATR

010636  104421                  CON,RDY                 ;DONE WITH PATRN GENRTNG,
                                                        ;WAIT FOR CNT RDY TO SET
                                                        ;(FROM PREVIOUS READ)

                                                        ;CNT RDY SET
010640  032777  040000  170610          BIT     #BIT14,@RKCS    ;HARD ERROR?
010646  001416                  BEQ     NOHE
                                
010650  012737  010564  001110          MOV     #LUPHE,$LPERR
010656  004737  016140                  JSR     PC,GETINF
010662  104023                  ERROR   23              ;HARD ERROR
010664  104415                  CON,RESET
010666  005237  001510                  INC     ERCNT3          ;ALLOW 12 ERRORS AT MOST
010672  001002                  BNE     1$
010674  000137  012114                  JMP     EXT7            ;IF MORE, EXIT
010700  000137  011436          1$:     JMP     FINISH
010704  032777  001000  170540  NOHE:   BIT     #BIT9,@RKDS     ;SIN SET?
010712  001417                  BEQ     NOSIN           ;NO

010714  012737  010556  001110          MOV     #LUPSIN,$LPERR
010722  004737  016140                  JSR     PC,GETINF
010726  104011                  ERROR   11              ;SIN ON READ
010730  104415                  CON,RESET
010732  104416                  DRV,RESET
010734  005237  001512                  INC     ERCNT4          ;ALLOW 5 ERRORS AT MOST
010740  001002                  BNE     1$
010742  000137  012114                  JMP     EXT7            ;IF MORE, EXIT
010746  000137  011436          1$:     JMP     FINISH

010752  005737  001504          NOSIN:  TST     ERCNT1          ;CHECKING CSE FOR 1ST TIME
010756  001031                  BNE     WRTCHK          ;NO,BRANCH
010760  005237  001504                  INC     ERCNT1          ;INDICATE THAT CSE HAS BEEN
                                                        ;CHECKED ONCE

010764  032777  000002  170462          BIT     #BIT1,@RKER     ;CHECK SUM EROR?
010772  001423                  BEQ     WRTCHK
010774  012737  010456  001110          MOV     #BEGIN,$LPERR
011002  004737  016140                  JSR     PC,GETINF
011006  013737  001252  001202          MOV     RETRY1,$REG10   ;GET THE RETRY #
011014  062737  000004  001202          ADD     #4,$REG10       ;SAVE IT FOR TYPEOUT
011022  104024                  ERROR   24              ;CSE
011024  005237  001514                  INC     ERCNT5          ;ALLOW 10 ERRORS AT MOST
011030  001002                  BNE     1$
011032  000137  012114                  JMP     EXT7            ;IF MORE, EXIT
011036  000137  011600          1$:     JMP     CSEROR          ;GO, SERVICE CSE

011042  005037  001506          WRTCHK: CLR     ERCNT2          ;CLR FLAG INDICATING SOFTWARE
                                                        ;COMPARE DONE
011046  022737  000003  001256          CMP     #3,RETRY3       ;WRT CHK DONE BEFORE OR
011054  001016                  BNE     SFTCMP          ;IT'S 1ST TIME?
                                                        ;IF DONE,BRANCH OTHERWISE DO IT
011056  013777  001440  170400  WCAGAIN: MOV    WDSKAD,@RKDA    ;WRT CHK FROM THIS DSK-ADRES
011064  013777  001444  170366          MOV     WWRDCN,@RKWC    ;THIS # FO WORDS
```



```
011072  013777  001442  170362          MOV     WBUSAD,@RKBA    ;WITH THIS BUFFER

011100  012777  000407  170350          MOV     #407,@RKCS      ;WRT CHK,GO,SSE

011106  005337  001256                  DEC     RETRY3          ;INDICATE WRT CHK DONE

011112  005737  001506          SFTCMP: TST     ERCNT2          ;SFTWARE CMPARE DONE ONCE BEFORE?
011116  001060                  BNE     WCREPT          ;IF SFTWARE CMPARE HAS BEEN DONE
                                                        ;ONCE DON'T DO IT AGAIN. OTHERWISE,
011120  005237  001506                  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

011124  012702  001266                  MOV     #BUFR,R2        ;INITLZE PTR TO BUFR STORING
                                                        ;ADRES OF BAD DATA
011130  012703  001320                  MOV     #BUFR1,R3       ;STORE EXPCTD DATA STARTING HERE
011134  012704  001352                  MOV     #BUFR2,R4       ;STORE RECVD (BAD) DATA
                                                        ;STARTING HERE

011140  005037  001500                  CLR     INDX3           ;CLR FLAG INDICATING MISCMPRE

011144  013700  001404          COMPAR: MOV     PBUF0,R0        ;INITLZE PTR TO 'RECVD DATA' BUFR
011150  012737  177764  001502          MOV     #-14,INDX4      ;STORE AND REPORT 12 OR LESS DATA ERRORS
011156  013701  001406                  MOV     PBUF1,R1        ;INITLZE PTR TO 'EXPCTD DATA' BUFR
011162  012737  176400  001260          MOV     #-1400,INADR    ;SET COUNT
011170  021011          CMPAGAN: CMP    (R0),(R1)       ;CORRECT WORD READ FROM DISK?
011172  001402                  BEQ     1$
011174  000137  011742                  JMP     MISCMP          ;BRANCH IF MISCMPRE ERROR
011200  005720          1$:     TST     (R0)+           ;INCRMNT PTRS
011202  005721                  TST     (R1)+           ;TO NXT WORDS
011204  005237  001260                  INC     INADR           ;DONE WITH CMPRISON?
011210  001367                  BNE     CMPAGAN         ;IF NOT, COMPARE THE REST

011212  005737  001500                  TST     INDX3           ;WAS THERE A BAD DATA WORD
                                                        ;(EVEN AFTR RETRYING)
011216  001420                  BEQ     WCREPT          ;NO, BRANCH
011220  012737  010532  001110  REPMSC: MOV     #READ,$LPERR
011226  104025                  ERROR   25              ;DATA ERROR
011230  005237  001516                  INC     ERCNT6          ;ALLOW 10 ERRORS AT MOST
011234  001002                  BNE     1$
011236  000137  012114                  JMP     EXT7            ;IF MORE, EXIT
011242  005737  001254          1$:     TST     RETRY2
011246  001404                  BEQ     WCREPT
011250  005237  001254                  INC     RETRY2
011254  000137  010532                  JMP     READ

011260  104421          WCREPT: CON,RDY                 ;WAIT FOR CNTRL RDY FROM
                                                        ;PREVIOUS WRT CHK
011262  022737  177776  001254          CMP     #-2,RETRY2      ;IF THERE WAS A RETRY AFTER MISC
                                                        ;=OMPARISON, DO WRT CHK AGAIN
011270  001417                  BEQ     ERWCHK
011272  000401                  BR      LUPWCE+2
```

```
2375  011274  104415                    LUPWCE: CON.RESET
2376  011276  013777  001440  170160            MOV     WDSKAD,@RKDA    ;WRT CHK WITH THIS DSK-ADRES
2377  011304  013777  001444  170146            MOV     WWRDCN,@RKWC    ;THIS # OF WORDS
2378  011312  013777  001442  170142            MOV     WBUSAD,@RKBA    ;THIS BUS ADRES
2379  011320  012777  000407  170130            MOV     #407,@RKCS
2380
2381  011326  104421                            CON.RDY
2382  011330  012737  011274  001110  ERWCHK:   MOV     #LUPWCE,$LPERR
2383  011336  032777  040000  170106            BIT     #BIT14,@RKDS    ;HARD EROR?(FROM WRT CHK)
2384  011344  001410                            BEQ     XHE             ;NO,BRANCH
2385
2386  011346  004737  016140                    JSR     PC,GETINF
2387  011352  104026                            ERROR   26              ;HE ON WRT CHK
2388  011354  005237  001520                    INC     ERCNT7          ;ALLOW 12 ERRORS AT MOST
2389  011360  001002                            BNE     XHE
2390  011362  000137  012114                    JMP     EXT7            ;IF MORE, EXIT
2391
2392  011366  032777  000001  170060  XHE:     BIT     #BIT0,@RKER     ;WRITE CHECK EROR?
2393  011374  001420                            BEQ     FINISH
2394  011376  004737  016140                    JSR     PC,GETINF
2395  011402  012737  000003  001202            MOV     #3,$REG10       ;GET TRY #
2396  011410  163737  001256  001202            SUB     RETRY3,$REG10   ;SAVE IT FOR TYPEOUT
2397  011416  104027                            ERROR   27              ;WRT CHK EROR
2398  011420  005237  001522                    INC     ERCNT8          ;ALLOW 10 ERRORS AT MOST
2399  011424  001002                            BNE     1$
2400  011426  000137  012114                    JMP     EXT7            ;IF MORE, EXIT
2401  011432  000137  012006            1$:     JMP     WCEROR
2402
2403
2404                                                                    ;THERE WAS NO WCE, DONE
2405                                                                    ;WITH ALL CHECKING FOR SOFT
2406                                                                    ;ERORS,ETC. MODIFY PARAMETERS
2407                                                                    ;TO CHECK NXT BLOCK ON
2408                                                                    ;THE DISK
2409
2410  011436  022737  001422  001424  FINISH: CMP     #PAT3,PRSPAT    ;FIND OUT THE NXT PATRN GENRATR
2411  011444  001404                            BEQ     1$              ;TO USE FOR GENRATING THE
2412  011446  062737  000002  001424            ADD     #2,PRSPAT       ;BUFR,STORE POINTER TO
2413  011454  000403                            BR      2$              ;GENRATR ROUTINE IN 'PRSPAT'
2414  011456  012737  001414  001424  1$:      MOV     #PAT0,PRSPAT    ;NOTE THER R 4 PAT-GENRATRS
2415                                                                    ;PAT0-PAT1-PAT2-PAT3----PAT0-
2416  011464  013701  001476            2$:     MOV     INDX2,R1        ;FORM SECTR # TO BE USED NEXT
2417  011470  116101  010352                    MOVB    SECPTR(R1),R1   ;GET SECTR #
2418  011474  042737  000017  001450  3$:      BIC     #17,ADRES       ;MASK SECTR BITS
2419  011502  050137  001450                    BIS     R1,ADRES        ;FORM THE NEW DISK-ADRES
2420                                                                    ;FORM THE NXT BLOCK TO BE
2421                                                                    ;CHECKED.
2422  011506  005237  001476                    INC     INDX2           ;DONE ALL 4 BLOCKS(3 SECS
2423                                                                    ;EACH) ON THIS TRACK?
2424  011512  022737  000004  001476            CMP     #4,INDX2
2425  011520  001025                            BNE     GOBAK           ;IF NOT, GO BAK & DO THE
2426                                                                    ;NXT BLOCK ON THIS TRACK
2427
2428  011522  005037  001476            DONTRK: CLR     INDX2           ;REINITLZE COUNT FOR
2429                                                                    ;4 BLOCKS ON THIS TRACK
2430  011526  032737  000001  001474            BIT     #BIT0,INDX1     ;WHICH SUR TO DO NXT? 0 OR 1
```



```
2431  011534  001407                            BEQ     DOSUR1          ;BRANCH, DO SUR 1 NXT
2432
2433  011536  062737  000040  001450            ADD     #40,ADRES
2434
2435  011544  042737  000020  001450            BIC     #20,ADRES       ;CLR SUR BIT, DO SUR 0 NXT
2436  011552  000403                            BR      DONE
2437
2438  011554  052737  000020  001450  DOSUR1: BIS     #20, ADRES      ;SET SUR BIT, DO SUR 1 NXT
2439
2440  011562  005237  001474            DONE:   INC     INDX1           ;DONE WITH ALL 626 TRACKS?
2441  011566  001002                            BNE     GOBAK
2442  011570  000137  012120                    JMP     TST10
2443
2444  011574  000137  010456            GOBAK:  JMP     BEGIN           ;IF NOT, GO BAK & CHECK
2445                                                                    ;THE NXT TRACK
2446
2447                                    ;CSEROR                         ;THIS IS THE ENTRY POINT
2448                                                                    ;FOR SERVICE ROUTINE FOR CHECK-
2449  011600                            CSEROR:                         ;SUM EROR. CONTROL IS XFERED
2450                                                                    ;HERE ONCE CSE OCCURS
2451
2452  011600  017700  167660                    MOV     @RKDA,R0        ;GET RKDA AFTER CSE
2453  011604  010001                            MOV     R0,R1           ;SAVE RKDA
2454  011606  032700  000017                    BIT     #17,R0          ;FORM THE ADRES OF THE SECTR
2455  011612  001002                            BNE     1$              ;WHICH GAVE CSE
2456  011614  162700  000004                    SUB     #4,R0
2457  011620  005300                    1$:     DEC     R0              ;R0 CONTAINS DSK-ADRES WHERE
2458                                                                    ;CSE OCURED
2459  011622  020037  001432                    CMP     R0,DSKADR       ;DID A PREVIOUS RETRY (IF ANY)
2460                                                                    ;GIVE CSE ON SAME ADRES
2461  011626  001021                            BNE     2$              ;NO, THIS IS A FRESH CSE, BRANCH.
2462                                                                    ;IF THIS WAS A CSE ON A
2463  011630  005237  001252                    INC     RETRY1          ;RETRY INCMNT RETRY COUNT.
2464                                                                    ;BRANCH IF 3 RETRIES HAVEN'T
2465  011634  001021                            BNE     3$              ;BEEN DONE
2466                                                                    ;GO REPORT CSE
2467                                                                    ;IF CSE WAS IN THE LAST SECTR OF
2468                                                                    ;THE 3 SEC BLOCK, GO TO 'WRTCHK'
2469                                                                    ;OTHERWISE CHECK THE REST OF
2470                                                                    ;THE SECTORS FOR CSE.
2471  011636  010102                            MOV     R1,R2
2472  011640  042702  177760                    BIC     #177760,R2
2473  011644  001002                    7$:     BNE     8$
2474  011646  000137  011042                    JMP     WRTCHK
2475  011652  162702  000003            8$:     SUB     #3,R2
2476  011656  100372                            BPL     7$
2477
2478  011660  010100                    6$:     MOV     R1,R0
2479  011662  012737  177775  001252            MOV     #-3,RETRY1
2480  011670  000403                            BR      3$
2481
2482  011672  012737  177776  001252  2$:      MOV     #-2,RETRY1      ;ALLOW 2 MORE TRIES FOR
2483                                                                    ;THE CSE ON SAME DISK-ADRES
2484
2485  011700  010037  001432            3$:     MOV     R0,DSKADR       ;SAVE DSK-ADRES FOR DOING
2486                                                                    ;THE RETRY-READ
```

```
2487  011704  163700  001450          SUB     ADRES,R0        ;MODIFY THE
2488  011710  005200                  INC     R0              ;BUS ADRES & WORD COUNT
2489  011712  005300          4$:     DEC     R0              ;TO BE USE ON RETRY-
2490  011714  001407                  BEQ     5$              ;READ
2491  011716  062737  001000  001434  ADD     #1000,BUSADR
2492  011724  062737  000400  001436  ADD     #400,WRDCNT
2493  011732  000767                  BR      4$
2494
2495  011734  104415          5$:     CON.RESET               ;CLR THE C&E IN RKER
2496  011736  000137  010566          JMP     RDAGAIN         ;GO BACK, READ AGAIN
2497                                                          ;RETRY
2498
2499
2500
2501                          ;MISCMP                         ;THIS IS THE ENTRY POINT
2502                                                          ;FOR SERVICE ROUTINE, FOR
2503                                                          ;DATA EROR (MISCOMPARISON)
2504                                                          ;ON SOFTWARE COMPARISON
2505                                                          ;OF DATA READ FROM THE DISK BLOCK
2506
2507
2508  011742  104421          MISCMP: CON.RDY                 ;WAIT FOR CNTRL RDY FROM
2509                                                          ;PREVIOUS WRT CHK
2510  011744  032777  040000  167504  BIT     #BIT14,@RKCS
2511  011752  001401                  BEQ     1$
2512  011754  104415                  CON.RESET               ;CLR WCE IN RKER
2513                                                          ;BUT STILL DATA EROR ENTER HERE
2514  011756  010022          1$:     MOV     R0,(R2)+        ;STORE MEM ADRES WHERE
2515                                                          ;DATA EROR OCCURED
2516  011760  012123                  MOV     (R1)+,(R3)+     ;SAVE EXPCTD DATA
2517  011762  012024                  MOV     (R0)+,(R4)+     ;SAVE DATA RECVD (BAD)
2518  011764  005237  001500          INC     INDX3           ;INDICATE MISCMPRISON
2519
2520  011770  005237  001502          INC     INDX4           ;STORE ONLY 12(DEC) ERORS
2521  011774  001402                  BEQ     4$              ;IF 12 ERORS, GO REPORT THEM
2522
2523  011776  000137  011170          JMP     CMPAGAN         ;GO BACK & CMPARE THE REST
2524
2525  012002  000137  011220  4$:     JMP     REPMSC
2526
2527
2528
2529                                                          ;THIS THE ENTRY POINT FOR
2530                                                          ;EROR SERVICE ON GETTING A
2531                                                          ;WRITE CHECK EROR.
2532
2533  012006  005737  001256  WCEROR: TST     RETRY3          ;DONE 3 TRIES?
2534  012012  003035                  BGT     CLRERR          ;IF NOT SKIP, OTHERWISE REPORT
2535                                                          ;W C EROR
2536
2537  012014  012737  000003  001256  MOV     #3,RETRY3       ;SET CONT FOR RETRIES
2538
2539  012022  017700  167436          MOV     @RKDA,R0        ;IF WCE WAS IN THE LAST SECTOR
2540  012026  010002                  MOV     R0,R2           ;OF THE BLOCK, NO MORE SECS
2541  012030  042702  177760          BIC     #177760,R2      ;TO CHECK, GO TO 'FINISH'
2542  012034  001002          3$:     BNE     4$              ;IF IT WASN'T LAST SECTOR OF THE
```



```
2543  012036  000137  011436          JMP     FINISH          ;BLOCK, THEN CHECK REMAINING
2544  012042  162702  000003  4$:     SUB     #3,R2           ;SECTORS. (STARTING FROM THE
2545  012046  100372                  BPL     3$              ;SEC AFTER THE ONE GIVING WCE)
2546  012050  010001          1$:     MOV     R0,R1           ;SAVE DISK ADRES
2547  012052  163700  001440          SUB     WDSKAD,R0
2548  012056  010137  001440          MOV     R1,WDSKAD       ;GET SAVED DISK ADRES
2549  012062  005200                  INC     R0
2550  012064  005300          2$:     DEC     R0
2551  012066  001407                  BEQ     CLRERR
2552  012070  062737  001000  001442  ADD     #1000,WBUSAD    ;FORM THE NEW BUS ADRES
2553  012076  062737  000400  001444  ADD     #400,WWRDCN     ;FORM THE NEW WORD COUNT
2554  012104  000767                  BR      2$
2555  012106  104415  CLRERR: CON.RESET
2556  012110  000137  011056          JMP     WCAGAIN
2557
2558  012114  004737  016716  EXT7:   JSR     PC,ABRT
2559
2560
2561                          ;;**********************************************************
2562                          ;*TEST 10        WRITE, WRITE CHECK ON CYLINDERS 127, 128
2563                                  ;*THIS TEST WRITES 12 UNIQUE PATTERNS (ONE FOR EACH
2564                                  ;*SECTOR) ON CYLINDERS 127 AND 128, THEN WRITE
2565                                  ;*CHECK IS DONE TO SEE IF THEY WERE WRITTEN
2566                                  ;*CORRECTLY.  IT SHOULD BE NOTED THAT THERE IS
2567                                  ;*CHANGE IN 'WRITE' CURRENT AT THIS CYLINDER.
2568                                  ;*PATTERNS ARE RELOCATED ON THE CYLINDERS AFTER EACH
2569                                  ;*WRITE/WRITE CHECK CYCLE.  THUS THE FIRST TIME
2570                                  ;*PATTERN 'SP1' IS WRITTEN ON SECTOR 0, PATTERN 'SP2' ON
2571                                  ;*SECTOR 2, ETC.  AFTER THIS WRITE/WRITE CHECK CYCLE
2572                                  ;*IS OVER PATTERN 'SP2' IS WRITTEN ON SECTOR 0, PATTERN
2573                                  ;*'SP3' ON SECTOR 1 AND SO ON.  THIS WRITE/WRITE
2574                                  ;*CHECK CYCLE IS REPEATED 12 TIMES, THUS
2575                                  ;*THE LAST WRITE/WRITE CHECK IS DONE USING
2576                                  ;*PATTERN 'SP12' ON SECTOR 0, PATTERN 'SP1' IS
2577                                  ;*WRITTEN ON SECTOR 1, PATTERN 'SP2' ON SECTOR 2,
2578                                  ;*ETC.  IF YOU WANT TO WRITE ANY OTHER PATTERNS
2579                                  ;*FILL IN THE PATTERNS YOU WANT IN LOCATIONS
2580                                  ;*'SP1', 'SP2',...ETC.
2581                          ;;**********************************************************
2582  012120  000004          TST10:  SCOPE
2583
2584  012122  012737  177764  001504  MOV     #-14,ERCNT1     ;ALLOW 12 ERRORS AT MOST
2585  012130  012737  177764  001506  MOV     #-14,ERCNT2     ;ALLOW 12 ERRORS AT MOST
2586  012136  012737  177723  001510  MOV     #-55,ERCNT3     ;ALLOW 15 ERRORS AT MOST
2587  012144  012702  012626          MOV     #SP1,R2         ;INITIALIZE POINTER TO PATRN
2588  012150  010201  DOWRT:  MOV     R2,R1
2589  012152  012700  007740          MOV     #7740,R0        ;SET UP CYL ADRES BITS (127)
2590  012156  053700  001230          BIS     DRIVAD,R0       ;SET UP DRIVE # BITS
2591  012162  005003  WRLO1:  CLR     R3
2592  012164  104415  WRERR:  CON.RESET
2593
2594  012166  010077  167272  WRLO:   MOV     R0,@RKDA        ;ADRES THE DRIVE
2595  012172  012777  177400  167260  MOV     #-400,@RKWC     ;WRITE 1 SECTOR
2596  012200  010177  167256          MOV     R1,@RKBA        ;USE THIS PATTERN
2597  012204  012777  004003  167244  MOV     #4003,@RKCS     ;WRITE, GO
2598  012212  104421                  CON.RDY
```

```
2599  012214  032777  140000  167234          BIT     #140000,@RKCS   ;ANY ERROR?
2600  012222  001414                          BEQ     4$
2601  012224  012737  012164  001110          MOV     #WRERR,$LPERR   ;SET ADRES FOR LOOPING ON ERROR
2602  012232  004737  016106                  JSR     PC,GT4RG        ;GET TKCS, ER, DS, DA
2603  012236  104021                          ERROR   21              ;ERROR OCCURRED ON DOING A WRITE
2604  012240  104415                          CON.RESET               ;CLEAR THE ERROR
2605  012242  005237  001504                  INC     ERCNT1          ;ALLOW 12 ERRORS ONLY
2606  012246  001002                          BNE     4$
2607  012250  000137  012656                  JMP     EXT10
2608
2609  012254  005200                  4$:     INC     R0
2610  012256  005203                          INC     R3              ;KEEP COUNT
2611  012260  022701  012654                  CMP     #SP12,R1        ;USE PATTERNS IN A CYCLIC FASHION
2612  012264  001002                          BNE     3$
2613  012266  012701  012624                  MOV     #SP1-2,R1
2614  012272  005721                  3$:     TST     (R1)+           ;INCREMENT POINTERS TO NEXT PATTERN
2615  012274  020327  000014                  CMP     R3,#14          ;DONE SURFACE 0?
2616  012300  002732                          BLT     WRLO            ;NO
2617  012302  001005                          BNE     2$              ;YES, IF CHANGING HEADS
2618  012304  010201                          MOV     R2,R1
2619  012306  042700  000017                  BIC     #17,R0          ;SET UP CORRECT ADDRESS ETC.
2620  012312  052700  000020                  BIS     #20,R0
2621
2622  012316  020327  000030          2$:     CMP     R3,#30          ;DONE WITH WRITING SURFACE 1?
2623  012322  001321                          BNE     WRLO            ;NO, BRANCH
2624
2625  012324  032700  007700          WRHI:   BIT     #7700,R0        ;DONE WITH BOTH CYLINDERS - 127 & 128?
2626  012330  001405                          BEQ     DOWCHK          ;YES
2627  012332  012700  010000                  MOV     #10000,R0       ;NO, DO CYLINDER 128
2628  012336  053700  001230                  BIS     DRIVAD,R0
2629  012342  000707                          BR      WRLO1           ;GO BACK
2630                                                                  ;CYLINDERS 127 AND 128 HAVE BEEN
2631
2632
2633  012344  010201                  DOWCHK: MOV     R2,R1           ;WRITTEN, NOW DO WRITE CHECK
2634  012346  012737  177775  001252          MOV     #-3,RETRY1      ;INITIALIZE POINTER TO FIRST PATTERN
2635  012354  012700  010000                  MOV     #10000,R0       ;RETRY COUNT
2636  012360  053700  001230                  BIS     DRIVAD,R0       ;DO CYLINDER 128 FIRST
2637  012364  005003                  WCHI1:  CLR     R3
2638  012366  010077  167072          WCERR:  MOV     R0,@RKDA        ;ADRES THE DRIVE
2639
2640  012372  012777  177400  167060  WCHI:   MOV     #-400,@RKWC     ;WRITE CHECK 1 SECTOR
2641  012400  010177  167056                  MOV     R1,@RKBA        ;WITH THIS PATTERN
2642  012404  012777  004007  167044          MOV     #4007,@RKCS     ;WRITE CHECK, GO
2643  012412  104421                          CON.RDY
2644
2645  012414  032777  040000  167030          BIT     #40000,@RKDS    ;HE?
2646  012422  001406                          BEQ     1$              ;NO
2647  012424  004737  016140                  JSR     PC,GETINF
2648  012430  104026                          ERROR   26              ;HE ON DOING WRT CHK
2649  012432  005237  001506                  INC     ERCNT2          ;ALLOW 12 ERRORS ONLY
2650  012436  001507                          BEQ     EXT10           ;IF MORE, EXIT
2651  012440  032777  000001  167006  1$:     BIT     #BIT0,@RKER     ;WCE?
2652  012446  001425                          BEQ     4$              ;NO
2653  012450  012737  012366  001110          MOV     #WCERR,$LPERR   ;SET ADRES FOR LOOPING ON ERROR
2654  012456  004737  016140                  JSR     PC,GETINF       ;GET INFO ON ERROR
```



```
2655  012462  013737  001252  001202          MOV     RETRY1,$REG10
2656  012470  062737  000004  001202          ADD     #4,$REG10       ;WCE ON DOING WRITE CHECK, WITH
2657  012476  104027                          ERROR   27              ;PATTERN STORED IN R1
2658  012500  005237  001252                  INC     RETRY1          ;DO 3 TIMES IN ALL
2659  012504  001330                          BNE     WCERR
2660  012506  005237  001510                  INC     ERCNT3          ;ALLOW 15 ERRORS ONLY
2661  012512  001461                          BEQ     EXT10           ;IF MORE, EXIT
2662  012514  012737  177775  001252          MOV     #-3,RETRY1
2663
2664  012522  005200                  4$:     INC     R0              ;KEEP TRACK OF DISK-ADRES
2665  012524  005203                          INC     R3              ;AND COUNT
2666  012526  022701  012654                  CMP     #SP12,R1        ;USE PATTERNS IN CYCLIC
2667  012532  001002                          BNE     3$
2668  012534  012701  012624                  MOV     #SP1-2,R1       ;FASHION
2669  012540  005721                  3$:     TST     (R1)+           ;INCREMENT POINTER TO NEXT PATTERN
2670  012542  020327  000014                  CMP     R3,#14          ;DONE SURFACE 0?
2671  012546  002711                          BLT     WCHI            ;NO
2672  012550  001005                          BNE     2$
2673  012552  010201                          MOV     R2,R1           ;IF CHANGING HEADS (0-1), SET CORRECT
2674  012554  042700  000017                  BIC     #17,R0          ;ADRES BITS
2675  012560  052700  000020                  BIS     #20,R0
2676
2677  012564  020327  000030          2$:     CMP     R3,#30          ;DONE WRITE CHECKING SURFACE 1?
2678  012570  001300                          BNE     WCHI            ;NO, GO BACK
2679
2680  012572  032700  007700          WCLO:   BIT     #7700,R0        ;DONE BOTH CYLINDERS - 127, 128?
2681  012576  001005                          BNE     REPEAT          ;YES, BRANCH
2682  012600  012700  007740                  MOV     #7740,R0        ;DO CYLINDER 127 NOW
2683  012604  053700  001230                  BIS     DRIVAD,R0
2684  012610  000665                          BR      WCHI1
2685
2686
2687  012612  005722                  REPEAT: TST     (R2)+           ;RELOCATE THE PATTERNS ON THE
2688  012614  020227  012656                  CMP     R2,$SP12+2      ;CYLINDERS AND DO IT AGAIN
2689  012620  001420                          BEQ     TST11           ;;EXIT
2690  012622  000137  012150                  JMP     DOWRT           ;THIS TEXT)
2691
2692
2693
2694                                  ;PATTERNS TO BE USED
2695
2696  012626  177777                  SP1:    .WORD   177777          ;IF YOU WANT TO WRITE ANY
2697  012630  052525                  SP2:    .WORD   052525          ;OTHER PATTERNS, CHANGE THESE
2698  012632  111111                  SP3:    .WORD   111111          ;12 LOCATIONS TO ANY PATTERN
2699  012634  010421                  SP4:    .WORD   010421          ;YOU WANT.
2700  012636  102041                  SP5:    .WORD   102041
2701  012640  010101                  SP6:    .WORD   010101
2702  012642  040201                  SP7:    .WORD   040201
2703  012644  000401                  SP8:    .WORD   000401
2704  012646  031463                  SP9:    .WORD   031463
2705  012650  070707                  SP10:   .WORD   070707
2706  012652  007417                  SP11:   .WORD   007417
2707  012654  041020                  SP12:   .WORD   041020
2708
2709  012656  004737  016716          EXT10:  JSR     PC,ABRT
2710
```

```
2711
2712
2713
2714
2715                                    ;;*****************************************************************
2716                                    ;*TEST 11        SEEK FUNCTION TIMER
2717                                    ;*SEEK TIMER
2718                                    ;*IN THIS PART OF THE PROGRAM SEEKS ARE TIMED BETWEEN A PARTICULAR SET
2719                                    ;*OF CYLINDERS, BOTH IN THE FORWRAD  DIRECTION (0-312) AND REVERSE(312-0).
2720                                    ;*******CAUTION******
2721                                    ;*IT SHOULD BE NOTED THAT THE SECTOR COUNTER (IN RKDS) IS USED AS THE REAL
2722                                    ;*TIME CLOCK TO DO THE SEEK TIMING. FOR THE TIMES TO BE RELIABLE, THE
2723                                    ;*SECTOR COUNTER SHOULD BE ACCURATE WITHIN THE SPECIFICATIONS OF THE DISK
2724                                    ;*SPEED:     1500 RPM = 40 MILI SECS/REV =3.33 MILI SECS/SECTOR.
2725                                    ;*VARIATION: +-30  RPM
2726                                    ;;*****************************************************************
2727  012662  000004                    TST11:  SCOPE
2728  012664  032777  000100  166246            BIT     #SW6,@SWR       ;INHIBIT TIMER?
2729  012672  001002                            BNE     .+6
2730  012674  000137  014032                    JMP     PLTGRPH
2731  012700  104415                            CON.RESET
2732  012702  104416                            DRV.RESET
2733
2734  012704  012737  177771  001252          MOV     #-7,RETRY1      ;COUNT FOR 7 DIFRNT SEEK TIMES
2735                                                                    ;TO BE RECORDED
2736
2737  012712  104401  012720                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2738  012716  000424                          BR      64$             ;;GET OVER THE ASCIZ
2739                                    ;;65$:  .ASCIZ  <15><12>/SEEK TIME SCALE FACTOR=0,01 MILI SECS/
2740  012770                            64$:
2741  012770  104401  012776                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2742  012774  000421                          BR      66$             ;;GET OVER THE ASCIZ
2743                                    ;;67$:  .ASCIZ  <15><12><12>/ #  OF    SEEK   #  OF   SEEK/
2744  013040                            66$:
2745  013040  104401  013046                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2746  013044  000421                          BR      68$             ;;GET OVER THE ASCIZ
2747                                    ;;69$:  .ASCIZ  <15><12>/ SEEKS    TIME   SEEKS   TIME/<15><12>
2748  013110                            68$:
2749
2750  013110  012737  001542  001260          MOV     #SIAD,INADR     ;INITLZE PTR TO INNER ADRES
2751  013116  012737  001524  001262          MOV     #SOAD,OUTADR    ;INITLZE PTR TO OUTER ADRES
2752
2753  013124  017777  166132  166332  REPTIM: MOV     @OUTADR,@RKDA   ;POSITION HEADS TO OUTER CYLINDER
2754                                                                  ;BEFORE STARTING TO TIME
2755  013132  053777  001230  166324          BIS     DRIVAD,@RKDA    ;SET DRIVE # BITS
2756  013140  012777  000011  166310          MOV     #11,@RKCS               ;SEEK, GO
2757  013146  104421                          CON.RDY                 ;WAIT FOR CNTRL RDY
2758  013150  104422                          TST.RWS                 ;WAIT FOR R/W/S RDY
2759
2760  013152  005037  001474                  CLR     INDX1           ;INDX1 = 0, GOING IN; OTHERWISE OUT
2761  013156  012704  026742                  MOV     #BUFR10,R4      ;STORE FWRD SEEK TIMES IN THIS BUFR
2762  013162  012705  027342                  MOV     #BUFR11,R5      ;STORE REVRSE "   "
2763  013166  012737  177634  001476          MOV     #-144,INDX2     ;SET COUNT FOR # OF SEEKS
2764
2765
2766  013174  013702  001464          BEGSK:  MOV     RKDA,R2
```



```
2767  013200  005737  001474                  TST     INDX1           ;GOING FWRD OR REVRSE?
2768  013204  001005                          BNE     1$              ;REVRSE, BRANCH
2769
2770  013206  017712  166046                  MOV     @INADR,@R2      ;FWRD, SET INNER CYL ADRES
2771  013212  053712  001230                  BIS     DRIVAD,@R2      ;SET DRIVE # BITS
2772  013216  000404                          BR      2$
2773
2774  013220  017712  166036          1$:     MOV     @OUTADR,@R2     ;SET OUTER CYL ADRES
2775  013224  053712  001230                  BIS     DRIVAD,@R2      ;SET DRIVE # BITS
2776  013230  004737  014722          2$:     JSR     PC,@#TIMSEK     ;GO, TIME THE SEEK FROM CYLINDER
2777                                                                  ;0 TO THE ABOVE CYL. RETURN WITH
2778                                                                  ;R3 CONTAINING THE TIME (MS, SCALE
2779                                                                  ;FACTOR= 0,01) REQUIRED FOR THE SEEK.
2780  013234  005737  001474                  TST     INDX1
2781  013240  001004                          BNE     3$
2782  013242  010324                          MOV     R3,(R4)+        ;STORE TIME TAKEN FOR FRWRD SEEK
2783  013244  005237  001474                  INC     INDX1           ;SET FLAG FOR DOING REVRSE SEEK
2784  013250  000751                          BR      BEGSK           ;GO DO IT
2785
2786  013252  010325                  3$:     MOV     R3,(R5)+        ;STORE TIME TAKEN FOR REVRSE SEEK
2787  013254  005037  001474                  CLR     INDX1           ;CLR FLG FOR DOING FRWRD SEEK
2788
2789  013260  005237  001476                  INC     INDX2           ;RECORDED 144 SEEK TIMES
2790  013264  001343                          BNE     BEGSK           ;IF NOT, GO BAK
2791
2792                                          ;AT THIS POINT 100 SEEKS HAVE BEEK PERFORMED BETWEEN TWO
2793                                          ;CYLINDERS (FORWARD & REVERSE DIRECTION).  FORWARD SEEK
2794                                          ;TIMES ARE STORED IN TABLE STARTING AT 'BUFR10' REVERSE
2795                                          ;STARTING AT 'BUFR11'.  THE FOLLOWING CODE FINDS OUT THE
2796                                          ;NUMBERS OF TIMES A PARTICULAR 'SEEK TIME' WAS OBTAINED.
2797                                          ;EXAMPLE: OUT OF 100 TIMES THE SEEK WAS DONE BETWEEN
2798                                          ;CYLINDER 0 & LAST.
2799                                          ;70 TIMES IT TOOK 95 MILI SECS
2800                                          ;20 TIMES IT TOOK 85 MILI SECS
2801                                          ;10 TIMES IT TOOK 100 MILI SECS
2802                                          ;THIS INDICATES HOW CONSISTENT WAS THE SEEKING TIME.
2803                                          ;NOTE THAT ONLY 5 DIFFERENT "SEEK TIMES" WILL BE TYPED
2804                                          ;OUT.
2805
2806                                          ;SORTING ROUTINE
2807
2808  013266  012705  177776          SORT:   MOV     #-2,R5          ;COUNT FOR FRWRD, REVRSE
2809  013272  012737  026742  001162          MOV     #BUFR10,@REG0   ;INTLZE PTR TO 'SEEK TIME'
2810  013300  012737  026744  001164          MOV     #BUFR10+2,@REG1
2811  013306  012737  026442  001166          MOV     #BUFR4,@REG2
2812  013314  012737  026522  001170          MOV     #BUFR5,@REG3    ;INTLZE PTR TO '# OF TIMES'
2813
2814  013322  013700  001162          1$:     MOV     @REG0,R0        ;PTR T 'SEEK TIME'
2815  013326  013701  001164                  MOV     @REG1,R1
2816  013332  012702  177635                  MOV     #-143,R2        ;COUNT FOR 143 ITEMS TO SORT
2817  013336  005003                          CLR     R3
2818
2819  013340  021011                  2$:     CMP     (R0),(R1)       ;SORT THE ITEMS & PUT THEM
2820  013342  003404                          BLE     3$              ;IN DESCENDING ORDER
2821  013344  011004                          MOV     (R0),R4         ;LARGER ITEMS AT TOP OF LIST,
2822  013346  011110                          MOV     (R1),(R0)       ;SMALLER AT THE BOTTOM
```

```
2823  013350  010411                        MOV     R4,(R1)
2824  013352  005203                        INC     R3
2825  013354  005720                3$:     TST     (R0)+
2826  013356  005721                        TST     (R1)+
2827  013360  005202                        INC     R2
2828  013362  001366                        BNE     2$
2829  013364  005703                        TST     R3              ;SORTED ALL ITEMS?
2830  013366  001355                        BNE     1$              ;IF NOT LOOP BACK
2831
2832  013370  013700  001162                MOV     $REG0,R0        ;PTR TO 'SEEK TIME'
2833  013374  013701  001166                MOV     $REG2,R1        ;SAVE 'SEEK TIME' HERE
2834  013400  013702  001170                MOV     $REG3,R2        ;SAVE '# OF TIMES' HERE
2835  013404  010204                        MOV     R2,R4
2836  013406  005024                        CLR     (R4)+           ;CLR OUT 5 WORDS OF
2837  013410  005024                        CLR     (R4)+           ;'# OF TIMES'BUFR
2838  013412  005024                        CLR     (R4)+
2839  013414  005024                        CLR     (R4)+
2840  013416  005024                        CLR     (R4)+
2841  013420  012703  177773                MOV     #-5,R3          ;SAVE ONLY 5 DIFRNT 'SEEK TIMES'
2842
2843  013424  011011                        MOV     (R0),(R1)       ;FIND OUT THE '# OF TIMES'
2844  013426  012703  177634                MOV     #-144,R3        ;EACH 'SEEK TIME' WAS
2845  013432  022011                4$:     CMP     (R0)+,(R1)
2846  013434  001411                        BEQ     5$              ;OBTAINED
2847
2848  013436  005721                        TST     (R1)+
2849  013440  016011  177776                MOV     -2(R0),(R1)     ;SAVE 'SEEK TIME'
2850  013444  005722                        TST     (R2)+
2851  013446  012712  000001                MOV     #1,(R2)         ;KEEP '# OF TIMES'
2852  013452  005203                        INC     R3
2853  013454  001404                        BEQ     6$
2854  013456  000765                        BR      4$
2855
2856  013460  005212                5$:     INC     (R2)            ;INCRMNT '# OF TIMES'
2857  013462  005203                        INC     R3              ;ALL DONE?
2858  013464  001362                        BNE     4$              ;IF NOT, GO BAK
2859
2860  013466  005205                6$:     INC     R5              ;SORTED BOTH FRWRD, REVRSE
2861  013470  001415                        BEQ     GOTYPE          ;'SEEK TIMES', IF YES GO TYPE
2862
2863  013472  012737  027342  001162        MOV     #BUFR11,$REG0   ;IF NOT, INITLZE PTR TO 'SEEK TIME'
2864  013500  012737  027344  001164        MOV     #BUFR11+2,$REG1
2865  013506  012737  026602  001166        MOV     #BUFR6,$REG2    ;SAVE 'SEEK TIME'
2866  013514  012737  026662  001170        MOV     #BUFR7,$REG3    ;SAVE '# OF TIMES' HERE
2867
2868  013522  000677                        BR      1$              ;GO BAK & DO SORTING FOR REVRSE SEEK TIMES
2869
2870                                                ;TYPE OUT CYL #'S BETWEEN WHICH SEEK
2871                                                ;WAS TIMED.  'OUTADR' TO 'INADR' 'INADR' TO 'OUTADR'
2872  013524  104401                GOTYPE: TYPE
2873  013526  001213                        $CRLF
2874  013530  104401  013536                TYPE    ,65$            ;;TYPE ASCIZ STRING
2875  013534  000403                        BR      64$             ;;GET OVER THE ASCIZ
2876                                ;;65$:  .ASCIZ  /CYLS:/
2877  013544                        64$:
2878
```



```
2879  013544  017700  165512                MOV     @OUTADR,R0      ;GET OUTER CYL #
2880  013550  006200                        ASR     R0
2881  013552  006200                        ASR     R0
2882  013554  006200                        ASR     R0
2883  013556  006200                        ASR     R0
2884  013560  006200                        ASR     R0
2885  013562  010046                        MOV     R0,-(SP)
2886  013564  104424                        TYPDSS                  ;TYPE IT OUT IN DECIMAL
2887  013566  104401  013574                TYPE    ,67$            ;;TYPE ASCIZ STRING
2888  013572  000401                        BR      66$             ;;GET OVER THE ASCIZ
2889                                ;;67$:  .ASCIZ  /-/
2890  013576                        66$:
2891  013576  017701  165456                MOV     @INADR,R1       ;GET INNER CYL #
2892  013602  006201                        ASR     R1
2893  013604  006201                        ASR     R1
2894  013606  006201                        ASR     R1
2895  013610  006201                        ASR     R1
2896  013612  006201                        ASR     R1
2897  013614  010146                        MOV     R1,-(SP)
2898  013616  104424                        TYPDSS                  ;TYPE IT OUT IN DECIMAL
2899  013620  104401  013626                TYPE    ,69$            ;;TYPE ASCIZ STRING
2900  013624  000405                        BR      68$             ;;GET OVER THE ASCIZ
2901                                ;;69$:  .ASCIZ  <15><12>/  FRWRD/
2902  013640                        68$:
2903  013640  104401  002101                TYPE    ,BLNKS9
2904  013644  104401  013652                TYPE    ,71$            ;;TYPE ASCIZ STRING
2905  013650  000404                        BR      70$             ;;GET OVER THE ASCIZ
2906                                ;;71$:  .ASCIZ  /REVRSE/
2907  013662                        70$:
2908
2909                                                ;TYPE OUT THE '# OF SEEKS' & 'SEEK
2910                                                ;TIME' OBTAINED FOR EACH OF THOSE
2911                                                ;SEEKS
2912  013662  005000                TYPTIM: CLR     R0
2913  013664  005005                        CLR     R5
2914  013666  104401                1$:     TYPE
2915  013670  001213                        $CRLF
2916  013672  016046  026522                MOV     BUFR5(R0),-(SP) ;GET '# OF SEEKS', IF NONE (0)
2917  013676  001424                        BEQ     3$              ;SKIP TYPING (FRWRD SEEK)
2918  013700  104405                        TYPDS                   ;GO TYPE OUT DECIMAL '# OF SEEKS'
2919  013702  104401                        TYPE
2920  013704  002110                        BLNKS2
2921  013706  016046  026442                MOV     BUFR4(R0),-(SP) ;GET 'SEEK TIME' FOR EACH OF
2922  013712  104405                        TYPDS                   ;OF THAT '# OF SEEKS'. 'GO
2923                                                ;TYPE OUT IN DECIMAL
2924
2925  013714  016046  026662          2$:   MOV     BUFR7(R0),-(SP) ;GET '# OF SEEKS', IF NONE (0)
2926  013720  001416                        BEQ     4$              ;SKIP TYPING (REVRSE SEEK)
2927  013722  005705                        TST     R5
2928  013724  001402                        BEQ     6$
2929  013726  104401  002075                TYPE    ,BLNK13
2930  013732  104405                6$:     TYPDS                   ;TYPE OUT IN DECIMAL
2931  013734  104401                        TYPE
2932  013736  002110                        BLNKS2
2933  013740  016046  026602                MOV     BUFR6(R0),-(SP) ;GET 'SEEK TIME' & TYPE IT
2934  013744  104405                        TYPDS                   ;OUT IN DECIMAL
```

```
2935  013746  000406                    BR      5$
2936
2937  013750  005726            3$:     TST     (SP)+           ;POP STACK
2938  013752  005205                    INC     R5
2939  013754  000757                    BR      2$
2940
2941  013756  005726            4$:     TST     (SP)+           ;POP STACK
2942  013760  005705                    TST     R5
2943  013762  001004                    BNE     TIMDON
2944
2945  013764  005720            5$:     TST     (R0)+           ;INCREMENT PTR TO TABLES
2946  013766  020027  000012            CMP     R0,#12          ;ALL DONE?
2947  013772  001335                    BNE     1$              ;IF NOT GO BAK
2948
2949  013774  062737  000002  001260  TIMDON: ADD     #2,INADR        ;INCRMNT POINTER TO NEXT
2950  014002  062737  000002  001262          ADD     #2,OUTADR       ;INNER & OUTER ADRES
2951  014010  005237  001252                  INC     RETRY1          ;ALL DONE?
2952  014014  001406                    BEQ     PLTGRPH
2953  014016  032777  000100  165114          BIT     #SW6,@SWR       ;INHIBIT TIMER? FURTHER ?
2954  014024  001402                    BEQ     PLTGRPH         ;YES, BRANCH
2955  014026  000137  013124            JMP     REPTIM          ;GO, BACK AND TIME REST
2956                                            ;OF SEEKS
2957
2958
2959
2960                                    ;PLOT GRAPH OF 'SEEK TIME' V/S 'CYLIDERS SEEKED'
2961
2962                                    ;PERFORM SEEK FROM CYLINDER 0 TO EVERY OTHER CYLINDER AND TIME IT,
2963                                    ;0       0,0     1,----0 312. NOTE 'SECTOR COUNTER' IS USED AS A READ
2964                                    ;TIME CLOCK TO TIME THERE SEEKS,  AFTER OBTAINING THE SEEK TIMES A
2965                                    ;GRAPH IS PLOTTED OF 'SEEK TIME' V/S 'CYLINDER'.
2966
2967                                    ;TIME THE SEEKS
2968  014032  032777  000040  165100  PLTGRPH: BIT #SW5,@SWR         ;SKIP THE GRAPH?
2969  014040  001002                    BNE     .+6
2970  014042  000137  015150            JMP     TST12           ;YES, BRANCH
2971  014046  104415                    CON,RESET
2972  014050  104416                    DRV,RESET
2973  014052  012737  177465  001500          MOV     #-313,INDX3     ;PERFORM 313 SEEKS 0-0,0-1,0-312
2974  014060  012704  026742                  MOV     #BUFR10,R4      ;STORE 'SEEK TIME' HERE
2975  014064  005037  001260                  CLR     INADR           ;CLR CYL ADRES BITS
2976
2977  014070  013777  001260  165366  1$:     MOV     INADR,@RKDA     ;ADRES THE RIGHT CYLINDER
2978  014076  053777  001230  165360          BIS     DRIVAD,@RKDA    ;ADRES THE RIGHT DRIVE
2979
2980  014104  004737  014722                  JSR     PC,TIMSEK       ;GO TIME THE SEEK FROM CYL 0
2981                                            ;TO THE ABOVE CYL, RETURN WITH
2982                                            ;R3 CONTAINING 'SEEK TIME' IN MS
2983                                            ;SCALE FACTOR OF 0.01
2984  014110  010324                    MOV     R3,(R4)+        ;STORE 'SEEK TIME'
2985  014112  042777  017777  165344          BIC     #17777,@RKDA            ;SEEK BACK TO CYL 0 FOR
2986  014120  012777  000011  165330          MOV     #11,@RKCS       ;TIMING NXT CYL SEEK
2987  014126  104421                    CON,RDY                 ;WAIT FOR CNTRL RDY?
2988  014130  104422                    TST,RWS                 ;WAIT FOR R/W/S RDY
2989  014132  062737  000040  001260          ADD     #40,INADR       ;FORM NXT CYL ADRES
2990  014140  005237  001500                  INC     INDX3
```



```
2991  014144  001351                    BNE     1$
2992
2993                                    ;PLOT A GRAPH USING 'SEEK TIMES' RECORDED BEFORE
2994
2995  014146                    PLOT:
2996  014146  104401  014154            TYPE    ,65$            ;;TYPE ASCIZ STRING
2997  014152  000422                    BR      64$             ;;GET OVER THE ASCIZ
2998                            ;;65$:  .ASCIZ  <15><12><12><12>/X AXIS - SEEK TIME - MILI SECS/
2999  014220                    64$:
3000  014220  104401  014226            TYPE    ,67$            ;;TYPE ASCIZ STRING
3001  014224  000423                    BR      66$             ;;GET OVER THE ASCIZ
3002                            ;;67$:  .ASCIZ  <15><12>/Y AXIS - CYLINDER SEEKED FROM 0/<15><12><12>
3003  014274                    66$:
3004
3005  014274  104401                    TYPE
3006  014276  002103                    BLNK$7
3007  014300  005000                    CLR     R0              ;TYPE OUT THE TIME UNITS
3008  014302  010046            1$:     MOV     R0,-(SP)        ;(MILI SECS) FOR THE X-AXIS
3009  014304  104424                    TYPD$$                  ;LIKE THIS:
3010  014306  005700                    TST     R0              ;0   20   30   40.....
3011  014310  001411                    BEQ     2$
3012  014312  022700  000144            CMP     #144,R0
3013  014316  003010                    BGT     4$
3014  014320  022700  000170            CMP     #170,R0
3015  014324  002412                    BLT     5$
3016  014326  104401                    TYPE
3017  014330  002110                    BLNK$2
3018  014332  000404                    BR      3$
3019  014334  104401            2$:     TYPE
3020  014336  002111                    BLNK$1
3021  014340  104401            4$:     TYPE
3022  014342  002107                    BLNK$3
3023  014344  062700  000012            ADD     #12,R0
3024  014350  000754                    BR      1$
3025
3026  014352  104401            5$:     TYPE
3027  014354  001213                    $CRLF
3028  014356  104401                    TYPE
3029  014360  002103                    BLNK$7
3030
3031  014362  012700  177763    PLT1:   MOV     #-15,R0         ;TYPE OUT THE X-AXIS MARKERS
3032  014366                    1$:
3033  014366  104401  014374            TYPE    ,65$            ;;TYPE ASCIZ STRING
3034  014372  000403                    BR      64$             ;;GET OVER THE ASCIZ
3035                            ;;65$:  .ASCIZ  /I----/
3036  014402                    64$:
3037  014402  005200                    INC     R0              ;I----I----I----
3038  014404  001370                    BNE     1$
3039
3040                                    ;IF SW 4 IS SET THEN TYPE THE COMPLETE GRAPH, IF NOT TYPE THE SMALL GRAPH,
3041
3042  014406  032777  000020  164524            BIT     #SW4,@SWR       ;TYPE COMPLETE GRAPH?
3043  014414  001054                    BNE     CMPGRP          ;YES BRANCH
3044                                            ;IF NOT, TYPE SMALL GRAPH
3045
3046
```

```
                                SMGRP:  CLR     R0
014420  032777  000040  164512  1$:     BIT     #SW5,@SWR       ;SKIP REST OF GRAPH?
014426  001445                          BEQ     5$              ;YES
014430  104401                          TYPE                    ;IN THIS GRAPH SEEK TIMES ARE
014432  001213                          $CRLF           ;PLOTTED ONLY FOR SELECTED
                                                        ;CYLINDERS (NOT ALL) SHOWN BELOW:
                                                        ;0,1,2,3,4,  6,8,10,12,14,16,18,20,
                                                        ;25,30,35,...,190,195,200,  203
014434  010046                          MOV     R0,-(SP)                ;TYPE THE MARKERS
014436  104405                          TYPDS
014440  104401  014446                  TYPE    ,65$            ;;TYPE ASCIZ STRING
014444  000401                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /-/
014450                          64$:
014450  010001                          MOV     R0,R1           ;FORM THE ADRES OF 'SEEK TIME'
014452  006301                          ASL     R1
014454  016103  026742                  MOV     BUFR10(R1),R3   ;GET THE SEEK TIME
014460  004737  014662                  JSR     PC,PLTPT                ;GO PLOT IT
014464  022700  000004                  CMP     #4,R0           ;PLOTTED UPTO CYL 4?
014470  003402                          BLE     2$              ;YES
014472  005200                          INC     R0
014474  000751                          BR      1$
014476  022700  000024          2$:     CMP     #24,R0  ;PLOTTED UPTO CYL 20?
014502  003403                          BLE     3$
014504  062700  000002                  ADD     #2,R0
014510  000743                          BR      1$
014512  022700  000310          3$:     CMP     #310,R0         ;PLOTTED UPTO CYL 200?
014516  003403                          BLE     4$
014520  062700  000005                  ADD     #5,R0
014524  000735                          BR      1$
014526  022700  000312          4$:     CMP     #312,R0         ;PLOTTED ALL CYLS?
014532  001403                          BEQ     5$
014534  062700  000002                  ADD     #2,R0
014540  000727                          BR      1$
014542  000137  015150          5$:     JMP     TST12


                                ;IF SW 4 IS SET THE COMPLETE GRAPH IS PRINTED OUT. IT GIVES TIMES FOR
                                ;SEEKS FROM CYLINDER 0 TO ALL OTHER CYLINDERS (0,1,2,3...202).

014546  005000                  CMPGRP: CLR     R0              ;INITLZE COUNT
014550  012701  177773                  MOV     #-5,R1          ;INITLZE COUNT FOR Y-AXIS MARKER
014554  012702  026742                  MOV     #BUFR10,R2      ;INITLZE PTR TO SEEK TIMES
014560  104401                          TYPE
014562  001213                          $CRLF
014564  000412                          BR      3$

014566  032777  000040  164344  2$:     BIT     #SW5,@SWR               ;SKIP REST OF GRAPH?
014574  001002                          BNE     .+6
014576  000137  015150                  JMP     TST12
014602  005201                          INC     R1              ;TYPE OUT Y-AXIS MARKER 'CYL #'
014604  001005                          BNE     4$              ;IF REQUIRED
014606  012701  177773                  MOV     #-5,R1
014612  010046                  3$:     MOV     R0,-(SP)        ;TYPE 'CYL #' ON Y-AXIS
014614  104405                          TYPDS                   ;(IN DECIMAL)
```



```
014616  000402                          BR      5$
014620  104401                  4$:     TYPE
014622  002104                          BLNKS6
014624                          5$:
014624  104401  014632                  TYPE    ,65$            ;;TYPE ASCIZ STRING
014630  000401                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /-/
014634                          64$:

014634  012203                          MOV     (R2)+,R3                ;GET SEEK TIME
014636  004737  014662                  JSR     PC,PLTPT        ;GO PLOT THE POINT


014642  104401                          TYPE
014644  001213                          $CRLF
014646  005200                          INC     R0              ;ALL DONE?
014650  022700  000312                  CMP     #312,R0
014654  001344                          BNE     2$              ;IF NOT, GO BAK
014656  000137  015150          6$:     JMP     TST12

                                ;PLTPT
                                ;THE ROUTINE IS ENTERED WITH R3 CONTAINING HORIZONTAL AXIS
                                ;COORDINATE- SEEK TIME
                                ;PLOT THE ACTUAL TIME ON THE GRAPH.  IN KEEPING WITH NORMAL
                                ;CONVENTION A NUMBER LESS THAN HALF THE CELL WIDTH IS
                                ;CONSIDERED AS FALLING UNDER THE PREVIOUS CELL, A NUMBER
                                ;GREATER THAN OR EQUAL TO HALF THE CELL WIDTH FALLS UNDER THE NEXT CELL
                                ;EX: IF SEEK TIME IS 11.5 MS, IT'S BETW'N 10 & 12, BUT > 11
                                ;HENCE IT WILL BE PLOTTED AS 12 IF SEEK TIME IS 10.8 MS,
                                ;IT'S BETW'N 10 & 12, BUT < 11 HENCE IT WILL BE PLOTTED
                                ;AS 10.0 MS

014662  162703  000310  PLTPT:  SUB     #310,R3         ;FIND OUT HOW MANY BLANKS TO
014666  002403                          BLT     7$              ;INSERT TO PLOT THE POINT
                                                        ;NOTE THE FIRST CELL = 0 MS
014670  104401                          TYPE
014672  002111                          BLNKS1
014674  000772                          BR      PLTPT
014676  062703  000144          7$:     ADD     #144,R3
014702  002402                          BLT     8$
014704  104401                          TYPE
014706  002111                          BLNKS1

014710                          8$:
014710  104401  014716                  TYPE    ,65$            ;;TYPE ASCIZ STRING
014714  000401                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /X/
014720                          64$:
014720  000207                          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.
```

```
3159                                    ;RETURN:        R3 CONTAINS THE SEEK TIME IN MILI SECS, SCALE FACTOR = 0,01
3160
3161
3162                                                                    ;R3 WILL COUNT REVOLUTIONS OF
3163  014722  010246                    TIMSEK: MOV     R2,-(SP)        ;DISK (FROM INDEX MARK TO INDEX MARK)
3164  014724  005003                            CLR     R3              ;40 MILI SECS FOR EACH REV
3165  014726  013701  001452                    MOV     RKDS,R1
3166  014732  011102                    1$:     MOV     @R1,R2
3167  014734  032702  000400                    BIT     #400,R2         ;WAIT FOR SOK
3168  014740  001774                            BEQ     1$
3169
3170  014742  032702  000010                    BIT     #BIT3,R2                ;WAIT FOR SECTOR 10 SO THAT
3171  014746  001771                            BEQ     1$              ;U CAN START WAITING FOR
3172                                                                    ;INDEX, SEC 0
3173
3174  014750  011102                    2$:     MOV     @R1,R2
3175  014752  032702  000400                    BIT     #400,R2         ;WAIT FOR SEC OK
3176  014756  001774                            BEQ     2$
3177  014760  021102                            CMP     @R1,R2
3178  014762  001372                            BNE     2$
3179  014764  032702  000017                    BIT     #17,R2                  ;WAIT FOR SEC 0, INDEX MARK
3180  014770  001367                            BNE     2$              ;AS SOON AS IT IS SEC 0, ISSUE
3181                                                                    ;A SEEK & START TIMING
3182                                                                    ;
3183  014772  012777  000011  164456            MOV     #11,@RKCS       ;ISSUE A SEEK, START TIMING
3184                                                                    ;THE SEC COUNTER
3185  015000  104421                            CON,RDY                 ;WAIT FOR CNTRL RDY
3186
3187  015002  011102                    3$:     MOV     @R1,R2          ;GET RKDS
3188  015004  032702  000400                    BIT     #400,R2         ;WAIT FOR SOK
3189  015010  001774                            BEQ     3$
3190  015012  020211                            CMP     R2,@R1          ;INFO CORRECT?
3191  015014  001372                            BNE     3$              ;NO
3192  015016  032702  000100                    BIT     #100,R2         ;R/W/S RDY SET?
3193  015022  001025                            BNE     SKDON           ;IF YES, BRANCH
3194  015024  032702  000017                    BIT     #17,R2          ;WAIT FOR SEC CNTR TO MOVE
3195  015030  001764                            BEQ     3$              ;FROM 0 TO 1
3196
3197  015032  011102                    4$:     MOV     @R1,R2
3198  015034  032702  000400                    BIT     #400,R2         ;WAIT FOR SOK
3199  015040  001774                            BEQ     4$
3200  015042  020211                            CMP     R2,@R1
3201  015044  001372                            BNE     4$
3202  015046  032702  000100                    BIT     #100,R2         ;R/W/S RDY SET, SEEK DONE?
3203  015052  001005                            BNE     5$              ;YES, BRANCH
3204  015054  032702  000017                    BIT     #17,R2                  ;IF NOT KEEP TRACK OF SEC
3205  015060  001364                            BNE     4$              ;COUNTER, INCREMENT R3 AT
3206                                                                    ;EVERY INDEX MARK,EVERY
3207  015062  005203                            INC     R3              ;40 MILI SECS
3208  015064  000746                            BR      3$              ;GO BAK, KEEP TIME
3209
3210  015066  032702  000017            5$:     BIT     #17,R2          ;CHECK, IS IT INDEX MARK -SEC 0
3211  015072  001001                            BNE     SKDON           ;IF NOT, SKIP
3212  015074  005203                            INC     R3              ;IF YES, INCREMENT COUNT
3213
3214                                                                            ;SEEK DONE, SAVE RKDS-SEC COUNTER.
```



```
3215  015076                    SKDON:
3216  015076  012746  000014                    MOV     #14,-(SP)               ;;PUT THE MULTIPLER ON THE STACK
3217  015102  010346                            MOV     R3,-(SP)                ;;PUT THE MULTIPLICAND ON THE STACK
3218  015104  004737  020414                    JSR     PC,@#$MULT      ;;CALL THE MULTPLY ROUTINE
3219  015110  012616                            MOV     (SP)+,(SP)      ;;DISREGARD THE MSB'S
3220  015112  012603                            MOV     (SP)+,R3                ;;GET THE LSB'S OF THE PRODUCT
3221  015114  042702  177760                    BIC     #177760,R2      ;SEEK, TOTAL TIME=(IN DECIMAL)
3222  015120  060203                            ADD     R2,R3           ;[(R3)X12+SEC COUNTER]X330X0,01
3223                                                                    ;NOTE THERE IS A SCALE FACTOR
3224  015122  012746  000512                    MOV     #512,-(SP)              ;;PUT THE MULTIPLER ON THE STACK
3225  015126  010346                            MOV     R3,-(SP)                ;;PUT THE MULTIPLICAND ON THE STACK
3226  015130  004737  020414                    JSR     PC,@#$MULT      ;;CALL THE MULTPLY ROUTINE
3227  015134  012616                            MOV     (SP)+,(SP)      ;;DISREGARD THE MSB'S
3228  015136  012603                            MOV     (SP)+,R3                ;;GET THE LSB'S OF THE PRODUCT
3229  015140  062703  000245                    ADD     #245,R3         ;ASSUMPTION THAT EACH SECTOR
3230                                                                    ;TAKES 3,3 MILI SECS, IF THE
3231                                                                    ;DISK SPEED IS VERY MUCH DIFRNT
3232                                                                    ;FROM THE SPEC SPEED OF
3233                                                                    ;1500 RPM (40 MS/REV), THEN
3234                                                                    ;SEC COUNTER WOULD NOT BE AN
3235                                                                    ;ACCURATE TIME CLOCK.
3236  015144  012602                            MOV     (SP)+,R2        ;POP R2 BAK
3237  015146  000207                            RTS     PC              ;RETURN
3238
3239
3240                                    ;;*****************************************************************
3241                                    ;*TEST 12        END OF PROGRAM
3242                                            ;*THIS IS NOT A TEST BUT IS JUST A LINKAGE
3243                                            ;*PROVIDED TO TEST ALL THE DRIVES.
3244                                    ;;*****************************************************************
3245  015150  000004                    TST12:  SCOPE
3246  015152  105237  001223                    INCB    DRVDON
3247  015156  123737  001223  001224    BTEOP:  CMPB    DRVDON,DRIVS
3248  015164  001402                            BEQ     .+6
3249  015166  000137  003704                    JMP     NXTDRV
3250
3251                                    .SBTTL  END OF PASS ROUTINE
3252
3253                                    ;;*****************************************************************
3254                                    ;*INCREMENT THE PASS NUMBER ($PASS)
3255                                    ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
3256                                    ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
3257                                    ;*IF THERES A MONITOR GO TO IT
3258                                    ;*IF THERE ISN'T JUMP TO ST3
3259
3260  015172                    $EOP:
3261  015172  000004                            SCOPE
3262  015174  005037  001102                    CLR     $TSTNM          ;;ZERO THE TEST NUMBER
3263  015200  005237  001100                    INC     $PASS           ;;INCREMENT THE PASS NUMBER
3264  015204  042737  100000  001100            BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
3265  015212  005327                            DEC     (PC)+           ;;LOOP?
3266  015214  000001                    $EOPCT: .WORD   1
3267  015216  003022                            BGT     $DOAGN          ;;YES
3268  015220  012737                            MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
3269  015222  000001                    $ENDCT: .WORD   1
3270  015224  015214                            $EOPCT
```

```
 015226  104401  015273                TYPE    ,$ENDMG         ;;TYPE "END PASS #"
 015232  013746  001100                MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 015236  104405                        TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 015240  104401  015270                TYPE    ,$ENULL         ;;TYPE A NULL CHARACTER
 015244  013700  000042        $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 015250  001405                        BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 015252  000005                        RESET                   ;;CLEAR THE WORLD
 015254  004710                $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 015256  000240                        NOP                     ;;SAVE ROOM
 015260  000240                        NOP                     ;;FOR
 015262  000240                        NOP                     ;;ACT11
 015264                        $DOAGN:
 015264  000137                        JMP     @(PC)+          ;;RETURN
 015266  003666                $RTNAD: .WORD   ST3
 015270     377     377     000 $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 015273     015  042412  042116 $ENDMG: .ASCIZ  <15><12>/END PASS #/
 015300  050040  051501  020123
 015306  000043
```



```
                                ;COMMON SUBROUTINES AND HANDLERS



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

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

 015310                         ESR15:
 015310  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 015312  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 015314  012701  001266                 MOV     #BUFR,R1        ;SEC #'S STORED HERE PREVIOUSLY
 015320  012702  001320                 MOV     #BUFR1,R2       ;BAD HDRS STORED HERE PRVSLY
 015324  012146                 1$:     MOV     (R1)+,-(SP)
 015326  104403                         TYPOS                   ;GO TYPE OUT BAD SEC # (OCTAL)
 015330     002                         .BYTE   2               ;ONLY 2 DIGITS
 015331     000                         .BYTE   0               ;SUPRES LDG 0'S
 015332  104401                         TYPE                    ;TYPE 3 BLNKS
 015334  002107                         BLNKS3
 015336  012246                         MOV     (R2)+,-(SP)     ;GO TYPE OUT BAD HEADER
 015340  104402                         TYPOC
 015342  104401                         TYPE
 015344  002106                         BLNKS4
 015346  104401                         TYPE
 015350  001213                         $CRLF
 015352  022711  177777                 CMP     #177777,(R1)    ;ALL BAD SEC #'S TYPD OUT?
 015356  001362                         BNE     1$              ;IF NOT GO BAK

 015360  104401                         TYPE
 015362  001716                         MSG6
 015364  010546                         MOV     R5,-(SP)        ;TYPE OUT EXPCTD HEADER FOR
 015366  042716  160037                 BIC     #160037,(SP)
 015372  104402                         TYPOC                   ;THAT CYLINDER

 015374  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
 015376  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
 015400  000207                         RTS     PC


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

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

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

015402  004737  015310          ESR13:  JSR     PC,ESR15
015406  104401  015414                  TYPE    ,65$            ;;TYPE ASCIZ STRING
015412  000404                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /  SUR=/
015424                          64$:
015424  005046                          CLR     -(SP)
015426  032705  000020                  BIT     #20,R5          ;SUR 0 OR 11?
015432  001401                          BEQ     1$
015434  005216                          INC     (SP)
015436  104402                  1$:     TYPOC

015440  104401  002053                  TYPE    ,MSG13
015444  013746  001254                  MOV     RETRY2,-(SP)
015450  005216                          INC     (SP)
015452  104402                          TYPOC
015454  000207                          RTS     PC


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

015456  004737  015402  016244  ESR20:  JSR     PC,ESR13        ;GO TYPE OUT SEC #'S, BAD HDRS
015462  004737  016244                  JSR     PC,ERR2         ;GET CYL #'S BETWN WHICH SEEK
                                                                ;WAS TRIED
015466  104401  015474                  TYPE    ,65$            ;;TYPE ASCIZ STRING
015472  000404                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /  CYLA=/
015504                          64$:
015504  013746  001162                  MOV     $REG0,-(SP)     ;GO TYPE CYL # FROM WHERE
015510  104403                          TYPOS                   ;SEEK BEGAN
015512     003                          .BYTE   3               ;TYPE 3 DIGITS
015513     000                          .BYTE   0               ;SUPRES LDG 0'S
015514  104401  015522                  TYPE    ,67$            ;;TYPE ASCIZ STRING
015520  000404                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /  CYLB=/
015532                          66$:
015532  013746  001164                  MOV     $REG1,-(SP)     ;TYPE CYL # TO WHICH SEEK
015536  104403                          TYPOS                   ;WAS DONE
015540     003                          .BYTE   3               ;TYPE 3 DIGITS
015541     000                          .BYTE   0               ;SUPRES LDG 0'S
015542  000207                          RTS     PC              ;RETURN


                                        .SBTTL  ESR25
```



```
015544  010205                  ESR25:  MOV     R2,R5           ;SAVE ADRES OF TERMINATOR

015546  012702  001266                  MOV     #BUFR,R2        ;INITLZE PTR TO TABLE STORING
                                                                ;ADRES OF BAD DATA
015552  012703  001320                  MOV     #BUFR1,R3       ;INITLZE PTR TO 'EXPCTD' DATA
015556  012704  001352                  MOV     #BUFR2,R4       ;INITLZE PTR TO 'RECVD' DATA

015562  032777  020000  163350  1$:     BIT     #SW13,@$WR      ;INHIBIT TYPE OUT?
015570  001076                          BNE     4$              ;YES, EXIT
015572  104401                          TYPE                    ;TYPE CR,LF
015574  001213                          $CRLF

015576  163712  001404                  SUB     PBUF0,(R2)      ;GET WORD # IN BUFR (0,1,2,..)
015602  006212                          ASR     (R2)
015604  011246                          MOV     (R2),-(SP)      ;WHICH WAS BAD.  NOTE YOU
                                                                ;CAN HAVE THE ACTUAL MEMORY
                                                                ;ADRES BY ADDING 'IOBUF0'
                                                                ;TO THIS
015606  104403                          TYPOS                   ;GO TYPE WORD # THAT WAS BAD
015610     004                          .BYTE   4
015611     000                          .BYTE   0
015612  104401                          TYPE
015614  002107                          BLNKS3                  ;2 BLANKS

015616  012346                          MOV     (R3)+,-(SP)     ;GET EXPCTD DATA
015620  104402                          TYPOC                   ;GO TYPE IT
015622  104401                          TYPE
015624  002110                          BLNKS2
015626  012446                          MOV     (R4)+,-(SP)     ;GET RECVD DATA (BAD)
015630  104402                          TYPOC                   ;GO TYPE IT
015632  104401                          TYPE
015634  002110                          BLNKS2

015636  012700  000400                  MOV     #400,R0         ;GET THE DISK ADRES FROM
015642  021200                  2$:     CMP     (R2),R0         ;WHICH THIS (BAD) DATA WAS
015644  002405                          BLT     3$              ;READ
015646  062700  000400                  ADD     #400,R0
015652  022700  002400                  CMP     #2400,R0
015656  001371                          BNE     2$

015660  000300                  3$:     SWAB    R0
015662  005300                          DEC     R0
015664  063700  001450                  ADD     ADRES,R0        ;R0 CONTAINS THE DISK
                                                                ;ADRES FROM WHICH THE (BAD)
015670  010037  001170                  MOV     R0,$REG3        ;DATA WAS READ

015674  004737  016144                  JSR     PC,BRKDA        ;GO BREAK ABOVE DISK ADRES
                                                                ;INTO CYL#, SUR#, SEC#

015700  013746  001174                  MOV     $REG5,-(SP)     ;GET THE CYL#
015704  104403                          TYPOS                   ;TYPE IT
015706     003                          .BYTE   3               ;ONLY 3 DIGITS
015707     000                          .BYTE   0               ;NO LEADING 0'S
015710  104401                          TYPE
015712  002107                          BLNKS3
```

```
3458  015714  013746  001176                  MOV     $REG6,-(SP)     ;GET SUR #
3459  015720  104403                          TYPOS                   ;TYPE
3460  015722     001                          .BYTE   1               ;1 DIGIT ONLY
3461  015723     000                          .BYTE   0
3462
3463  015724  104401                          TYPE
3464  015726  002106                          BLNKS4
3465
3466  015730  013746  001200                  MOV     $REG7,-(SP)     ;GET SEC#
3467  015734  104403                          TYPOS                   ;TYPE
3468  015736     002                          .BYTE   2               ;2 DIGITS
3469  015737     000                          .BYTE   0
3470
3471  015740  005722                          TST     (R2)+           ;INCREMNT PTR
3472  015742  020205                          CMP     R2,R5           ;TYPED OUT ALL BAD DATA
3473                                                                  ;INFO?
3474  015744  001306                          BNE     1$              ;IF NOT LUP BAK
3475  015746  104401                          TYPE
3476  015750  002053                          MSG13                   ;'  TRY #:'
3477  015752  013746  001254                  MOV     RETRY2,-(SP)    ;GET RETRY COUNT
3478  015756  062716  000003                  ADD     #3,(SP)         ;FORM THE RETRY NO.
3479  015762  104403                          TYPOS                   ;TYPE IT OUT
3480  015764     001                          .BYTE   1
3481  015765     000                          .BYTE   0
3482
3483  015766  000207                  4$:     RTS     PC              ;IF YES, RETURN
3484                                  ;MESSAGE HANDLER
3485                                  ;THE MESSAGE HANDLER IS USED FOR TYPING OUT MESSAGES & DATA
3486                                  ;RELATED TO THE MESSAGE. IF SW13 IS SET, THE TYPEOUT IS
3487                                  ;INHIBITED. THE CALL IS:
3488                                  ;       MESAGE  ,XX
3489                                  ;XXX IS THE MESSAGE NUMBER & PROVIDES AN INDEX TO THE
3490                                  ;'ERROR ITEMS TABLE' WHERE THAT MESAGE ITEM
3491                                  ;IS LOCATED.
3492                                  ;THE MESAGE ITEM CONTAINS:
3493                                  ;       MS:     POINTER TO THE ASCII MESAGE
3494                                  ;       DH:     POINTER TO THE DATA HEADER
3495                                  ;       DT:     POINTER TO THE DATA
3496                                  ;       0       TERMINATOR
3497                                  ;IF 'DT' IS 0 THE DATA IS TO BE PRINTED USING THE SUBROUTINE
3498                                  ;INDICATED IN PLACE OF THE TERMINATOR
3499
3500  015770  032777  020000  163142  MSGE:   BIT     #SW13,@SWR      ;INHIBIT TYPEOUT?
3501  015776  001012                          BNE     1$              ;IF YES, EXIT
3502  016000  011637  001116                  MOV     (SP),$ERRPC     ;GET ADRES OF 'MESAGE' CALL
3503  016004  162737  000002  001116          SUB     #2,$ERRPC       ;STORE IT
3504  016012  117637  000000  001114          MOVB    @(SP),$ITEMB    ;GET MESAGE # (INDEX TO ITEM TABLE)
3505  016020  004737  017362                  JSR     PC,@#ERRTYP     ;GO TO 'ERRTYP' & TYPE OUT
3506                                                                  ;INFO
3507  016024  062716  000002          1$:     ADD     #2,(SP)         ;ADJUST RETURN ADDRES
3508  016030  000002                          RTI                     ;EXIT
3509
3510                                  ;THIS ROUTINE IS USED FOR TYPING OUT ASCII MESSAGES. BEFORE
3511                                  ;THE MESSAGE IS TYPED SW13 IS CHECKED & IF SET THE
3512                                  ;TYPEOUT IS INHIBITED & AN EXIT IS MADE.
3513                                  ;THE CALL FOR THIS ROUTINE IS "TYPMSG", AN ENCODED
```



```
3514                                  ;TRAP INSTRUCTION.
3515                                  ;THE POINTER TO THE ASCII MESSAGE TO BE TYPED IS LOCATED IN  THE
3516                                  ;WORD FOLLOWING THE "TYPMSG" CALL.
3517
3518  016032  032777  020000  163100  TY.MSG: BIT     #SW13,@SWR      ;INHIBIT TYPEOUT?
3519  016040  001005                          BNE     2$              ;YES, EXIT
3520  016042  017637  000000  016052          MOV     @(SP),1$        ;GET POINTER TO ASCII MESSAGE
3521  016050  104401                          TYPE                    ;GO TYPE ASCII STRING
3522  016052  000000                  1$:     0
3523  016054  062716  000002          2$:     ADD     #2,(SP)         ;ADJUST RETURN ADRES, SKIP OVER
3524                                                                  ;POINTER ON RETURN
3525  016060  000002                          RTI                     ;EXIT
3526
3527
3528
3529
3530                                  ;GT5RG
3531                                  ;THIS ROUTINE EXTRACTS THE CYLINDER # FROM RKDA AND STORES IT
3532                                  ;IN $REG4. THEN TRANSFERS RKCS, ER, DS, DA TO $REG0, $REG1, $REG2, $REG3
3533  016062  017746  163376          GT5RG:  MOV     @RKDA,-(SP)     ;PUSH RKDA ONTO STACK
3534  016066  042716  160037                  BIC     #160037,(SP)    ;MASK OUT NON-CYLINDER BITS
3535  016072  006316                          ASL     (SP)            ;SHIFT 8 BITS OF CYL ADRES TO LO BYTE
3536  016074  006316                          ASL     (SP)
3537  016076  006316                          ASL     (SP)
3538  016100  000316                          SWAB    (SP)
3539  016102  112637  001172                  MOVB    (SP)+,$REG4     ;UP STACK
3540
3541
3542
3543                                  ;GT4RG
3544                                  ;THIS ROUTINE TRANSFERS THE CONTENTS OF RKCS, RKER, RKDS
3545                                  ;RKDA TO $REG0, $REG1, $REG2, $REG3 RESPECTIVELY. $REG'S
3546                                  ;ARE USED FOR TYPING OUT THERE CONTENTS AT THE TIME OF ERROR
3547
3548  016106  017737  163344  001162  GT4RG:  MOV     @RKCS,$REG0     ;GET RKCS
3549  016114  017737  163334  001164          MOV     @RKER,$REG1     ;    RKER
3550  016122  017737  163324  001166          MOV     @RKDS,$REG2     ;    RKDS
3551  016130  017737  163330  001170          MOV     @RKDA,$REG3     ;    RKDA
3552  016136  000207                          RTS     PC              ;EXIT FROM THIS ROUTINE
3553
3554                                  ;GETINF
3555                                  ;THIS ROUTINE SAVES THE CONTENTS OF RKCS IN $REG0
3556                                  ;RKER IN $REG1, RKDS IN $REG2.  THEN IT BREAKS RKDA
3557                                  ;INTO DRIVE NO, CYLINDER #, SURFACE AND SECTOR #.
3558                                  ;AND SAVES THEM IN $REG4, $REG5, $REG6, $REG7.
3559
3560  016140  004737  016106          GETINF: JSR     PC,GT4RG
3561  016144  010046                  BRKDA:  MOV     R0,-(SP)
3562  016146  010146                          MOV     R1,-(SP)
3563  016150  010246                          MOV     R2,-(SP)
3564  016152  012700  001202                  MOV     #$REG7+2,R0
3565  016156  013701  001170                  MOV     $REG3,R1
3566  016162  010102                          MOV     R1,R2
3567  016164  042702  177760                  BIC     #177760,R2
3568  016170  010240                          MOV     R2,-(R0)
3569  016172  006201                          ASR     R1
```

```
                                        ASR     R1
                                        ASR     R1
                                        ASR     R1
                                        MOV     R1,R2
                                        BIC     #177776,R2
                                        MOV     R2,-(R0)
                                        ASR     R1
                                        MOV     R1,R2
                                        BIC     #177400,R2
                                        MOV     R2,-(R0)
                                        SWAB    R1
                                        BIC     #177770,R1
                                        MOV     R1,-(R0)
                                        MOV     (SP)+,R2
                                        MOV     (SP)+,R1
                                        MOV     (SP)+,R0
                                        RTS     PC
                                        .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 $REG0 CONTAINS CYL #
                                ;FROM WHICH SEEK WAS INITIATED, $REG1 CONTAINS CYL # TO WHICH SEEK WAS DONE
016244  013737  001260  001162  ERR2:  MOV     INADR,$REG0     ;GET CYL ADRES
016252  004737  016360          JSR     PC,GCYL         ;GO GET CYL# FROM IT
016256  013737  001162  001164  MOV     $REG0,$REG1     ;SAVE
016264  013737  001262  001162  MOV     OUTADR,$REG0    ;GET CYL ADRES
016272  004737  016360          JSR     PC,GCYL         ;GO GET CYL # FROM IT
016276  005704                  TST     R4              ;GOING WHICH WAY?
016300  001407                  BEQ     1$              ;'OUTADR' TO 'INADR', BRANCH
016302  013746  001162          MOV     $REG0,-(SP)     ;EXCHANG CYL" TO GET
016306  013737  001164  001162  MOV     $REG1,$REG0     ;CORRECT 'TO' & 'FROM' CYLS
016314  012637  001164          MOV     (SP)+,$REG1
016320  000207                  1$:     RTS     PC              ;RETURN
                                        .SBTTL  ERR1
                                ;ERR1
                                ;THIS SUBROUTINE FINDS OUT THE CYLINDER NOS. BETWEEN WHICH THE SEEK
                                ;IS DONE.THE CYLINDER # WHERE THE  HEADS WHERE PRIOR TO MOVING, IS
                                ;DEPOSITED IN $REG0, THE CYLINDER # WHERE THE HEADS SHOULD BE AFTER
                                ;MOVEMENT, IS DEPOSITED IN $REG1.  R4 INDICATES WHICH DIRECTION THE
                                ;HEADS WERE MOVING, IN OR OUT. R5 CONTAINS THE
                                ;DISK ADDRESS (IN OR OUT AS THE CASE MAY BE).
016322  010537  001162          ERR1:  MOV     R5,$REG0
016326  004737  016360          JSR     PC,GCYL ;GO GET CYL #
016332  005704                  TST     R4              ;WAS GOING IN OR OUT?
016334  001006                  BNE     1$              ;OUT
016336  013737  001162  001164  MOV     $REG0,$REG1
016344  005037  001162          CLR     $REG0
```



```
016350  000207                  RTS     PC
016352  005037  001164          1$:     CLR     $REG1
016356  000207                  RTS     PC
                                        .SBTTL  GCYL
                                ;GCYL
                                ;THIS ROUTINE EXTRACTS THE CYLINDER NO. FROM THE DISK ADDRESS
                                ;CONTAINED IN '$REG0', AND THEN STORES IT BACK IN '$REG0'
016360  010046                  GCYL:   MOV     R0,-(SP)        ;PUSH R0 ONTO STACK
016362  013700  001162          MOV     $REG0,R0
016366  042700  160037          BIC     #160037,R0      ;MASK OUT DRV # BITS &
                                                        ;SUR, SEC BITS IF PRESENT
016372  006200                  ASR     R0
016374  006200                  ASR     R0              ;SHIFT CYL BITS RIGHT
016376  006200                  ASR     R0              ;BY 5
016400  006200                  ASR     R0
016402  006200                  ASR     R0
016404  010037  001162          MOV     R0,$REG0        ;STORE CYL # IN $REG0
016410  012600                  MOV     (SP)+,R0        ;POP R0 FROM STACK
016412  000207                  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.
016414  005037  001174          DR.RST: CLR     $REG5           ;INITIALIZE THE COUNT
016420  013777  001230  163036  MOV     DRIVAD,@RKDA
016426  012777  000015  163022  MOV     #15,@RKCS       ;DRIVE RESET, GO
016434  104421                  CON.RDY
016436  000402                  BR      RES.DO+4
016440  005037  001174          RES.DO: CLR     $REG5
016444  032777  000100  163000  1$:     BIT     #100,@RKDS              ;DID R/W/S RDY SET?
016452  001024                  BNE     2$
016454  012746  177778          MOV     #-10,-(SP)      ;PUSH COUNT ON SP
016460  005216                  INC     (SP)            ;COUNT IT DOW
016462  001376                  BNE     .-2
016464  005726                  TST     (SP)+           ;POP UP $P
016466  005237  001174          INC     $REG5                   ;IF NOT WAIT
016472  001364                  BNE     1$                      ;WAITED LONG?
016474  032777  020000  162436  BIT     #SW13,@SWR
016502  001010                  BNE     2$
016504  104420                  TYPMSG
016506  002027                  MSG12
016510  104420  001733          TYPMSG  ,MSG7
016514  011646                  MOV     (SP),-(SP)
016516  162716  000002          SUB     #2,(SP)
016522  104402                  TYPOC
016524  000002                  2$:     RTI
```

```
                                        .SBTTL  CON.RESET - CONTROL RESET ROUTINE
                                        .SBTTL  CON.RDY - WAIT FOR CONTROL READY
                                ;CON.RESET
                                ;CON.RDY
                                ;THIS ROUTINE IS CALLED BY USING 'CNT.RESET' WHICH IS ACTUALLY
                                ;'TRAP' INSTRUCTION WITH THE LOWER BYTE ENCODED TO PROVIDE
                                ;AN INDEX TO THE CONTROL-RESET ROUTINE BELOW.
                                ;THE ROUTINE ISSUES A CONTROL RESET AND WAITS FOR
                                ;THE 'CNTRL RDY' FLAG TO SET.  WHEN THE FLAG SETS
                                ;AN EXIT IS MADE OUT OF THE ROUTINE.  IF 'CNTRL-RDY'
                                ;DOES NOT SET WITHIN A CERTAIN TIME AN ERROR MESSAGE
                                ;       CNT RDY DIDN'T SET
                                ;       PC=XXXXXX RKCS=XXXXXX
                                ;IS GIVEN.
                                ;THIS ROUTINE IS CALLED THROUGH THE 'TRAP' INSTRUCTION
                                ;USING THE LOWER BYTE AS AN INDEX TO THIS ROUTINE.
                                ;THE TRAP DECODER LOCATED AT '$TRAP'.


                                ;CN.RDY
                                ;THE CN.RDY ROUTINE IS CALLED BY USING CNT.RDY WHICH IS A TRAP
                                ;INSTRUCTION WITH ITS LOWER BYTE ENCODED.
                                ;THIS ROUTINE WAITS FOR THE CONTROL READY BIT TO SET AND WHEN IT
                                ;SETS EXITS OUT. IF WITHIN A CERTAIN TIME CNTRL RDY DOES
                                ;NOT SET AN ERROR IS REPORTED. WAITING TIME IS 883 MS FOR 11/20
                                ;175 MS FOR 11/45 WITH BIPOLAR MEMORY.
016526  012777  000001  162722  CN.RST: MOV     #1,@RKCS        ;ISSUE A CONTROL RESET
016534  012737  177500  001170          MOV     #-300,$REG3     ;SET UP COUNT
016542  000402                  BR      CN.RDY+4                ;SKIP OVER CN.RDY
016544  005037  001170          CN.RDY: CLR     $REG3
016550  105777  162702          1$:     TSTB    @RKCS           ;DID CNTRL-RDY SET?
016554  100431                  BMI     2$              ;YES, EXIT
016556  005237  001170                  INC     $REG3           ;WAITED LONG?
016562  001372                  BNE     1$              ;IF NOT, GO BAK & WAIT
016564  104420                  TYPMSG
016566  001742                  MSG10
016570  104401  016576                  TYPE    ,65$            ;;TYPE ASCIZ STRING
016574  000403                  BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/PC=/
016604                          64$:
016604  011646                  MOV     (SP),-(SP)
016606  162716  000002                  SUB     #2,(SP)
016612  104402                  TYPOC                   ;GO TYPE PC IN THE MAIN PROGRAM,
                                                        ; WHERE ERROR OCCURRED
016614  104401  016622                  TYPE    ,67$            ;;TYPE ASCIZ STRING
016620  000404                  BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  / RKCS=/
016632                          66$:
016632  017746  162620                  MOV     @RKCS,-(SP)     ;GET RKCS
016636  104402                  TYPOC                   ;GO TYPE IT

016640  000002                  2$:     RTI                     ;RETURN FROM THIS
                                                        ;ROUTINE TO THE MAIN
```



```
                                                        ;PROGRAM

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

016642  005037  001264          TSTRWS: CLR     TIMER
016646  032777  000100  162576  1$:     BIT     #100,@RKDS
016654  001017                  BNE     2$
016656  005237  001264                  INC     TIMER
016662  001371                  BNE     1$
016664  032777  020000  162246          BIT     #BIT13,@SWR
016672  001010                  BNE     2$
016674  104420  002027                  TYPMSG  ,MSG12
016700  104420  001733                  TYPMSG  ,MSG7
016704  011646                  MOV     (SP),-(SP)
016706  162716  000002                  SUB     #2,(SP)
016712  104402                  TYPOC
016714  000002                  2$:     RTI

                                        .SBTTL  TEST ABORT ROUTINE
                                ;ABRT

016716  104401  001616          ABRT:   TYPE    ,MSG3
016722  113746  001102                  MOVB    $TSTNM,-(SP)
016726  104402                  TYPOC
016730  000207                  RTS     PC


                                ;COMMON SUBROUTINES & HANDLERS

                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;!*************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

016732                          $SCOPE:
016732  104407                  CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
016734  032777  000400  162176          BIT     #SW8,@SWR               ;WAS SW8 USED TO SELECT
016742  001053                  BNE     $OVER           ;A TEST? IF YES, SKIP OVER
                                                        ;THE REST, U ARE LOOPING ON
016744  032777  040000  162166  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
016752  001047                  BNE     $OVER           ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
016754  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
```

```
3794                                                    ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
3795  016756  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
3796  016762  012737  017002  000004  MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
3797  016770  005737  177060          TST     @#177060        ;;TIME OUT ON XOR?
3798  016774  012637  000004          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
3799  017000  000421                  BR      $SVLAD          ;;GO TO THE NEXT TEST
3800  017002  022626          5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
3801  017004  012637  000004          MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
3802  017010  000407                  BR      7$              ;;LOOP ON THE PRESENT TEST
3803  017012                  6$:;####END OF CODE FOR THE XOR TESTER#####
3804  017012  105737  001103  2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
3805  017016  001412                  BEQ     $SVLAD          ;;BR IF NO
3806  017020  032777  001000  162112  BIT     #BIT09,@$SWR    ;;LOOP ON ERROR?
3807  017026  001404                  BEQ     4$              ;;BR IF NO
3808  017030  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
3809  017036  000415                  BR      $OVER
3810  017040  105037  001103  4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
3811  017044  105237  001102  $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
3812  017050  011637  001106          MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
3813  017054  011637  001110          MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
3814  017060  005037  001204          CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
3815  017064  112737  000001  001115  MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
3816  017072  013777  001102  162042  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
3817  017100  013716  001106          MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
3818  017104  000002                  RTI                     ;;FIXES PS
3819
3820                                  ;;******************************************************
3821
3822
3823                                  .SBTTL  ERROR HANDLER ROUTINE
3824
3825                                  ;*SW15=1        HALT ON ERROR
3826                                  ;*SW13=1        INHIBIT ERROR TYPEOUTS
3827                                  ;*SW10=1        BELL ON ERROR
3828                                  ;*SW09=1        LOOP ON ERROR
3829                                  ;*SW12=1        CYCLE ON ERROR TO PREVIOS 'SCOPE' STATEMENT
3830                                  ;*GO TO ERRTYP ON ERROR
3831                                  ;*NOT FROM SYSMAC
3832
3833  017106  104407          $ERROR: CKSWR                           ;CHECK FOR SOFTWARE SWITCH REGISTER REQUEST
3834  017110  105237  001103  7$:     INCB    $ERFLG          ;SET THE ERROR FLAG
3835  017114  001775                  BEQ     7$              ;DON'T LET THE FLAG GO TO ZERO
3836  017116  013777  001102  162016  MOV     $TSTNM,@DISPLAY
3837  017124  032777  002000  162006  BIT     #SW10,@$SWR
3838  017132  001402                  BEQ     1$
3839  017134  104401  001206          TYPE    ,$BELL
3840  017140  005237  001112  1$:     INC     $ERTTL
3841  017144  011637  001116          MOV     (SP),$ERRPC
3842
3843  017150  032777  000004  161762  BIT     #SW2,@$SWR      ;DROP THE DRIVE?
3844  017156  001404                  BEQ     5$              ;SW NOT SET, SKIP
3845  017160  023727  001112  000006  CMP     $ERTTL,#6       ;MORE THAN 6 ERRORS ON THIS DRIVE?
3846  017166  101040                  BHI     6$              ;YES, DROP THE DRIVE
3847
3848  017170  162737  000002  001116  5$:     SUB     #2,$ERRPC
3849  017176  117737  161714  001114          MOVB    @#ERRPC,$ITEMB
```



```
3850  017204  032777  020000  161726          BIT     #SW13,@$SWR
3851  017212  001004                          BNE     2$
3852  017214  004737  017362                  JSR     PC,@#ERRTYP
3853  017220  104401  001213                  TYPE    ,$CRLF
3854  017224  005777  161710          2$:     TST     @$WR
3855  017230  100002                          BPL     3$
3856  017232  000000                          HALT
3857  017234  104407                          CKSWR                   ;CHECK FOR SOFTWARE SWITCH REGISTER REQUEST
3858  017236  032777  010000  161674  3$:     BIT     #SW12,@$SWR
3859  017244  001402                          BEQ     .+6
3860  017246  013716  001106                  MOV     $LPADR,(SP)
3861  017252  032777  001000  161660          BIT     #SW09,@$SWR
3862  017260  001402                          BEQ     4$
3863  017262  013716  001110                  MOV     $LPERR,(SP)
3864  017266  000002                  4$:     RTI
3865
3866  017270  013746  001226          6$:     MOV     DRVPTR,-(SP)    ;GET POINTER TO DRIVE #
3867  017274  162716  000002                  SUB     #2,(SP)
3868  017300  042736  000377                  BIC     #377,@(SP)+     ;CLEAR THE DRIVE PRESENT FLAG
3869  017304  104401  002064                  TYPE    ,MSG14
3870  017310  013746  001230                  MOV     DRIVAD,-(SP)
3871  017314  000241                          CLC                     ;GET THE DRIVE #
3872  017316  006116                          ROL     (SP)
3873  017320  006116                          ROL     (SP)
3874  017322  006116                          ROL     (SP)
3875  017324  006116                          ROL     (SP)
3876  017326  104402                          TYPOC                   ;TYPE IT OUT
3877  017330  104401  017336                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3878  017334  000405                          BR      64$             ;;GET OVER THE ASCIZ
3879                                  ;;65$:  .ASCIZ  / DROPPED/
3880  017350                          64$:
3881  017350  105337  001224                  DECB    DRIVS           ;DECRMNT # OF DRIVS PRESENT
3882  017354  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK
3883  017356  000137  015156                  JMP     BTEOP           ;EXIT
3884
3885  017362                          ERRTYP:
3886  017362  104401  001213                  TYPE    ,$CRLF          ;"CARRIAGE RETURN" & LINE FEED"
3887  017366  010046                          MOV     R0,-(SP)        ;SAVE R0
3888  017370  005000                          CLR     R0              ;PICKUP THE ITEM INDEX
3889  017372  153700  001114                  BISB    @#ITEMB,R0
3890  017376  001011                          BNE     1$              ;IF ITEM NUMBER IS ZERO, JUST
3891
3892                                                                  ;TYPE THE PC OF THE ERROR
3893  017400  013746  001116                  MOV     $ERRPC,-(SP)    ;SAVE $ERRPC FOR TYPEOUT
3894                                                                  ;ERROR ADDRESS
3895  017404  104402                          TYPOC                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3896  017406  104401                          TYPE
3897  017410  001733                          MSG7
3898  017412  013746  001116                  MOV     $ERRPC,-(SP)
3899  017416  104402                          TYPOC
3900  017420  000440                          BR      6$              ;GET OUT
3901  017422  005300                  1$:     DEC     R0              ;ADJUST THE INDEX SO THAT IT WILL
3902  017424  006300                          ASL     R0              ;       WORK FOR THE ERROR TABLE
3903  017426  006300                          ASL     R0
3904  017430  006300                          ASL     R0
3905  017432  062700  002122                  ADD     #@#ERRTB,R0     ;FORM TABLE POINTER
```

```
3906  017436  012037  017446          MOV     (R0)+,2$        ;PICKUP "ERROR MESSAGE" POINTER
3907  017442  001404                  BEQ     3$              ;SKIP TYPEOUT IF NOT POINTER
3908  017444  104401                  TYPE                    ;TYPE THE "ERROR MESSAGE"
3909  017446  000000          2$:     .WORD   0               ;"CARRIAGE RETURN" & LINE FEED"
3910  017450  104401  001213          TYPE    ,$CRLF          ;PICKUP "DATA HEADER" POINTER
3911  017454  032777  004000  161456  3$:     BIT     #SW11,@SWR      ;DUMP OUT ALL RK REGISTERS
3912  017462  001042                  BNE     10$             ;YES, BRANCH
3913  017464  012037  017474          MOV     (R0)+,4$        ;PICKUP "DATA HEADER" POINTER
3914  017470  001412                  BEQ     5$              ;SKIP TYPEOUT IF 0
3915  017472  104401                  TYPE                    ;TYPE THE "DATA HEADER"
3916  017474  000000          4$:     .WORD   0               ;"DATA HEADER" POINTER GOES HERE
3917  017476  104401  001213          TYPE    ,$CRLF          ;"CARRIAGE RETURN" & LINE FEED"
3918  017502  062700  000002          ADD     #2,R0           ;FORM POINTER TO TERMINATOR
3919  017506  005710                  TST     (R0)            ;IS THE TERMINATOR 0?
3920  017510  001017                  BNE     9$              ;IF NOT, BRANCH
3921  017512  162700  000002          SUB     #2,R0           ;YES, IT IS 0,  REPOINT TO "DATA"
3922                                                          ;GO TYPE OUT DATA AS USUAL
3923  017516  011000          5$:     MOV     (R0),R0         ;PICKUP "DATA TABLE" POINTER
3924  017520  001004                  BNE     7$              ;GO TYPE THE DATA
3925  017522  012600          6$:     MOV     (SP)+,R0        ;RESTORE R0
3926  017524  104401  001213          TYPE    ,$CRLF          ;"CARRIAGE RETURN" & LINE FEED"
3927  017530  000207                  RTS     PC              ;RETURN
3928  017532                  7$:
3929  017532  013046                  MOV     @(R0)+,-(SP)    ;SAVE @(R0)+ FOR TYPEOUT
3930  017534  104402                  TYPOC                   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
3931  017536  005710                  TST     (R0)            ;IS THERE ANOTHER NUMBER?
3932  017540  001770                  BEQ     6$              ;BR IF NO
3933  017542  104401  002110          TYPE    ,BLNKS2
3934  017546  000771                  BR      7$
3935  017550  004770  000000          9$:     JSR     PC,@(R0)        ;GO TO THE SPECIAL ERROR
3936                                                          ;DATA HANDLING SUBROUTINE
3937                                                          ;NOTE THAT THIS ROUTINE IS
3938                                                          ;THE ONE INDICATED IN THE
3939                                                          ;LAST WORD OF AN ERROR
3940                                                          ;ITEM IN THE ERROR TABLE
3941                                                          ;(STARTING AT $ERRTB)
3942  017554  104401                  TYPE
3943  017556  001733                  MSG7
3944  017560  013746  001116          MOV     $ERRPC,-(SP)
3945  017564  104402                  TYPOC
3946  017566  000755                  BR      6$              ;GO BACK, TO THE EXIT POINT
3947                                                          ;FOR 'ERRTYP'
3948
3949  017570  004737  017576  10$:    JSR     PC,DMPREG
3950  017574  000752                  BR      6$
3951
3952
3953
3954                          ;DMPREG
3955                          ;DUMPS OUT ALL RK REGISTERS WHEN SW 11 IS SET
3956
3957  017576                  DMPREG:
3958  017576  104401  017604          TYPE    ,65$            ;;TYPE ASCIZ STRING
3959  017602  000441                  BR      64$             ;;GET OVER THE ASCIZ
3960                          ;;65$:  .ASCIZ  <15><12>/  PC     RKDS    RKER    RKCS    RKWC    RKBA    RKDA    RKDB/<
3961  017706                  64$:
```



```
3962  017706  013746  001116          MOV     $ERRPC,-(SP)
3963  017712  104402                  TYPOC
3964  017714  104401  002110          TYPE    ,BLNKS2
3965  017720  010046                  MOV     R0,-(SP)
3966  017722  012700  001452          MOV     #RKDS,R0
3967  017726  013046          1$:     MOV     @(R0)+,-(SP)
3968  017730  104402                  TYPOC
3969  017732  104401  002110          TYPE    ,BLNKS2
3970  017736  020027  001466          CMP     R0,#RKDB
3971  017742  003771                  BLE     1$
3972  017744  012600                  MOV     (SP)+,R0
3973  017746  000207                  RTS     PC
3974
3975
3976
3977
3978
3979                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
3980
3981                          ;;**************************************************************
3982                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
3983                          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT, DEPENDING ON WHETHER THE
3984                          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
3985                          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
3986                          ;*REPLACED WITH SPACES.
3987                          ;*CALL:
3988                          ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
3989                          ;*      TYPDS                   ;;GO TO THE ROUTINE
3990
3991  017750                  $TYPDS:
3992  017750  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
3993  017752  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
3994  017754  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
3995  017756  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
3996  017760  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
3997  017762  012746  020200          MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
3998  017766  016605  000020          MOV     20(SP),R5       ;;GET THE INPUT NUMBER
3999  017772  100004                  BPL     1$              ;;BR IF INPUT IS POS.
4000  017774  005405                  NEG     R5              ;;MAKE THE BINARY NUMBER POS.
4001  017776  112766  000055  000001  MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
4002  020004  005000          1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
4003  020006  012703  020164          MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
4004  020012  112723  000040          MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
4005  020016  005002          2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
4006  020020  016001  020154          MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
4007  020024  160105          3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
4008  020026  002402                  BLT     4$              ;;BR IF DONE
4009  020030  005202                  INC     R2              ;;INCREASE THE BCD DIGIT BY 1
4010  020032  000774                  BR      3$
4011  020034  060105          4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
4012  020036  005702                  TST     R2              ;;CHECK IF BCD DIGIT=0
4013  020040  001002                  BNE     5$              ;;FALL THROUGH IF 0
4014  020042  105716                  TSTB    (SP)            ;;STILL DOING LEADING 0'S?
4015  020044  100407                  BMI     7$              ;;BR IF YES
4016  020046  106316          5$:     ASLB    (SP)            ;;MSD?
4017  020050  103003                  BCC     6$              ;;BR IF NO
```

```
 4018  020052  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 4019  020060  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 4020  020064  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 4021  020070  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
 4022  020072  005720                          TST     (R0)+           ;;JUST INCREMENTING
 4023  020074  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
 4024  020100  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
 4025  020102  003002                          BGT     8$              ;;GO TO EXIT
 4026  020104  010502                          MOV     R5,R2           ;;GET THE LSD
 4027  020106  000764                          BR      6$              ;;GO CHANGE TO ASCII
 4028  020110  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 4029  020112  100003                          BPL     9$              ;;BR IF NO
 4030  020114  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 4031  020122  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 4032  020124  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
 4033  020126  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
 4034  020130  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
 4035  020132  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
 4036  020134  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 4037  020136  104401  020164                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 4038  020142  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 4039  020150  012616                          MOV     (SP)+,(SP)
 4040  020152  000002                          RTI                     ;;RETURN TO USER
 4041  020154  023420                  $DTBL:  10000.
 4042  020156  001750                          1000.
 4043  020160  000144                          100.
 4044  020162  000012                          10.
 4045  020164  000004                  $DBLK:  .BLKW   4
 4046
 4047                                  .SBTTL  TYPE ROUTINE
 4048
 4049                                  ;;*****************************************************************
 4050                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 4051                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 4052                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 4053                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 4054                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 4055                                  ;*
 4056                                  ;*CALL:
 4057                                  ;*1) USING A TRAP INSTRUCTION
 4058                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 4059                                  ;*OR
 4060                                  ;*      TYPE
 4061                                  ;*      MESADR
 4062                                  ;*
 4063
 4064  020174  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 4065  020200  100002                          BPL     1$              ;;BR IF YES
 4066  020202  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
 4067  020204  000407                          BR      3$              ;;LEAVE
 4068  020206  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
 4069  020210  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 4070  020214  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 4071  020216  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 4072  020220  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 4073  020222  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
```



```
 4074  020224  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 4075  020230  000002                          RTI                     ;;RETURN
 4076  020232  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 4077  020236  001430                          BEQ     8$
 4078  020240  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 4079  020244  001006                          BNE     5$
 4080  020246  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
 4081  020250  104401                          TYPE                    ;;TYPE A CR AND LF
 4082  020252  001213                          $CRLF
 4083  020254  105037  020410                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 4084  020260  000755                          BR      2$              ;;GET NEXT CHARACTER
 4085  020262  004737  020344          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 4086  020266  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 4087  020272  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
 4088  020274  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
 4089                                                                  ;;AND THE NULL CHAR.
 4090  020300  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 4091  020304  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 4092  020306  004737  020344                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
 4093  020312  105337  020410                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 4094  020316  000770                          BR      7$              ;;LOOP
 4095
 4096                                  ;HORIZONTAL TAB PROCESSOR
 4097
 4098  020320  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 4099  020324  004737  020344          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 4100  020330  132737  000007  020410          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 4101  020336  001372                          BNE     9$              ;;TAB STOP
 4102  020340  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
 4103  020342  000724                          BR      2$              ;;GET NEXT CHARACTER
 4104  020344  105777  160600          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 4105  020350  100375                          BPL     $TYPEC
 4106  020352  116677  000002  160572          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 4107  020360  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 4108  020366  001003                          BNE     1$              ;;BRANCH IF NO
 4109  020370  105037  020410                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 4110  020374  000406                          BR      $TYPEX          ;;EXIT
 4111  020376  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 4112  020404  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
 4113  020406  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
 4114  020410  000000                  $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 4115  020412  000207                  $TYPEX: RTS     PC
 4116
 4117
 4118
 4119                                  .SBTTL  INTEGER MULTIPLY ROUTINE
 4120
 4121                                  ;;*****************************************************************
 4122                                  ;*CALL
 4123                                  ;*      MOV     MULTIPLER,-(SP)
 4124                                  ;*      MOV     MULTIPLICAND,-(SP)
 4125                                  ;*      JSR     PC,@#$MULT
 4126                                  ;*      RETURN  ;;PRODUCT IS ON THE STACK
 4127                                  ;*
 4128                                  ;*      STACK   PRODUCT
 4129                                  ;*      -----   -------
```

```
                                ;*      TOP     LSB'S
                                ;*      +2      MSB'S
                                ;
020414                          $MULT:
020414  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
020416  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
020420  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
020422  005046                          CLR     -(SP)           ;;CLEAR THE SIGN KEY
020424  016601  000012                  MOV     12(SP),R1       ;;GET THE MULTIPLICAND
020430  100002                          BPL     1$              ;;BR IF PLUS
020432  005216                          INC     (SP)            ;;SET THE SIGN KEY
020434  005401                          NEG     R1              ;;MAKE THE MULTIPLICAND POSTIVE
020436  016602  000014          1$:     MOV     14(SP),R2       ;;GET THE MULTIPLIER
020442  100002                          BPL     2$              ;;BR IF PLUS
020444  005316                          DEC     (SP)            ;;UPDATE THE SIGN KEY
020446  005402                          NEG     R2              ;;MAKE THE MULTIPLIER POSTIVE
020450  012746  000021          2$:     MOV     #17.,-(SP)      ;;SET THE LOOP COUNT
020454  005000                          CLR     R0              ;;SETUP FOR THE MULTIPLY LOOP
020456  103001                  3$:     BCC     4$              ;;DON'T ADD IF MULTIPLICAND = 0
020460  060200                          ADD     R2,R0
020462  006000                  4$:     ROR     R0              ;;POSITION THE PARITIAL PRODUCT AND
020464  006001                          ROR     R1              ;;THE MULTIPLICAND
020466  005316                          DEC     (SP)            ;;HAS ALL BITS OF THE MULTIPLICAND BEEN DONE?
020470  001372                          BNE     3$              ;;BR IF NO
020472  022616                          CMP     (SP)+,(SP)      ;;SHOULD PRODUCT BE NEGATIVE?
020474  001403                          BEQ     5$              ;;GO TO EXIT IF NO
020476  005400                          NEG     R0              ;;YES--SO MAKE IT SO
020500  005401                          NEG     R1
020502  005600                          SBC     R0
020504  005726                  5$:     TST     (SP)+           ;;CLEAR SIGN INFO, OFF OF STACK
020506  010066  000012                  MOV     R0,12(SP)       ;;PUT THE PRODUCT ON THE STACK (MSB'S)
020512  010166  000010                  MOV     R1,10(SP)       ;;LSB'S
020516  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
020520  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
020522  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
020524  000207                          RTS     PC
                                
                                .SBTTL  TTY INPUT ROUTINE
                                
                                ;;*************************************************************
                                .ENABL  LSB
020526  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
020530  000000                  $TKQIN: .WORD   0               ;;INPUT POINTER
020532  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
020534  000001                  $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
        020535                  $TKQEND=.
        020536                  .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
                                ;*      RETURN
                                ;
```



```
020536  005037  020526        $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
020542  012737  020534  020530        MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
020550  013737  020530  020532        MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
020556  012737  020606  000060        MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
020564  012737  000200  000062        MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
020572  005777  160350                TST     @$TKB           ;;CLEAR DONE FLAG
020576  012777  000100  160340        MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
020604  000207                        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.
                                ;
020606  117746  160334        $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
020612  042716  177600                BIC     #^C177,(SP)     ;;STRIP THE JUNK
020616  021627  000007          1$:   CMP     (SP),#7         ;;IS IT A CONTROL G?
020622  001004                        BNE     2$              ;;BRANCH IF NO
020624  022737  000176  001140        CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
020632  001500                        BEQ     6$              ;;GO TO SWR CHANGE
                                
020634                          2$:
020634  022737  000001  020526        CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
020642  001004                        BNE     3$              ;;BRANCH IF NO
020644  104401  001206                TYPE    ,$BELL          ;;RING THE TTY BELL
020650  005726                        TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
020652  000451                        BR      5$              ;;EXIT
020654  021627  000023          3$:   CMP     (SP),#23        ;;IS IT A CONTROL-S?
020660  001021                        BNE     32$             ;;BRANCH IF NO
020662  005077  160256                CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
020666  005726                        TST     (SP)+           ;;CLEAN CHAR OFF STACK
020670  105777  160250          31$:  TSTB    @$TKS           ;;WAIT FOR A CHAR
020674  100375                        BPL     31$             ;;LOOP UNTIL ITS THERE
020676  117746  160244                MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
020702  042716  177600                BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
020706  022627  000021                CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
020712  001366                        BNE     31$             ;;BRANCH IF NO
020714  012777  000100  160222        MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
020722  000002                        RTI                     ;;RETURN
020724  005237  020526          32$:  INC     $TKCNT          ;;COUNT THIS CHARACTER
020730  021627  000140                CMP     (SP),#140       ;;IS IT UPPER CASE?
020734  002405                        BLT     4$              ;;BRANCH IF YES
020736  021627  000175                CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
020742  003002                        BGT     4$              ;;BRANCH IF YES
020744  042716  000040                BIC     #40,(SP)        ;;MAKE IT UPPER CASE
020750  112677  177554          4$:   MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
020754  005237  020530                INC     $TKQIN          ;;UPDATE THE POINTER
020760  023727  020530  020535        CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
020766  001003                        BNE     5$              ;;BRANCH IF NO
020770  012737  020534  020530        MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
020776  000002                  5$:   RTI                     ;;RETURN
                                
                                ;;*************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
```

```
                                ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
021000  022737  000176  001140  $CKSWR: CMP     @#SWREG,SWR     ;;IS THE SOFT-SWR SELECTED
021006  001104                         BNE     15$             ;;EXIT IF NOT
021010  105777  160130                 TSTB    @#TKS           ;;IS A CHAR WAITING?
021014  100101                         BPL     15$             ;;IF NOT, EXIT
021016  117746  160124                 MOVB    @#TKB,-(SP)     ;;YES
021022  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
021026  021627  000007                 CMP     (SP),#7         ;;IS IT A CONTROL-G?
021032  001300                         BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                ;;AND EXIT

                                ;;**************************************************************
                                ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
021034  123727  001134  000001  6$:      CMPB    @AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
021042  001674                         BEQ     2$              ;;BRANCH IF YES
021044  005726                         TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
021046  004737  020536                 JSR     PC,@TKINT       ;;FLUSH THE TTY INPUT QUEUE
021052  005077  160066                 CLR     @#TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
021056  112737  000001  001135         MOVB    #1,@INTAG       ;;SET INTERRUPT MODE INDICATOR

021064  104401  021643                 TYPE    ,@CNTLG         ;;ECHO THE CONTROL-G (^G)
021070  104401  021650  $GTSWR: TYPE    ,@MSWR          ;;TYPE CURRENT CONTENTS
021074  013746  000176                 MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
021100  104402                         TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
021102  104401  021661                 TYPE    ,@MNEW          ;;PROMPT FOR NEW SWR
021106  005046                 19$:     CLR     -(SP)           ;;CLEAR COUNTER
021110  005046                         CLR     -(SP)           ;;THE NEW SWR
021112  105777  160026                 7$:      TSTB    @#TKS           ;;CHAR THERE?
021116  100375                         BPL     7$              ;;IF NOT TRY AGAIN

021120  117746  160022                 MOVB    @#TKB,-(SP)     ;;PICK UP CHAR
021124  042716  177600                 BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII


021130  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
021134  001005                         BNE     10$             ;;BRANCH IF NOT
021136  104401  021636                 TYPE    ,@CNTLU         ;;YES, ECHO CONTROL-U (^U)
021142  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
021146  000757                         BR      19$             ;;LET'S TRY IT AGAIN


021150  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
021154  001022                         BNE     16$             ;;BRANCH IF NO
021156  005766  000004                 TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
021162  001403                         BEQ     11$             ;;BRANCH IF YES
021164  016677  000002  157746         MOV     2(SP),@SWR      ;;SAVE NEW SWR
021172  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
021176  104401  001213          14$:    TYPE    ,@CRLF          ;;ECHO <CR> AND <LF>
021202  123727  001135  000001         CMPB    @INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
021210  001003                         BNE     15$             ;;BRANCH IF NOT
021212  012777  000100  157724         MOV     #100,@#TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
021220  000002                 15$:    RTI                     ;;RETURN
021222  004737  020344          16$:    JSR     PC,@TYPEC       ;;ECHO CHAR
```



```
021226  021627  000060                 CMP     (SP),#60        ;;CHAR < 0?
021232  002420                         BLT     18$             ;;BRANCH IF YES
021234  021627  000067                 CMP     (SP),#67        ;;CHAR > 7?
021240  003015                         BGT     18$             ;;BRANCH IF YES
021242  042726  000060                 BIC     #60,(SP)+       ;;STRIP-OFF ASCII
021246  005766  000002                 TST     2(SP)           ;;IS THIS THE FIRST CHAR
021252  001403                         BEQ     17$             ;;BRANCH IF YES
021254  006316                         ASL     (SP)            ;;NO, SHIFT PRESENT
021256  006316                         ASL     (SP)            ;;   CHAR OVER TO MAKE
021260  006316                         ASL     (SP)            ;;   ROOM FOR NEW ONE.
021262  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
021266  056616  177776                 BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
021272  000707                         BR      7$              ;;GET THE NEXT ONE
021274  104401  001212          18$:    TYPE    ,@QUES          ;;TYPE ?<CR><LF>
021300  000720                         BR      20$             ;;SIMULATE CONTROL-U
                                ,DSABL  LSB


                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                ;*CALL:
                                ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                ;

021302  011646                 $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
021304  016666  000004  000002         MOV     4(SP),2(SP)     ;;THE PS
021312  005066  000004                 CLR     4(SP)           ;;GET READY FOR A CHARACTER
021316  005046                         CLR     -(SP)           ;;PUT NEW PS ON STACK
021320  012746  021326                 MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
021324  000002                         RTI                     ;;POP NEW PC AND PS
021326                          64$:
021326  005737  020526          1$:     TST     @TKCNT          ;;WAIT ON A CHARACTER
021332  001775                         BEQ     1$
021334  005337  020526                 DEC     @TKCNT          ;;DECREMENT THE COUNTER
021340  117766  177166  000004         MOVB    @#TKQOUT,4(SP)  ;;GET ONE CHARACTER
021346  005237  020532                 INC     @TKQOUT         ;;UPDATE THE POINTER
021352  023727  020532  020535         CMP     @TKQOUT,@#TKQEND ;;DID IT GO OFF OF THE END?
021360  001003                         BNE     2$              ;;BRANCH IF NO
021362  012737  020534  020532         MOV     @#TKQ&RT,@TKQOUT ;;RESET THE POINTER
021370  000002                 2$:     RTI                     ;;RETURN
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

021372  010346                 $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
021374  005046                         CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
021376  012703  021626          1$:     MOV     #@TTYIN,R3      ;;GET ADDRESS
021402  022703  021636          2$:     CMP     #@TTYIN+8.,R3   ;;BUFFER FULL?
021406  101456                         BLOS    4$              ;;BR IF YES
021410  104410                         RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
021412  112613                         MOVB    (SP)+,(R3)      ;;GET CHARACTER
```

```
4354  021414  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
4355  021420  001022                          BNE     5$              ;;BR IF NO
4356  021422  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
4357  021424  001007                          BNE     6$              ;;BR IF NO
4358  021426  112737  000134  021624          MOVB    #'\,9$          ;;TYPE A BACK SLASH
4359  021434  104401  021624                  TYPE    ,9$
4360  021440  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
4361  021444  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
4362  021446  020327  021626                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
4363  021452  103434                          BLO     4$              ;;BR IF YES
4364  021454  111337  021624                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
4365  021460  104401  021624                  TYPE    ,9$             ;;GO TYPE
4366  021464  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
4367  021466  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
4368  021470  001406                          BEQ     7$              ;;BR IF NO
4369  021472  112737  000134  021624          MOVB    #'\,9$          ;;TYPE A BACK SLASH
4370  021500  104401  021624                  TYPE    ,9$
4371  021504  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
4372  021506  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
4373  021512  001003                          BNE     8$              ;;BR IF NO
4374  021514  104401  021636                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
4375  021520  000726                          BR      1$              ;;GO START OVER
4376  021522  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
4377  021526  001011                          BNE     3$              ;;BRANCH IF NO
4378  021530  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
4379  021532  104401  001213                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
4380  021536  104401  021626                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
4381  021542  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
4382  021544  104401  001212          4$:     TYPE    ,$QUES          ;;TYPE A '?'
4383  021550  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
4384  021552  111337  021624          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
4385  021556  104401  021624                  TYPE    ,9$
4386  021562  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
4387  021566  001305                          BNE     2$              ;;LOOP IF NOT RETURN
4388  021570  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
4389  021574  104401  001214                  TYPE    ,$LF            ;;TYPE A LINE FEED
4390  021600  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
4391  021602  012603                          MOV     (SP)+,R3        ;;RESTORE R3
4392  021604  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
4393  021606  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
4394  021614  012766  021626  000004          MOV     #$TTYIN,4(SP)
4395  021622  000002                          RTI                     ;;RETURN
4396  021624     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
4397  021625     000                          .BYTE   0               ;;TERMINATOR
4398  021626  000010          $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
4399  021636  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
4400  021643     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
4401  021650  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
4402  021656  020075     000
4403  021661     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
4404  021666  036440  000040
4405
4406                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
4407
4408                                  ;;*****************************************************************
4409                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
```



```
4410                                  ;*CHANGE IT TO BINARY.
4411                                  ;*CALL:
4412                                  ;*      RDOCT                   ;;READ AN OCTAL NUMBER
4413                                  ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
4414                                  ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
4415
4416  021672  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
4417  021674  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
4418  021702  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4419  021704  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4420  021706  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4421  021710  104411                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
4422  021712  012600                          MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
4423  021714  005001                          CLR     R1              ;;CLEAR DATA WORD
4424  021716  005002                          CLR     R2
4425  021720  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
4426  021722  001412                          BEQ     3$              ;;IF ZERO GET OUT
4427  021724  006301                          ASL     R1              ;;*2
4428  021726  006102                          ROL     R2
4429  021730  006301                          ASL     R1              ;;*4
4430  021732  006102                          ROL     R2
4431  021734  006301                          ASL     R1              ;;*8
4432  021736  006102                          ROL     R2
4433  021740  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
4434  021744  062601                          ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
4435  021746  000764                          BR      2$              ;;LOOP
4436  021750  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
4437  021752  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
4438  021756  010237  021772                  MOV     R2,$HIOCT
4439  021762  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
4440  021764  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
4441  021766  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
4442  021770  000002                          RTI                     ;;RETURN
4443  021772  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
4444
4445                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
4446
4447                                  ;;*****************************************************************
4448                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
4449                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
4450                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
4451                                  ;*CALL:
4452                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
4453                                  ;*      TYPOS                   ;;CALL FOR TYPEOUT
4454                                  ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
4455                                  ;*      .BYTE   N               ;;N=1 OR 0
4456                                  ;*                                      ;;1=TYPE LEADING ZEROS
4457                                  ;*                                      ;;0=SUPPRESS LEADING ZEROS
4458                                  ;*
4459                                  ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
4460                                  ;*$TYPOS OR $TYPOC
4461                                  ;*CALL:
4462                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
4463                                  ;*      TYPON                   ;;CALL FOR TYPEOUT
4464                                  ;*
4465                                  ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
```

```
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
                                        ;
022774  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
022000  116637  000001  022217          MOVB    1(SP),$0FILL    ;;LOAD ZERO FILL SWITCH
022006  112637  022221                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
022012  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
022016  000406                          BR      $TYPON
022020  112737  000001  022217  $TYPOC: MOVB    #1,$0FILL       ;;SET THE ZERO FILL SWITCH
022026  112737  000006  022221          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
022034  112737  000005  022216  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
022042  010346                          MOV     R3,-(SP)        ;;SAVE R3
022044  010446                          MOV     R4,-(SP)        ;;SAVE R4
022046  010546                          MOV     R5,-(SP)        ;;SAVE R5
022050  113704  022221                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
022054  005404                          NEG     R4
022056  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
022062  110437  022220                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
022066  113704  022217                  MOVB    $0FILL,R4       ;;GET THE ZERO FILL SWITCH
022072  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
022076  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
022100  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
022102  000404                          BR      3$              ;;GO DO MSB
022104  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
022106  006105                          ROL     R5
022110  006105                          ROL     R5
022112  010503                          MOV     R5,R3
022114  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
022116  105337  022220                  DECB    $OMODE          ;;TYPE THIS DIGIT?
022122  100016                          BPL     7$              ;;BR IF NO
022124  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
022130  001002                          BNE     4$              ;;TEST FOR 0
022132  005704                          TST     R4              ;;SUPPRESS THIS 0?
022134  001403                          BEQ     5$              ;;BR IF YES
022136  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
022140  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
022144  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
022150  110337  022214                  MOVB    R3,6$           ;;SAVE FOR TYPING
022154  104401  022214                  TYPE    ,6$             ;;GO TYPE THIS DIGIT
022160  105337  022216          7$:     DECB    $OCNT           ;;COUNT BY 1
022164  003347                          BGT     2$              ;;BR IF MORE TO DO
022166  002402                          BLT     6$              ;;BR IF DONE
022170  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
022172  000744                          BR      2$              ;;GO DO THE LAST DIGIT
022174  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
022176  012604                          MOV     (SP)+,R4        ;;RESTORE R4
022200  012603                          MOV     (SP)+,R3        ;;RESTORE R3
022202  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
022210  012616                          MOV     (SP)+,(SP)
022212  000002                          RTI                     ;;RETURN
022214     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
022215     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
022216     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
022217     000                  $0FILL: .BYTE   0               ;;ZERO FILL SWITCH
022220  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```



```


                                                .SBTTL  TYPDSS - TYPE DECIMAL, LEADING ZEROES SUPPRESSED
                                        ;TYPDSS
                                        ;ROUTINE FOR TYPING OUT DECIMAL NUMBERS, LEADING 0'S ARE SUPPRESSED
                                        ;THE NUMBER IS LEFT JUSTIFIED. NOTE THE 16 BIT BINARY NUMBER SHOULD
                                        ;BE POSITIVE (BIT 15= 0).
                                        ;CALL:  MOV     NUMBER,-(SP)    ;PUT BINARY NUMBER ON STACK
                                        ;       TYPDSS                  ;GO TYPE DECIMAL
                                        ;
022222  016637  000004  022262  TYPDES: MOV     4(SP),1$                ;GET THE NUMBER
022230  012746  022262                  MOV     #1$,-(SP)               ;PUT PTR ON THE STACK
022234  004737  022422                  JSR     PC,@#$DB2D              ;GO CONVERT BINARY NO. TO
                                                                        ;ASCII STRING
022240  004737  022266                  JSR     PC,@#$SUPRS             ;GO TYPE OUT DECIMAL STRING
                                                                        ;SUPRESING LEADING 0'S
022244  016666  000002  000004          MOV     2(SP),4(SP)             ;ADJUST RETURN
022252  011666  000002                  MOV     (SP),2(SP)              ;ADJUST RETURN ADRES
022256  005726                          TST     (SP)+                   ;POP STACK
022260  000002                          RTI                             ;RETURN
                                        ;
022262  000000  000000          1$:     .WORD   0,0
                                        ;
                                        ;
                                        .SBTTL  TYPE NUMERICAL ASCIZ STRING SUPPRESS LEADING ZEROS
                                        ;
                                        ;;*****************************************************************
                                        ;*THIS ROUTINE IS USED TO TYPE AN ASCIZ NUMBER SUPPRESSING THE
                                        ;*LEADING NUMBERS.
                                        ;*CALL
                                        ;*      MOV     #NUMADR,-(SP)   ;;FIRST ADDRESS OF ASCIZ STRING
                                        ;*      JSR     PC,@#$SUPRS
                                        ;
                                        ;
022266  010046                  $SUPRS: MOV     R0,-(SP)        ;;SAVE R0
022270  016600  000004                  MOV     4(SP),R0        ;;PICKUP THE POINTER
022274  105710                  1$:     TSTB    (R0)            ;;TERMINATEOR?
022276  001403                          BEQ     2$              ;;BR IF YES
022300  122720  000060                  CMPB    #'0,(R0)+       ;;IS THIS AN ASCII "0" ?
022304  001773                          BEQ     1$              ;;BR IF YES
022306  005300                  2$:     DEC     R0              ;;BACKUP BY "1"
022310  010037  022316                  MOV     R0,3$           ;;SAVE FOR TYPING
022314  104401                          TYPE                    ;;GO TYPE
022316  000000                  3$:     .WORD   0               ;;ASCIZ POINTER GOES HERE
022320  012600                          MOV     (SP)+,R0        ;;RESTORE R0
022322  012616                          MOV     (SP)+,(SP)      ;;RESTORE THE STACK
022324  000207                          RTS     PC              ;;RETURN
                                        ;
                                        .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
                                        ;
                                        ;;*****************************************************************
                                        ;*SAVE R0-R5
                                        ;*CALL:
                                        ;*      SAVREG
                                        ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                        ;*
```

```
                                ;*TOP---(+16)
                                ;* +2---(+18)
                                ;* +4---R5
                                ;* +6---R4
                                ;* +8---R3
                                ;*+10---R2
                                ;*+12---R1
                                ;*+14---R0
                                $SAVREG:
022326 010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
022330 010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
022332 010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
022334 010346                           MOV     R3,-(SP)        ;;PUSH R3 ON STACK
022336 010446                           MOV     R4,-(SP)        ;;PUSH R4 ON STACK
022340 010546                           MOV     R5,-(SP)        ;;PUSH R5 ON STACK
022342 016646 000022                    MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
022346 016646 000022                    MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
022352 016646 000022                    MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
022356 016646 000022                    MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
022362 000002                           RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
                                $RESREG:
022364 012666 000022                    MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
022370 012666 000022                    MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
022374 012666 000022                    MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
022400 012666 000022                    MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
022404 012605                           MOV     (SP)+,R5        ;;POP STACK INTO R5
022406 012604                           MOV     (SP)+,R4        ;;POP STACK INTO R4
022410 012603                           MOV     (SP)+,R3        ;;POP STACK INTO R3
022412 012602                           MOV     (SP)+,R2        ;;POP STACK INTO R2
022414 012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
022416 012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
022420 000002                           RTI

                                .SBTTL  DOUBLE LENGTH BINARY TO DECIMAL ASCII CONVERT ROUTINE

                                ;;**************************************************************
                                ;*THIS ROUTINE WILL CONVERT A 32-BIT BINARY NUMBER TO AN UNSIGNED
                                ;*DECIMAL (ASCII) NUMBER. THE SIGN OF THE BINARY NUMBER MUST BE
                                ;*POSITIVE.
                                ;*CALL
                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
                                ;*      JSR     PC,@#$DB2D
                                ;*      RETURN                  ;;THE FIRST ADDRESS OF ASCIZ
                                                                ;;IS ON THE STACK

022422 104413                   $DB2D:  SAVREG                  ;;SAVE REGISTERS
022424 016602 000002                    MOV     2(SP),R2        ;;PICKUP THE DATA POINTER
022430 012700 022602                    MOV     #$DECVL,R0      ;;GET ADDRESS OF "$DECVL" STRING
022434 010066 000002                    MOV     R0,2(SP)        ;;PUT ADDRESS OF ASCIZ STRING ON STACK
022440 012201                           MOV     (R2)+,R1        ;;PICKUP THE BINARY NUMBER
```



```
022442 012202                           MOV     (R2)+,R2
022444 012737 000012 022520             MOV     #10.,4$         ;;SET UP TO DO 10 CONVERSIONS
022452 012704 022532                    MOV     #$TNPWR,R4      ;;ADDRESS OF TEN POWER
022456 012705 022534                    MOV     #$TNPWR+2,R5
022462 005003                   1$:     CLR     R3              ;;CLEAR PARTIAL
022464 161401                   2$:     SUB     (R4),R1         ;;SUBTRACT TEN POWER
022466 005602                           SBC     R2
022470 161502                           SUB     (R5),R2
022472 002402                           BLT     3$              ;;BR IF TEN POWER TO LARGE
022474 005203                           INC     R3              ;;ADD 1 TO PARTIAL
022476 000772                           BR      2$              ;;LOOP
022500 062401                   3$:     ADD     (R4)+,R1        ;;RESTORE SUBTRACTED VALUE
022502 005502                           ADC     R2
022504 062402                           ADD     (R4)+,R2
022506 022525                           CMP     (R5)+,(R5)+     ;;MOVE TO NEXT TEN POWER
022510 052703 000060                    BIS     #'0,R3          ;;CHANGE PARTIAL TO ASCII
022514 110320                           MOVB    R3,(R0)+        ;;SAVE IT
022516 005327                           DEC     (PC)+           ;;DONE?
022520 000000                   4$:     .WORD   0
022522 001357                           BNE     1$              ;;BR IF NO
022524 105020                           CLRB    (R0)+           ;;TERMINATOR
022526 104414                           RESREG                  ;;RESTORE REGISTERS
022530 000207                           RTS     PC              ;;RETURN
022532 145000                   $TNPWR: 145000                  ;;1.0E09
022534 035632                           35632
022536 160400                           160400                  ;;1.0E08
022540 002765                           2765
022542 113200                           113200                  ;;1.0E07
022544 000230                           230
022546 041100                           041100                  ;;1.0E06
022550 000017                           17
022552 103240                           103240                  ;;1.0E05
022554 000001                           1
022556 023420                           23420                   ;;1.0E04
022560 000000                           0
022562 001750                           1750                    ;;1.0E03
022564 000000                           0
022566 000144                           144                     ;;1.0E02
022570 000000                           0
022572 000012                           12                      ;;1.0E01
022574 000000                           0
022576 000001                           1                       ;;1.0E00
022600 000000                           0
022602 000014                   $DECVL: .BLKB   12.             ;;RESERVE STORAGE FOR ASCIZ STRING


                                .SBTTL  TRAP DECODER

                                ;;**************************************************************
                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                                ;*GO TO THAT ROUTINE.

022616 010046                   $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
```

```
4690  022620  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
4691  022624  005740                  TST     -(R0)           ;;BACKUP BY 2
4692  022626  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
4693  022630  006300                  ASL     R0              ;;POSITION FOR INDEXING
4694  022632  016000  022652          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
4695  022636  000200                  RTS     R0              ;;GO TO ROUTINE
4696
4697
4698                                  ;;THIS IS USE TO HANDLE THE "GETPPI" MACRO
4699
4700  022640  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
4701  022642  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
4702  022650  000002                  RTI                     ;;RESTORE THE PSW
4703
4704                                  .SBTTL  TRAP TABLE
4705
4706                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
4707                                  ;*BY THE "TRAP" INSTRUCTION.
4708
4709                                  ;       ROUTINE
4710                                  ;       -------
4711  022652  022640                  $TRPAD: .WORD   $TRAP2
4712  022654  020174                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
4713  022656  022020                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
4714  022660  021774                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
4715  022662  022034                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
4716  022664  017750                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
4717
4718  022666  021070                          $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
4719
4720  022670  021000                          $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
4721  022672  021302                          $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
4722  022674  021372                          $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
4723  022676  021672                          $RDOCT  ;;CALL=RDOCT    TRAP+12(104412) READ AN OCTAL NUMBER FROM TTY
4724  022700  022326                          $SAVREG ;;CALL=SAVREG   TRAP+13(104413) SAVE R0-R5 ROUTINE
4725  022702  022364                          $RESREG ;;CALL=RESREG   TRAP+14(104414) RESTORE R0-R5 ROUTINE
4726
4727  022704  016526                          CN.RST  ;;CALL=CON,RESET        TRAP+15(104415) CONTROL RESET ROUTINE
4728
4729  022706  016414                          DR.RST  ;;CALL=DRV,RESET        TRAP+16(104416) DRIVE RESET ROUTINE
4730
4731  022710  015770                          MSGE    ;;CALL=MESAGE   TRAP+17(104417) MESSAGE HANDLER
4732
4733  022712  016032                          TY.MSG  ;;CALL=TYPMSG   TRAP+20(104420) MESSAGE TYPEOUT ROUTINE
4734
4735  022714  016544                          CN.RDY  ;;CALL=CON,RDY  TRAP+21(104421) WAIT FOR CONTROL READY
4736
4737  022716  016642                          TSTRWS  ;;CALL=TST.RWS  TRAP+22(104422) TEST R/W/S RDY SET
4738
4739  022720  016440                          RES.DO  ;;CALL=RESDON   TRAP+23(104423) DRIVE RESET DONE?
4740
4741  022722  022222                          TYPDES  ;;CALL=TYPDSS   TRAP+24(104424) TYPE DECIMAL, SUPRES LDG 0'S
4742
4743
4744                                  .SBTTL  POWER DOWN AND UP ROUTINES
4745
```



```
4746                                  ;;*****************************************************************
4747                                  ;POWER DOWN ROUTINE
4748  022724  012737  023070  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
4749  022732  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
4750  022740  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4751  022742  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4752  022744  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4753  022746  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
4754  022750  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
4755  022752  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
4756  022754  017746  156160                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
4757  022760  010637  023074                  MOV     SP,$SAVR6       ;;SAVE SP
4758  022764  012737  022776  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
4759  022772  000000                          HALT
4760  022774  000776                          BR      .-2             ;;HANG UP
4761
4762                                  ;;*****************************************************************
4763                                  ;POWER UP ROUTINE
4764  022776  012737  023070  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
4765  023004  013706  023074                  MOV     $SAVR6,SP       ;;GET SP
4766  023010  005037  023074                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
4767  023014  005237  023074          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
4768  023020  001375                          BNE     1$              ;;OF  WORD
4769  023022  012677  156112                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
4770  023026  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
4771  023030  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
4772  023032  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
4773  023034  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
4774  023036  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
4775  023040  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
4776  023042  012737  022724  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
4777  023050  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
4778  023056  104401                          TYPE                    ;REPORT THE POWER FAILURE
4779  023060  023076                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
4780  023062  012716                          MOV     (PC)+,(SP)      ;;RESTART AT PWRFL
4781  023064  004170                  $PWRAD: .WORD   PWRFL           ;;RESTART ADDRESS
4782  023066  000002                          RTI
4783  023070  000000                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
4784  023072  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
4785  023074  000000                  $SAVR6: 0                       ;;PUT THE SP HERE
4786  023076  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
4787  023104  000122
4788                                          .EVEN
4789
4790
```

```
                                .SBTTL  FUNCTION SELECTION PROGRAM

                        ;THIS IS THE FUNCTION SELECTION PROGRAM.
                        ;ON ENTERING THIS SUB-PROGRAM THE FIRST QUESTION ASKED IS
                        ;       FUNCTION?       IN REPLY TYPE IN ONE OF THE FOLLOWING COMMANDS.
                        ;COMMANDS:      CR = CONTROL RESET
                        ;               DR = DRIVE RESET
                        ;               SK = SEEK
                        ;               WR = WRITE
                        ;               RD = READ
                        ;               WC = WRITE CHECK
                        ;               RC = READ CHECK
                        ;TERMINATE EVERY COMMAND WITH <CR>. FURTHER QUESTIONS (RK8A? , RKDA? ,
                        ;#WORDS? ) WILL BE ASKED. TYPE IN THE BUS ADDRESS (OCTAL), DISK ADDRESS
                        ;(OCTAL), AND NUMBER OF WORDS TO BE TRANSFERRED (OCTAL). IF A NON-EXISTENT
                        ;CYLINDER OR A NON-EXISTENT SECTOR IS TYPED IN, THE QUESTION (RKDA?) WILL
                        ;BE ASKED AGAIN.

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

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

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

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

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

                                                        ;R2 CONTAINS RKCS CONTENTS
                                                        ;R3 CONTAINS RKDA CONTENTS
                                                        ;R4,R5 CONTAIN THE CYLINDER #S BETWEEN
                                                        ;WHICH SEEK IS  TO BE DONE.
 023106                         FUNBEG:
 023106  104401  023114                 TYPE    ,65$            ;;TYPE ASCIZ STRING
 023112  000407                         BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <15><12>/FUNCTION? /
 023132                         64$:
 023132  104411                         RDLIN
```



```
 023134  012600                         MOV     (SP)+,R0
 023136  112001                         MOVB    (R0)+,R1
 023140  120127  000127                 CMPB    R1,#'W
 023144  001026                         BNE     2$
 023146  121027  000122                 CMPB    (R0),#'R
 023152  001010                         BNE     1$
 023154  004737  024000                 JSR     PC,CHKSW0       ;CHECK SW 0 SET?
 023160  012702  004003                 MOV     #4003,R2
 023164  000536                         BR      NXTDA

 023166  012702  000003         9$:     MOV     #3,R2
 023172  000521                         BR      NXTBA
 023174  121027  000103         1$:     CMPB    (R0),#'C
 023200  001342                         BNE     FUNBEG
 023202  004737  024000                 JSR     PC,CHKSW0       ;CHECK SW 0 SET?
 023206  012702  004007                 MOV     #4007,R2
 023212  000523                         BR      NXTDA

 023214  012702  000007                 MOV     #7,R2
 023220  000506                         BR      NXTBA

 023222  120127  000122         2$:     CMPB    R1,#'R
 023226  001014                         BNE     3$
 023230  121027  000104                 CMPB    (R0),#'D
 023234  001003                         BNE     8$
 023236  012702  000005                 MOV     #5,R2
 023242  000475                         BR      NXTBA
 023244  121027  000103         8$:     CMPB    (R0),#'C
 023250  001316                         BNE     FUNBEG
 023252  012702  000013                 MOV     #13,R2
 023256  000501                         BR      NXTDA

 023260  120127  000104         3$:     CMPB    R1,#'D
 023264  001006                         BNE     4$
 023266  121027  000122                 CMPB    (R0),#'R
 023272  001305                         BNE     FUNBEG
 023274  012702  000015                 MOV     #15,R2
 023300  000470                         BR      NXTDA

 023302  120127  000103         4$:     CMPB    R1,#'C
 023306  001006                         BNE     5$
 023310  121027  000122                 CMPB    (R0),#'R
 023314  001274                         BNE     FUNBEG
 023316  012702  000001                 MOV     #1,R2
 023322  000533                         BR      EXEC

 023324  120127  000123         5$:     CMPB    R1,#'S
 023330  001266                         BNE     FUNBEG
 023332  121027  000113                 CMPB    (R0),#'K
 023336  001263                         BNE     FUNBEG
 023340  012702  000011                 MOV     #11,R2

 023344                         6$:
 023344  104401  023352                 TYPE    ,67$            ;;TYPE ASCIZ STRING
 023350  000404                         BR      66$             ;;GET OVER THE ASCIZ
                        ;;67$:  .ASCIZ  /CYL1? /
```

```
 4903  023362                          66$:
 4904  023362  004737  023746                  JSR     PC,INPT
 4905  023366  000766                          BR      6$
 4906  023370  010004                          MOV     R0,R4
 4907
 4908  023372                          7$:
 4909  023372  104401  023400                  TYPE    ,69$            ;;TYPE ASCIZ STRING
 4910  023376  000404                          BR      68$             ;;GET OVER THE ASCIZ
 4911                                  ;;69$:  .ASCIZ  /CYL2? /
 4912  023410                          68$:
 4913  023410  004737  023746                  JSR     PC,INPT
 4914  023414  000766                          BR      7$
 4915  023416  010005                          MOV     R0,R5
 4916  023420  017700  155514                  MOV     @SWR,R0         ;GET DRIVE # FROM SW REG<15-13>
 4917  023424  042700  017777                  BIC     #17777,R0               ;CLR UNWANTED BITS
 4918  023430  050004                          BIS     R0,R4                   ;SET DRIVE # IN DSK ADRES
 4919  023432  050005                          BIS     R0,R5
 4920  023434  000466                          BR      EXEC
 4921
 4922
 4923  023436                          NXTBA:
 4924  023436  104401  023444                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 4925  023442  000404                          BR      64$             ;;GET OVER THE ASCIZ
 4926                                  ;;65$:  .ASCIZ  /RKBA? /
 4927  023454                          64$:
 4928  023454  104412                          RDOCT
 4929  023456  012637  001476                  MOV     (SP)+,INDX2
 4930
 4931  023462                          NXTDA:
 4932  023462  104401  023470                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 4933  023466  000404                          BR      64$             ;;GET OVER THE ASCIZ
 4934                                  ;;65$:  .ASCIZ  /RKDA? /
 4935  023500                          64$:
 4936  023500  104412                          RDOCT
 4937  023502  012600                          MOV     (SP)+,R0
 4938  023504  010001                          MOV     R0,R1
 4939  023506  006201                          ASR     R1
 4940  023510  006201                          ASR     R1
 4941  023512  006201                          ASR     R1
 4942  023514  006201                          ASR     R1
 4943  023516  006201                          ASR     R1
 4944  023520  042701  177400                  BIC     #177400,R1
 4945  023524  020127  000312                  CMP     R1,#312
 4946  023530  003354                          BGT     NXTDA
 4947  023532  010001                          MOV     R0,R1
 4948  023534  042701  177760                  BIC     #177760,R1
 4949  023540  020127  000013                  CMP     R1,#13
 4950  023544  003346                          BGT     NXTDA
 4951  023546  010003                          MOV     R0,R3
 4952  023550  022702  000015                  CMP     #15,R2
 4953  023554  001416                          BEQ     EXEC
 4954
 4955
 4956  023556                          NXTWC:
 4957  023556  104401  023564                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 4958  023562  000405                          BR      64$             ;;GET OVER THE ASCIZ
```



```
 4959                                  ;;65$:  .ASCIZ  /#WORDS? /
 4960  023576                          64$:
 4961  023576  104412                          RDOCT
 4962  023600  005416                          NEG     (SP)
 4963  023602  012637  001502                  MOV     (SP)+,INDX4
 4964  023606  000401                          BR      EXEC
 4965
 4966  023610  104415                  EXEC1:  CON,RESET                       ;CLR EROR, CONTROL RESET
 4967  023612  022702  000011          EXEC:   CMP     #11,R2                  ;SEEK FUNCTION?
 4968  023616  001005                          BNE     2$                      ;NO
 4969  023620  020403                          CMP     R4,R3                   ;IF SEEK, INSERT THE RIGHT
 4970  023622  001402                          BEQ     3$              ;CYLINDER ADDRESS
 4971  023624  010403                          MOV     R4,R3
 4972  023626  000401                          BR      2$
 4973  023630  010503                  3$:     MOV     R5,R3
 4974  023632  013777  001476  155622  2$:     MOV     INDX2,@RKBA
 4975  023640  010377  155620                  MOV     R3,@RKDA
 4976  023644  013777  001502  155606          MOV     INDX4,@RKWC
 4977  023652  010277  155600                  MOV     R2,@RKCS
 4978  023656  105777  155574          1$:     TSTB    @RKCS           ;WAIT FOR CNTROL RDY
 4979  023662  100375                          BPL     1$
 4980  023664  022702  000001                  CMP     #1,R2           ;IF IT'S CON RESET FUNCTION
 4981  023670  001401                          BEQ     4$              ;DONT WAIT FOR R/W/S RDY
 4982  023672  104423                          RESDON                  ;R/W/S RDY?
 4983
 4984  023674  032777  140000  155554  4$:     BIT     #140000,@RKCS   ;ERROR?
 4985  023702  001006                          BNE     FUNERR          ;YES
 4986  023704  032777  000010  155226  CHSW:   BIT     #SW3,@SWR       ;TERMINATE THIS FUNCTION OR REPEAT?
 4987  023712  001737                          BEQ     EXEC            ;REPEAT
 4988  023714  000137  023106                  JMP     FUNBEG          ;TERMINATE
 4989
 4990
 4991  023720  012737  023610  001110  FUNERR: MOV     #EXEC1,#LPERR   ;SET UP FOR LUPING
 4992  023726  012737  023610  001106          MOV     #EXEC1,#LPADR
 4993  023734  004737  016106                  JSR     PC,GT4RG
 4994  023740  104030                          ERROR   30              ;REPORT ERROR
 4995  023742  104415                          CON,RESET               ;CLR ERROR
 4996  023744  000757                          BR      CHSW
 4997
 4998  023746  104412                  INPT:   RDOCT
 4999  023750  012600                          MOV     (SP)+,R0
 5000  023752  020027  000312                  CMP     R0,#312
 5001  023756  003007                          BGT     1$
 5002  023760  006300                          ASL     R0
 5003  023762  006300                          ASL     R0
 5004  023764  006300                          ASL     R0
 5005  023766  006300                          ASL     R0
 5006  023770  006300                          ASL     R0
 5007  023772  062716  000002                  ADD     #2,(SP)
 5008  023776  000207                  1$:     RTS     PC
 5009
 5010  024000  032777  000001  155132  CHKSW0: BIT     #SW0,@SWR       ;WRITE A PATTERN GIVEN BY USER?
 5011  024006  001416                          BEQ     1$              ;NO
 5012                                                                  ;YES, ASK FOR PATTERN
 5013  024010  104401  024016                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 5014  024014  000404                          BR      64$             ;;GET OVER THE ASCIZ
```

```
                                        ;;65$:  .ASCIZ  /PATRN?/
                                        64$:
                                                RDOCT
                                                MOV     (SP)+,IOBUF1    ;SAVE THE PATTERN
                                                MOV     #IOBUF1,INDX2
                                                RTS     PC
                                        1$:     ADD     #6,(SP)         ;SKIP NXT 2 INST ON RETURN
                                                RTS     PC

                                        FCHECK: DRV.RESET
                                                CON.RESET
                                                MOV     DRIVAD,DRHOLD           ;SAVE DRIVE ADRR
                                                BIT     #20000,DRIVAD           ;SEE IF ODD
                                                BEQ     1$
                                                BIC     #20000,DRIVAD           ;MAKE EVEN
                                                BR      2$
                                        1$:     BIS     #20000,DRIVAD           ;MAKE ODD
                                        2$:     MOV     DRIVAD,@RKDA    ;DRIVE ADDR
                                                MOV     #11,@RKCS               ;DRIVE SEEK
                                                CON.RDY
                                                MOV     DRHOLD,@RKDA            ;OTHER DRIVE
                                                CON.RDY
                                                BIT     #100,@RKDS              ;HEAEDS IN MOTIONN?
                                                BNE     3$                      ;NO SO RK-05J
                                                TST     (R5)+                   ;YES RK-05F
                                        3$:     MOV     DRHOLD,DRIVAD           ;RESTORE ADDR
                                                DRV.RESET
                                                RTS     R5
                                        SIZEF:  CLR     DRIVAD          ;START AT DR0
                                                MOV     #DRIV0,R0               ;TABLE OF AVAIL DRIVES
                                        4$:     TSTB    (R0)                    ;THIS DRIVE HERE?
                                                BEQ     2$                      ;NO
                                                TSTB    2(R0)                   ;COMPLEMENT HERE?
                                                BEQ     2$                      ;NO
                                                JSR     R5,FCHECK               ;SEE IF F MODEL
                                                BR      2$                      ;J MODEL
                                                BIS     #2,(R0)         ;SET SIGN FOR F
                                                BIS     #2,2(R0)                ;BOTH DRIVES
                                        2$:     TST     (R0)+
                                                TST     (R0)+                   ;NEXT PAIR OF DRIVES
                                                ADD     #40000,DRIVAD           ;NEXT ACTUL ADDR
                                                CMP     #DRIV7+1,R0             ;CHECKED ALL?
                                                BGT     4$                      ;NOT YET
                                                RTS     PC

                                        ;ERROR MESSAGES


024250  047103  051124  020114  EM1:    .ASCIZ  /CNTRL RDY DIDN'T SET AFTR SEEK/
024256  042122  020131  044504
024264  047104  052047  051440
024272  052105  040440  052106
024300  020122  042523  045505
024306     000

024307     123  047111  047440  EM2:    .ASCIZ  /SIN ON SEEK/
```



```
024314  020116  042523  045505
024322     000

024323     104  042522  047440  EM3:    .ASCIZ  /DRE ON SEEK/
024330  020116  042523  045505
024336     000

024337     047  051105  023522  EM4:    .ASCIZ  /'ERR' ON SEEK/
024344  047440  020116  042523
024352  045505     000

024355     104  052522  047440  EM5:    .ASCIZ  /DRU ON SEEK, PUT DRV ON 'LOAD' & 'RUN'/
024362  020116  042523  045505
024370  020054  052520  020124
024376  051104  020126  047117
024404  023440  047514  042101
024412  020047  020046  051047
024420  047125  000047

024424  027522  027527  020123  EM6:    .ASCIZ  "R/W/S RDY NOT SET AFTER SEEK"
024432  042122  020131  047516
024440  020124  042523  020124
024446  043101  042524  020122
024454  042523  045505     000

024461     123  047111  047440  EM7:    .ASCIZ  /SIN ON WRT FMT/
024466  020116  051127  020124
024474  046506  000124

024500  042447  051122  020047  EM10:   .ASCIZ  /'ERR' ON DOING WRITE FMT/
024506  047117  042040  044517
024514  043516  053440  044522
024522  042524  043040  052115
024530     000

024531     123  047111  047440  EM11:   .ASCIZ  "SIN ON RD/FMT"
024536  020116  042122  043057
024544  052115     000

024547     047  051105  023522  EM12:   .ASCIZ  /'ERR' ON READ FMT/
024554  047440  020116  042522
024562  042101  043040  052115
024570     000

024571     127  047522  043516  EM13:   .ASCIZ  /WRONG HDRS FROM 'SEC#'/
024576  044040  051104  020123
024604  051106  046517  023440
024612  042523  021503  000047

024620  051105  051117  047440  EM14:   .ASCIZ  /EROR ON IMPLIED SEEK FROM 'CYLA' TO 'CYLB'/
024626  020116  046511  046120
024634  042511  020104  042523
024642  045505  043040  047522
024650  020115  041447  046131
024656  023501  052040  020117
024664  041447  046131  023502
024672     000

024673     122  040505  020104  MS15:   .ASCIZ  /READ WRONG HDRS FROM 'CYLB' ABOVE/
```

```
5127  024700  051127  047117  020107
5128  024706  042110  051522  043040
5129  024714  047522  020115  041447
5130  024722  046131  023502  040440
5131  024730  047502  042526     000
5132
5133  024735     122  040505  020104  EM16:   .ASCIZ  /READ WRONG, 1ST WRD FROM SEC 0, 'CYLB' (ON IMPLIED SEEK FROM 'CYLA')/
5134  024742  051127  047117  026107
5135  024750  030440  052123  053440
5136  024756  042122  043040  047522
5137  024764  020115  042523  020103
5138  024772  026060  023440  054503
5139  025000  041114  020047  047450
5140  025006  020116  046511  046120
5141  025014  042511  020104  042523
5142  025022  045505  043040  047522
5143  025030  020115  041447  046131
5144  025036  023501  000051
5145
5146  025042  042522  042101  030440  MS17:   .ASCIZ  /READ 1ST WRD FROM SEC 1, 'CYLB' ABOVE/
5147  025050  052123  053440  042122
5148  025056  043040  047522  020115
5149  025064  042523  020103  026061
5150  025072  023440  054503  041114
5151  025100  020047  041101  053117
5152  025106  000105
5153
5154  025110  042522  042101  053440  EM20:   .ASCIZ  /READ WRONG HDRS ON IMPLIED SEEK FROM 'CYLA' TO 'CYLB'/
5155  025116  047522  043516  044040
5156  025124  051104  020123  047117
5157  025132  044440  050115  044514
5158  025140  042105  051440  042505
5159  025146  020113  051106  046517
5160  025154  023440  054503  040514
5161  025162  020047  047524  023440
5162  025170  054503  041114  000047
5163
5164  025176  051105  051117  047440  EM21:   .ASCIZ  /EROR ON DOING WRITE ON DISK/
5165  025204  020116  047504  047111
5166  025212  020107  051127  052111
5167  025220  020105  047117  042040
5168  025226  051511  000113
5169
5170  025232  044523  020116  047117  EM22:   .ASCIZ  /SIN ON DOING WRITE/
5171  025240  042040  044517  043516
5172  025246  053440  044522  042524
5173  025254     000
5174
5175  025255     110  020105  047117  EM23:   .ASCIZ  /HE ON DOING READ/
5176  025262  042040  044517  043516
5177  025270  051040  040505  000104
5178
5179  025276  051503  020105  047117  EM24:   .ASCIZ  /CSE ON READ/
5180  025304  051040  040505  000104
5181
5182  025312  040504  040524  042440  EM25:   .ASCIZ  /DATA EROR ON READ FROM DISK ADRES/
```



```
5183  025320  047522  020122  047117
5184  025326  051040  040505  020104
5185  025334  051106  046517  042040
5186  025342  051511  020113  042101
5187  025350  042522  000123
5188
5189  025354  042510  047440  020116  EM26:   .ASCIZ  /HE ON WRT CHK/
5190  025362  051127  020124  044103
5191  025370  000113
5192
5193  025372  051127  020124  044103  EM27:   .ASCIZ  /WRT CHK EROR/
5194  025400  020113  051105  051117
5195  025406     000
5196
5197  025407     105  047522  000122  EM30:   .ASCIZ  /EROR/
5198
5199                                  ;DATA HEADERS
5200
5201
5202  025414  020040  041520  020040  DH1:    .ASCIZ  /  PC     RKCS    RKER    RKDS    RKDA/
5203  025422  020040  051040  041513
5204  025430  020123  020040  051040
5205  025436  042513  020122  020040
5206  025444  051040  042113  020123
5207  025452  020040  051040  042113
5208  025460  000101
5209
5210  025462  042523  045505  021440  DH6:    .ASCIZ  /SEEK #  SEEKING  ERRORS/
5211  025470  020040  042523  045505
5212  025476  047111  020107  042440
5213  025504  051122  051117  000123
5214
5215  025512  020040  041520  020040  DH7:    .ASCIZ  /  PC     RKCS    RKER    RKDS    RKDA    CYL/
5216  025520  020040  051040  041513
5217  025526  020123  020040  051040
5218  025534  042513  020122  020040
5219  025542  051040  042113  020123
5220  025550  020040  051040  042113
5221  025556  020101  020040  041440
5222  025564  046131     000
5223
5224  025567     040  050040  020103  DH11:   .ASCIZI/  PC    RKCS    RKER    RKDS   RKDA: DR  CYL     SUR     SEC/
5225  025574  020040  051040  041513
5226  025602  020123  020040  051040
5227  025610  042513  020122  020040
5228  025616  051040  042113  020123
5229  025624  020040  045522  040504
5230  025632  020072  051104  020040
5231  025640  054503  020114  020040
5232  025646  020040  052523  020122
5233  025654  020040  020040  042523
5234  025662  000103
5235  025664  042523  021503  020040  DH13:   .ASCIZ  /SEC#  HDR RCVD/
5236  025672  042110  020122  041522
5237  025700  042126     000
5238
```

```
5239  025703  040  050040  020103  DH14:  .ASCIZ  /  PC     CYLA    CYLB     RKER   RKDS     TRY#/
5240  025710  020040  020040  054503
5241  025716  040514  020040  020040
5242  025724  054503  041114  020040
5243  025732  020040  051040  042513
5244  025740  020122  020040  045522
5245  025746  051504  020040  020040
5246  025754  052040  054522  000043
5247
5248  025762  020040  041520  020040  DH16:  .ASCIZ  /  PC     CYLA   CYLB    EXPCT   RECVD   TRY#/
5249  025770  020040  041440  046131
5250  025776  020101  020040  054503
5251  026004  041114  020040  020040
5252  026012  054105  041520  020124
5253  026020  020040  042522  053103
5254  026026  020104  020040  051124
5255  026034  021531     000
5256
5257  026037     040  050040  020103  DH17:  .ASCIZ  /  PC     CYLB   EXPCT   RECVD/
5258  026044  020040  020040  054503
5259  026052  041114  020040  042440
5260  026060  050130  052103  020040
5261  026066  051040  041505  042126
5262  026074     000
5263
5264  026073     040  050040  020103  DH24:  .ASCIZ/  PC     TRY#   RKCS    RKER    RKDS  RKDA: DR  CYL     SUR      SEC/
5265  026102  020040  020040  051124
5266  026110  021531  020040  051040
5267  026116  041513  020123  020040
5268  026124  051040  042513  020122
5269  026132  020040  051040  042113
5270  026140  020123  051040  042113
5271  026146  035101  042040  020122
5272  026154  041440  046131  020040
5273  026162  020040  051440  051125
5274  026170  020040  020040  020040
5275  026176  042523  000103
5276
5277  026202  047527  042122  020043  DH25:  .ASCIZ  /WORD#  EXPCT   RECVD   CYL  SUR  SEC/
5278  026210  042440  050130  052103
5279  026216  020040  051040  041505
5280  026224  042126  020040  041440
5281  026232  046131  020040  052523
5282  026240  020122  051440  041505
5283  026246     000
5284
5285
5286                                  ;ERROR DATA POINTERS
5287
5288
5289          026250                  .EVEN
5290
5291  026250  001116  001162  001164  DT1:   .WORD   $ERRPC,$REG0,$REG1,$REG2,$REG3,0
5292  026256  001166  001170  000000
5293
5294  026264  001116  001162  001164  DT7:   .WORD   $ERRPC,$REG0,$REG1,$REG2,$REG3,$REG4,0
```



```
5295  026272  001166  001170  001172
5296  026300  000000
5297
5298  026302  001116  001162  001164  DT11:  .WORD   $ERRPC,$REG0,$REG1,$REG2,$REG4,$REG5,$REG6,$REG7,0
5299  026310  001166  001172  001174
5300  026316  001176  001200  000000
5301
5302  026324  001116  001162  001164  DT17:  .WORD   $ERRPC,$REG0,$REG1,$REG2,0
5303  026332  001166  000000
5304
5305  026336  001116  001202  001162  DT24:  .WORD   $ERRPC,$REG10,$REG0,$REG1,$REG2,$REG4,$REG5,$REG6,$REG7,0
5306  026344  001164  001166  001172
5307  026352  001174  001176  001200
5308  026360  000000
5309
5310                                  ;DATA BUFFERS
5311
5312  026362                          IOBUF0:
5313  026362  000030                  OUTBUF: .BLKW   24.      ;IOBUF0 AND IOBUF1 ARE
5314  026442  000030                  BUFR4:  .BLKW   24.      ;TWO - 768 WORDS LONG BUFFERS
5315  026522  000030                  BUFR5:  .BLKW   24.      ;NORMALLY USED FOR DATA TRANSFERS
5316  026602  000030                  BUFR6:  .BLKW   24.      ;TO AND FROM DISK
5317  026662  000030                  BUFR7:  .BLKW   24.
5318  026742  000200                  BUFR10: .BLKW   128.
5319  027342  000200                  BUFR11: .BLKW   128.
5320  027742  000200                  BUFR12: .BLKW   128.
5321  030342  000200                  BUFR13: .BLKW   128.
5322  030742  000210                  BUFR14: .BLKW   136.
5323
5324  031362  001400                  IOBUF1: .BLKW   768.
5325
5326
5327
5328          000001                          .END
```

```
ABRT    016716          BUFR13  030342          DRVDON  001223          EXT10   012656          NOSIN   010752
ADRES   001450          BUFR14  030742          DRVPTR  001226          EXT4    006540          NXTBA   023436
BADTMO  002422          BUFR2   001352          DRV.RE= 104416          EXT5    007276          NXTDA   023462
BEGIN   010456          BUFR4   026442          DR.RST  016414          EXT6    010356          NXTDRV  003704
BEGSK   013174          BUFR5   026522          DSKADR  001432          EXT7    012114          NXTPAT  001426
BIT0  = 000001          BUFR6   026602          DSWR  = 177570          FCHECK  024052          NXTWC   023556
BIT00 = 000001          BUFR7   026662          DT1     026250          FDRIVE  002114          OUTADR  001262
BIT01 = 000002          BUSADR  001434          DT11    026302          FDRVE1  002116          OUTBUF  026362
BIT02 = 000004          CHKHDR  007204          DT17    026324          FFUNC   001216          PAT0    001414
BIT03 = 000010          CHKSW0  024000          DT24    026336          FINISH  011436          PAT1    001416
BIT04 = 000020          CHSW    023704          DT7     026264          FUNBEG  023106          PAT2    001420
BIT05 = 000040          CKSWR = 104407          EMTVEC= 000030          FUNERR  023720          PAT3    001422
BIT06 = 000100          CLRERR  012106          EM1     024250          GCYL    016360          PBUF0   001404
BIT07 = 000200          CMPAGA  011170          EM10    024500          GETBUF  007770          PBUF1   001406
BIT08 = 000400          CMPGRP  014546          EM11    024531          GETINF  016140          PGSUBR  001430
BIT09 = 001000          CN.RDY  016544          EM12    024547          GOBAK   011574          PIRQ  = 177772
BIT1  = 000002          CN.RST  016526          EM13    024571          GOTYPE  013524          PIRQVE= 000240
BIT10 = 002000          COMPAR  011144          EM14    024620          GTSWR = 104406          PUSH    014146
BIT11 = 004000          CON.RD= 104421          EM16    024735          GT4RG   016106          PLTGRP  014032
BIT12 = 010000          CON.RE= 104415          EM2     024307          GT5RG   016062          PLTPT   014662
BIT13 = 020000          CR    = 000015          EM20    025110          HT    = 000011          PLT1    014362
BIT14 = 040000          CRLF  = 000200          EM21    025176          INADR   001260          PRSPAT  001424
BIT15 = 100000          CSEROR  011600          EM22    025232          INDX1   001474          PR0   = 000000
BIT2  = 000004          DDI&P = 177570          EM23    025255          INDX2   001476          PR1   = 000040
BIT3  = 000010          DH1     025414          EM24    025276          INDX3   001500          PR2   = 000100
BIT4  = 000020          DH11    025567          EM25    025312          INDX4   001502          PR3   = 000140
BIT5  = 000040          DH13    025664          EM26    025354          INPT    023746          PR4   = 000200
BIT6  = 000100          DH14    025703          EM27    025372          IOBUF0  026362          PR5   = 000240
BIT7  = 000200          DH16    025762          EM3     024323          IOBUF1  031362          PR6   = 000300
BIT8  = 000400          DH17    026037          EM30    025407          IOTVEC= 000020          PR7   = 000340
BIT9  = 001000          DH24    026075          EM4     024337          LF    = 000012          PS    = 177776
BLNKS1  002111          DH25    026202          EM5     024355          LUPHE   010564          PSW   = 177776
BLNKS2  002110          DH6     025462          EM6     024424          LUPSIN  010556          PTGEN0  010032
BLNKS3  002107          DH7     025512          EM7     024461          LUPSW   001222          PTGEN1  010114
BLNKS4  002106          DISPLA  001142          ERCNT1  001504          LUPWCE  011274          PTGEN2  010216
BLNKS5  002105          DISPRE  000174          ERCNT2  001506          MESAGE= 104417          PTGEN3  010260
BLNKS6  002104          DMPREG  017576          ERCNT3  001510          MISCMP  011742          PTRN01  010110
BLNKS7  002103          DONE    011562          ERCNT4  001512          MSGE    015770          PTRN02  010112
BLNKS8  002102          DONTRK  011522          ERCNT5  001514          MSG1    001602          PT1     001550
BLNKS9  002101          DOSUR1  011554          ERCNT6  001516          MSG10   001742          PT10    001576
BLNK10  002100          DOWCHK  012344          ERCNT7  001520          MSG11   001771          PT11    001600
BLNK13  002075          DOWRT   012150          ERCNT8  001522          MSG12   002027          PT2     001562
BPTVEC= 000014          DRHOLD  002120          ERRTYP  017362          MSG13   002053          PT3     001564
BRKDA   016144          DRIVAD  001230          ERRVEC= 000004          MSG14   002064          PT4     001566
BTEOP   015156          DRIVS   001224          ERR1    016322          MSG2    001610          PT5     001570
BUFLG0  001410          DRIV0   001232          ERR2    016244          MSG3    001616          PT6     001572
BUFLG1  001412          DRIV1   001234          ERWCHK  011330          MSG4    001640          PT7     001574
BUFN0   001446          DRIV2   001236          ESR13   015402          MSG5    001657          PWRFL   004170
BUFR    001266          DRIV3   001240          ESR15   015310          MSG6    001716          PWRVEC= 000024
BUFR1   001320          DRIV4   001242          ESR20   015456          MSG7    001733          RDAGAI  010566
BUFR10  026742          DRIV5   001244          ESR25   015544          MS15    024673          RDCHR = 104410
BUFR11  027342          DRIV6   001246          EXEC    023612          MS17    025042          RDLIN = 104411
BUFR12  027742          DRIV7   001250          EXEC1   023610          NOHE    010704          RDOCT = 104412
```



```
READ    010532          ST2     003436          TYPDES  022222          $EOP    015172          $REG4   001172
REPEAT  012612          ST3     003666          TYPDS = 104405          $EOPCT  015214          $REG5   001174
REPMSC  011220          SWR     001140          TYPDSS= 104424          $ERFLG  001103          $REG6   001176
REPTIM  013124          SWREG   000176          TYPE  = 104401          $ERMAX  001115          $REG7   001200
RESDON= 104423          SW0   = 000001          TYPMSG= 104420          $ERROR  017106          $RESRE  022364
RESREG= 104414          SW00  = 000001          TYPOC = 104402          $ERRPC  001116          $RTNAD  015266
RESVEC= 000010          SW01  = 000002          TYPON = 104404          $ERRTB  002122          $SAVRE  022326
RES.DO  016440          SW02  = 000004          TYPOS = 104403          $ERTTL  001112          $SAVR6  023074
RETRY1  001252          SW03  = 000010          TYPTIM  013662          $ESCAP  001204          $SCOPE  016732
RETRY2  001254          SW04  = 000020          TY.MSG  016032          $FILLC  001156          $SETUP= 000117
RETRY3  001256          SW05  = 000040          WBUSAD  001442          $FILLS  001155          $STUP = 177777
RKBA    001462          SW06  = 000100          WCAGAI  011056          $GDADR  001120          $SUPRS  022266
RKCS    001456          SW07  = 000200          WCEROR  012006          $GDDAT  001124          $SVLAD  017044
RKDA    001464          SW08  = 000400          WCERR   012366          $GET42  015244          $SVPC = 000220
RKDB    001466          SW09  = 001000          WCHI    012372          $GTSWR  021670          $SWR  = 163000
RKDS    001452          SW1   = 000002          WCHI1   012364          $HD   = 000000          $SWRMK= 000000
RKER    001454          SW10  = 002000          WCLO    012572          $HIOCT  021772          $TKB    001146
RKPRI   001470          SW11  = 004000          WCREPT  011260          $ICNT   001104          $TKCNT  020526
RKVEC   001472          SW12  = 010000          WDSKAD  001440          $ILLUP  023070          $TKINT  020536
RKWC    001460          SW13  = 020000          WRDCNT  001436          $INTAG  001135          $TKQEN= 020535
R6    =%000006          SW14  = 040000          WRERR   012164          $ITEMB  001114          $TKQIN  020530
R7    =%000007          SW15  = 100000          WRHI    012324          $LF     001214          $TKQOU  020532
SAVREG= 104413          SW2   = 000004          WRLO    012166          $LPADR  001106          $TKQSR  020534
SBR1    006452          SW3   = 000010          WRLO1   012162          $LPERR  001110          $TKS    001144
SBR2    006476          SW4   = 000020          WRTCHK  011042          $MNEW   021661          $TKSRV  020606
SBR3    007150          SW5   = 000040          WWRDCN  001444          $M&WR   021650          $TN   = 000013
SECPTR  010352          SW6   = 000100          XHE     011366          $MUL  = 000003          $TNPWR  022532
SFTCMP  011112          SW7   = 000200          XXDPMD  001220          $MULT   020414          $TPB    001152
SIAD    001542          SW8   = 000400          $AUTOB  001134          $NULL   001154          $TPFLG  001157
SIZEF   024164          SW9   = 001000          $BDADR  001122          $NWTST= 000001          $TPS    001150
SKDON   015076          TBITVE= 000014          $BDDAT  001126          $OCNT   022216          $TRAP   022616
SMGRP   014416          TIMDON  013774          $BELL   001206          $OMODE  022220          $TRAP2  022640
SOAD    001524          TIMER   001264          $CHARC  020410          $OVER   017072          $TRP  = 000025
SORT    013266          TIMSEK  014722          $CKSWR  021000          $PASS   001100          $TRPAD  022652
SP1     012626          TKVEC = 000060          $CNTAG  001100          $POWER  023076          $TSTNM  001102
SP10    012650          TPVEC = 000064          $CM1  = 000011          $PWRAD  023064          $TTYIN  021626
SP11    012652          TRAPVE= 000034          $CM2  = 000022          $PWRDN  022724          $TYPDS  017750
SP12    012654          TRTVEC= 000014          $CM3  = 000011          $PWRMG  023060          $TYPE   020174
SP2     012630          TSTRWS  016642          $CNTLG  021643          $PWRUP  022776          $TYPEC  020344
SP3     012632          TST.RW= 104422          $CNTLC  021636          $QUES   001212          $TYPEX  020412
SP4     012634          TST1    004204          $CRLF   001213          $RDCHR  021302          $TYPOC  022020
SP5     012636          TST10   012120          $DBLK   020164          $RDLIN  021372          $TYPON  022034
SP6     012640          TST11   012662          $DB2D   022422          $RDOCT  021672          $TYPOS  021774
SP7     012642          TST12   015150          $DECVL  022602          $RDSZ = 000010          $XTSTR  016754
SP8     012644          TST2    004550          $DOAGN  015264          $REGAD  001160          $&GET4= 000000
SP9     012646          TST3    005046          $DTBL   020154          $REG0   001162          $0FILL  022217
STACK = 001100          TST4    005536          $ENDAD  015254          $REG1   001164          .     = 034362
START   002462          TST5    006544          $ENDCT  015222          $REG10  001202
START1  003020          TST6    007302          $ENDMG  015273          $REG2   001166
STKLMT= 177774          TST7    010362          $ENULL  015270          $REG3   001170

. ABS.  034362     000
```

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

```
DZRKLD,DZRKLD/LI:ME/NL:MC:MD:CND/SOL/MSQ_DZRKLD.P11
RUN-TIME:  62 43 1 SECONDS
RUN-TIME RATIO: 237/108=2.1
CORE USED:  27K  (53 PAGES)
```