

RK11/05F/J

BASIC LOGIC TESTS
CZRKKF0

AH-9246F-MC

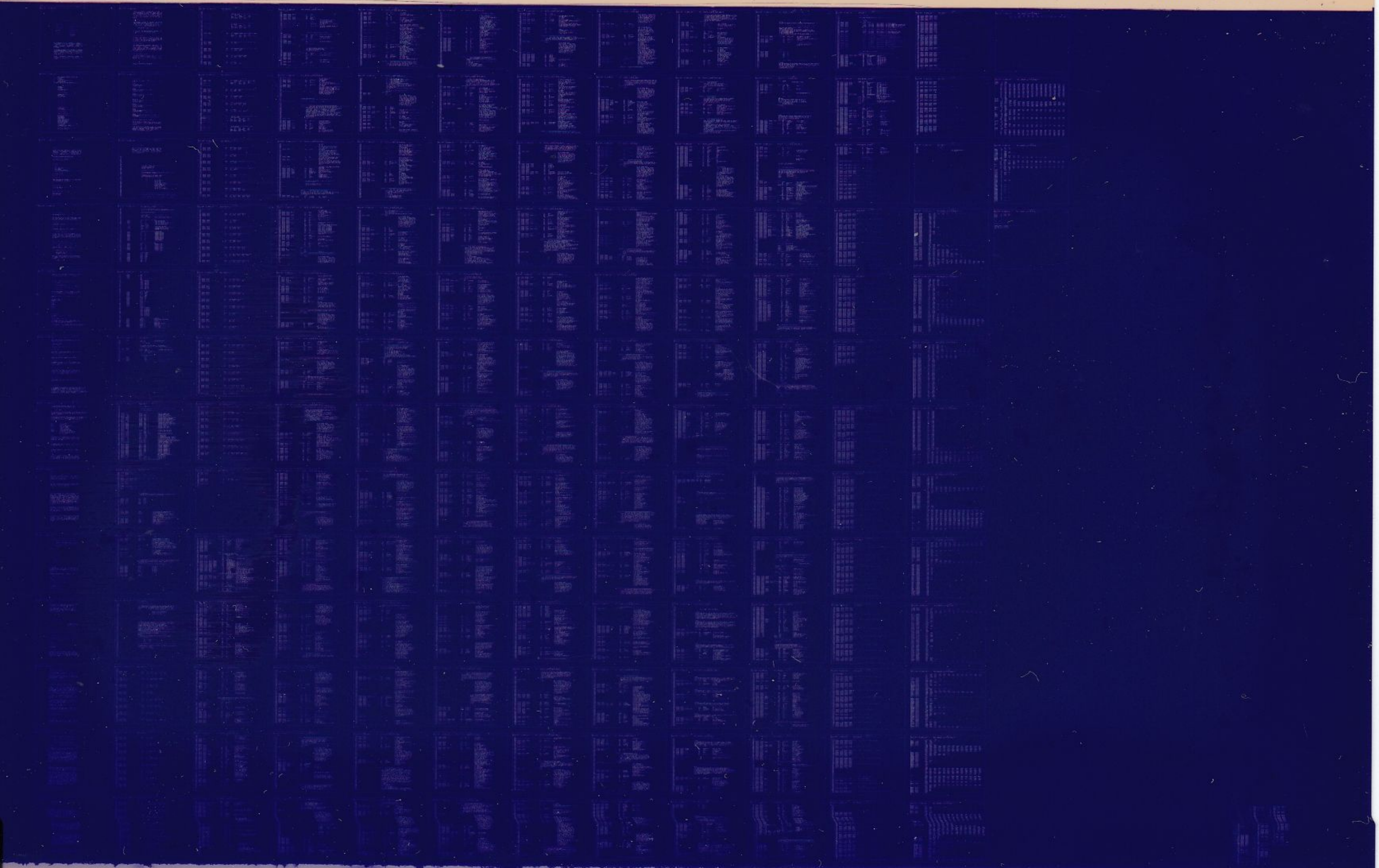
COPYRIGHT © 75-78

FICHE 1 OF 1

JUL 1978

digital

MADE IN USA



.REM %

IDENTIFICATION

PRODUCT CODE: AC-9244F-MC
PRODUCT NAME: CZRKKFO RK11 BASIC LOGIC TEST 2
DATE CREATED: JUNE 1978
MAINTAINER: DIAGNOSTIC GROUP
AUTHOR: JIM KAPADIA
REVISED BY: PERVEZ ZAKI
TOM SAWYER
CHUCK HESS

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

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

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

COPYRIGHT (C) 1975, 1978 BY DIGITAL EQUIPMENT CORPORATION

QUICK LOOK-UP OPERATING INSTRUCTIONS
FOR A QUICK REFERENCE, LOOK UP THE FOLLOWING SECTIONS:
1.0 ABSTRACT
2.0 REQUIREMENTS
4.1 LOADING AND OPERATOR ACTION
7.0 SWITCH OPTIONS
FOR A MORE COMPLETE EXPLANATION REFER TO THE TABLE OF
CONTENTS BELOW AND THE FOLLOWING DOCUMENT.

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	EQUIPMENT
2.2	PRELIMINARY PROGRAMS
2.3	EXECUTION TIME
3.0	STARTING ADDRESS
4.0	PROGRAM CONTROL MODES & OPERATOR ACTION
4.1	PAPER TAPE
4.2	RKDP DUMP MODE
4.3	RKDP CHAIN MODE
4.4	ACT11
5.0	DRIVE SELECTION
6.0	DRIVE-LESS TEST
7.0	SWITCH OPTIONS
8.0	SCOPE LOOPS
9.0	PROGRAM STRUCTURE
9.1	SET-UP PHASE
9.2	DRIVE DEPENDENT CONTROLLER TESTS
10.0	ERROR REPORTING
11.0	ERROR INTERPRETATION
12.0	HANDLERS AND COMMON ROUTINES
12.1	TRAP HANDLER
12.2	SCOPE HANDLER
12.3	ERROR HANDLER
12.4	CONTROL RESET ROUTINE
12.5	CONTROL READY ROUTINE
12.6	DRIVE RESET ROUTINE
12.7	TIME DELAY ROUTINE
12.8	WAIT FOR INTERRUPT ROUTINE
12.9	OTHER ROUTINES
	TTY HANDLER (I/O), ERROR TYPEOUT ROUTINE
	POWER DOWN/POWER UP ROUTINE
13.0	UNEXPECTED TIMEOUTS & RK11 INTERRUPTS
14.0	QUICK VERIFYING MODE

1.0 ABSTRACT

THE RK11 LOGIC TESTS CONSIST OF A SERIES OF TESTS AIMED AT CHECKING THE BASIC LOGIC OF THE RK11 CONTROLLER. THIS PROGRAM IS THE SECOND PART OF THE TWO-PART RK11 LOGIC TESTS. IT SHOULD BE NOTED THAT LOGIC TEST I AND LOGIC TEST II TOGETHER CONSTITUTE A COMPLETE PROGRAM AND BOTH OF THEM SHOULD BE RUN.

WHEN USED IN CONJUNCTION WITH A DRIVE IT IS CAPABLE OF DETECTING FAULTS IN THE DRIVE ALSO.

USED CORRECTLY THIS PROGRAM CAN BE AN EFFECTIVE ANALYTIC AND DIAGNOSTIC TOOL.

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 OR THE RK05 SIMULATOR (DRIVE TYPES MAY BE MIXED)

2.2 PRELIMINARY PROGRAMS

RK11 BASIC LOGIC TEST I (MD-11-DZRKJ)

2.3 EXECUTION TIME

ERROR FREE FIRST PASS ON PDP11/20 WITH CORE MEMORY TAKES APPROXIMATELY TWO MINUTES. CONSIDERABLY LESS FOR FASTER MACHINES OR MEMORIES.

3.0 STARTING ADDRESS

200 FOR ANY MODE OF OPERATION. NORMAL START UP WITH ALL SWITCHES DOWN.

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 .ABS 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 7.0) IF TESTING ON SIMULATOR PUT SW<10> UP.
- PRESS START.
- 4.1.5 THE PROGRAM IDENTIFIES ITSELF (NAME, MAINDEC NO), THEN THE FOLLOWING QUESTION IS ASKED:

DRIVES TO BE TESTED?

THE USER SHOULD TYPE IN THE DRIVE NUMBERS THAT ARE IN 'RUN' AND TO BE TESTED. CARRIAGE RETURN SHOULD TERMINATE THE STRING. IF AN RK-05F IS TO BE TESTED, TYPE THE SUFFIX 'F' WITH THE FIRST DRIVE OF THE PAIR. FOR EXAMPLE, IF DRIVES 2 AND 3 ARE ON AN RK-05F, TYPE ONLY 2F.

EXMP: DRIVES TO BE TESTED? 0,1,2<CR>

THE DRIVES DO NOT HAVE TO BE IN LOGICAL ORDER.

EXMP: DRIVES TO BE TESTED? 2,4<CR>

IF ANY ONE DRIVE IS TO BE TESTED, TYPE IN THAT NUMBER. IT DOES NOT HAVE TO BE DRIVE 0.

THUS A NORMAL SEQUENCE WITH DRIVES 0,1 WOULD BE:

RK11 BASIC LOGIC TEST 2
MAINDEC-11-CZRKKF
DRIVES TO BE TESTED? 0,1<CR>

- 4.1.6 THERE IS A 'RUBOUT' FEATURE WHICH ALLOWS RUBBING OUT ANY NUMBER OF CHARACTERS THAT WERE TYPED IN WRONG. THE RUBBED OUT CHARACTERS ARE ECHOED BACK WITHIN SLASHES.

" U " DELETES THE ENTIRE LINE

- 4.1.7 IF REPLY TO ANY OF THE ABOVE QUESTION IS IN A WRONG FORMAT (EX: 012<CR>;0,8<CR>; 0,A<CR>; M<CR> ETC), IT IS AUTOMATICALLY REJECTED, A "??" IS PRINTED OUT;

THE CORRECT ANSWER CAN NOW BE RETYPED AGAIN.

- 4.1.8 THE DRIVE NUMBER BEING TESTED OUT IS PRINTED:

DRIVE N ;N=0,1...7
IF THE DRIVE IS AN RK-05F, AN F IS APPENDED

AT THE END OF A PASS THE FOLLOWING TYPE-OUT OCCURS

END PASS # X

WHERE X= PASS NUMBER (1,2,3---), CONTROL IS PASSED TO THE BEGINNING OF THE PROGRAM AND RE-EXECUTION BEGINS. NO QUESTIONS ARE TO BE ANSWERED AGAIN.

- 4.1.9 ERROR FREE PASSES OF THE PROGRAM APPEAR AS SHOWN BELOW.

```
RK11 BASIC LOGIC TEST 2
MAINDEC-11-CZRKKF
DRIVES TO BE TESTED?
0,1<CR>
DRIVE 0
DRIVE 1
END PASS # 1
0
DRIVE 1
END PASS # 2
...
...
```

4.2 RKDP DUMP MODE

- 4.2.1 THE PROGRAM IS LOADED INTO THE MEMORY BY THE RKDP MONITOR

- 4.2.2 START AS NORMALLY USING SA 200

- 4.2.3 THE PROGRAM IDENTIFIES ITSELF (NAME,MAINDEC NO.). ON FINDING OUT THAT THE LOADING WAS BY RKDP (DUMP MODE), THE FOLLOWING MESSAGE APPEARS:

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

IF DRIVE 'N' IS TO BE TESTED, THE RKDP PACK ON THAT

DRIVE SHOULD BE REPLACED BY ANOTHER PACK, THE DRIVE SHOULD BE PUT ON 'WRT ENABL' (BECAUSE RKDP WRITE PROTECTS THE DRIVE).

IF DRIVE 'N' IS NOT TO BE CHECKED, THEN THE MESSAGE SHOULD BE IGNORED.

AFTER THIS, THE SEQUENCE OF QUESTIONING IS AS EXPLAINED IN SEC 4.1.5.

4.3 RKDP CHAIN MODE

THE PROGRAM IS CHAIN-LOADED FROM THE RKDP PACK ON DRIVE 'N'. AFTER THE PROGRAM IDENTIFIES ITSELF THE FOLLOWING PRINTOUT OCCURS.

'DRIVE 'N' NOT TESTED'

THERE IS NO OPERATOR INTERVENTION REQUIRED. THE PROGRAM FINDS OUT THE NUMBER OF DRIVES PRESENT.

4.4 ACT11 MODE

THE PROGRAM IS LOADED BY THE ACT11 MONITOR. ON STARTING, IDENTIFIES ITSELF, ASCERTAINS THE NUMBER OF DRIVES AND PROCEEDS WITH THE EXECUTION OF THE TESTS AS BEFORE.

5.0 DRIVE SELECTION

IF ANY PARTICULAR DRIVE IS TO BE SELECTED FOR TESTING, PUT THAT DRIVE ON 'RUN', 'WRITE ENABLE'; PUT REST OF THE DRIVES ON 'LOAD', 'WRITE LOCK' AND IN REPLY TO THE QUESTIONS (TO BE TESTED?) TYPE IN THE DRIVE NUMBER FOLLOWED BY CR. SEE SEC 4.1.5.

6.0 DRIVE-LESS TEST

USE RK11 BASIC LOGIC TEST I, WHICH IS ACTUALLY THE FIRST PART OF THE TWO-PART RK11 BASIC LOGIC TESTS. SEE SEC 1.0, 2.2.

7.0 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	INHIBIT ITERATIONS
SW<10>=1	TESTING ON SIMULATOR
SW<09>=1	LOOP ON SPECIFIC ERROR
SW<08>=1	LOOP ON TEST AS PER SW<07:00>
SW<06>=1	DROP THE DRIVE AFTER MAXIMUM ALLOWABLE NUMBER OF ERRORS OCCUR

7.1 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.

7.2 SW<14>

THE PROGRAM LOOPS ON THE SUBTEST THAT IS BEING EXECUTED WHEN THE SWITCH IS PUT ON. THIS SWITCH IS USED NORMALLY ALONG SW 15. SEE SEC 8.0.

7.3 SW <13>

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

7.4 SW <12>

THIS SWITCH ALLOWS THE PORGRAM TO CYCLE FROM THE POINT OF ERROR TO THE PREVIOUS SCOPE STATEMENT. NOTE THAT IN DOING SO ANY INITIALIZATION BEING DONE AT THE BEGINING OF THE SUBTEST WILL BE DONE AGAIN AND AGAIN. SEE SEC 8.0 FOR DIFFERENT SCOPE LOOPS

AVAILABLE.

7.5 SW <11>

EACH SUBTEST WILL BE EXECUTED ONLY ONCE. NORMALLY AFTER THE FIRST PASS, EACH SUBTEST IS ITERATED A NUMBER OF TIMES (USUALLY 50, 5 IN SOME CASES). SETTING THIS SWITCH INHIBITS ITERATIONS, SO THAT QUICK PASSES CAN BE MADE.

7.6 SW <10>

THIS SWITCH WHEN SET INDICATES THAT TESTING IS BEING DONE ON A SIMULATOR. THE SWITCH SHOULD BE PUT UP BEFORE STARTING THE PROGRAM. NOTE THAT RK11C IS NOT COMPATIBLE WITH THE SIMULATOR.

7.7 SW <09>

THIS SWITCH PROVIDES THE TIGHTEST POSSIBLE SCOPE LOOP. NOTE THAT THE 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

7.8 SW <08>

THIS SWITCH IS USED TO SELECT A PARTICULAR TEST (AS PER SW<00-07>) FOR EXECUTION AND SUBSEQUENT LOOPING. THUS IF TEST 15 IS TO BE SELECTED THE SWITCH SETTING WOULD BE 000415. IT SHOULD BE NOTED THAT BEFORE SELECTING TEST 15, ALL THE PREVIOUS TESTS (1-14) WILL BE EXECUTED.

7.9 SW<06>

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 5, AFTER 5 ERRORS HAVE OCCURED DRIVE IS DROPPED AND A MESSAGE (DRIVE # XXX DROPPED) IS PRINTED.

8.0 SCOPE LOOPS

THERE ARE THREE KINDS OF SCOPE LOOPS AVAILABLE

1. SW14: LOOPING IS DONE FOR THE ENTIRE SUB-TEST
2. SW12: LOOPING IS DONE FROM THE POINT OF ERROR BACK TO THE PREVIOUS 'SCOPE' STATEMENT.
3. SW09: PROVIDE THE TIGHTEST POSSIBLE SCOPE LOOP SEE SEC. 7.7

EXAMPLE:

TST1: SCOPE
:

INITIALIZATION
:
ERROR 1
:
ERROR 2
:
ERROR 3
:
ERROR 4
:
:
TST2: SCOPE

THE SEQUENCE OF LOOPING FOR DIFFERENT CASES IS EXPLAINED BELOW. NOTE THAT 'TST1' AND 'TST2' ARE TAGS WHICH DEFINE THE BOUNDARY OF A TEST, (IN THIS CASE TEST 1). TEST 1 STARTS AT 'TST1' AND ENDS JUST BEFORE 'TST2'.

IN THE ILLUSTRATION BELOW --> INDICATES THE POINT FROM WHERE RETURN IS MADE AND LOOPING IS DONE.

1. ERROR 2 OCCURS, SW 14 SET.

TST1..ERROR 2..TST2-->TST1..ERROR 2..TST2-->TST1...

2. ERROR 2 OCCURS, SW 12 SET.

TST1...ERROR 2-->TST1...ERROR2-->TST1...

3. ERROR 2,3; SW 14 SET.

TST1..ERROR 2..ERROR 3..TST2-->TST1..ERROR 2..ERROR 3..TST2-->TST1...

4. ERROR 2,3; SW 12 SET.

TST1...ERROR 2-->TST1...ERROR 2-->TST1....

NOTE THAT LOOPING IS DONE FROM THE VERY FIRST ERROR ENCOUNTERED. THE MORE BASIC AND EARLIER IT OCCURS AND IS DETECTED AND SHOULD BE FIXED.

IN THE ABOVE EXAMPLE NO PART OF THE SUB-TEST IS BEING REPEASING DIFFERENT PARAMETERS, HENCE IT SO HAPPENS THAT SW 9 AND 12 GIVE THE SAME KIND OF LOOPS. THE EXAMPLE BELOW WILL DEMONSTRATE THE DIFFERENCE BETWEEN SW 9 AND 12.

TST1: SCOPE
 :

INITIALIZATION

```

:
ERROR 1
:
MOV #1$, $LPERR ; '$LPERR' CONTAINS
; THE ADDRESS TO LOOP
; BACK ON ERROR- SW 9

```

```

1$: :
: I
: ER I N REPETITIONS
: I
TST2: SCOPE ----

```

1. SW 12 SET, ERROR 2 OCCURS DURING K.TH REPETITIONS

TST1..1,2...K.ERROR 2-->TST1..1,2...K.ERROR 2-->TST1..

2. SW 9 SET, ERROR 2 OCCURS DURING K.TH REPETITION

1\$..K..ERROR 2-->1\$..K..ERROR 2-->1\$...

9.0 PROGRAM STRUCTURE

THERE ARE THREE DISTINCT PARTS OF THE PROGRAM.

SET-UP PHASE
 DRIVE-DEPENDENT CONTROLLER TESTS

9.1 SET-UP PHASE

SETTING UP OF INITIAL POINTERS, VECTORS, TABLES IS DONE IN THIS PART. IN THIS SECTION THE DECISION IS MADE ABOUT THE PROGRAM MODE-PAPER TAPE, RKDP DUMP, CHAIN OR ACT11. IF IN A NON-INTERVENTION MODE (CHAIN, ACT11) NUMBER OF DRIVES AND THE TYPE OF CONTROLLER IS FOUND OUT. FLAGS ARE SET TO INDICATE

WHICH DRIVES ARE TO BE TESTED, ETC.

9.2 DRIVE DEPENDENT CONTROLLER TESTS

THIS SECTION FORMS A MAJOR PART OF THE PROGRAM WHEREIN MOST OF THE CONTROLLER IS CHECKED.

JUST BEFORE ENTERING THIS SECTION THE PROGRAM FINDS OUT WHICH DRIVE IS TO BE CHECKED. IF IN RKDP CHAIN MODE, DRIVE 'N' IF PRESENT, IS SKIPPED AND THE NEXT AVAILABLE DRIVE IS SELECTED.

THE DRIVE NUMBER BEING TESTED IS PRINTED OUT:

DRIVE N ;N=0,1,2...7

THE TESTING IS DONE IN A LOGICAL HIERCHY, SIMPLER THINGS FIRST, THEN MORE COMPLEX AND SO ON.

IN ONE OF THE TESTS THE ENTIRE DISK PACK IS FORMATTED, CHECKS ARE MADE FOR ERROR CONDITIONS. THE FIRST WORD OF EVERY SECTOR IS WRITTEN AS A PSUEDO-HEADER, REFLECTING THE ABSOLUTE ADDRESS OF THAT SECTOR (DRIVE #, CYLINDER #, SURFACE #, SECTOR #). EXAMPLE: THE PSUEDO-HEADER FOR SECTOR 5, SURFACE 0, CYLINDER 20, DRIVE 0 WOULD BE 001005.

IN THE NEXT TEST THE HEADERS FROM THE ENTIRE PACK ARE READ AND CHECKED FOR CORRECTNESS. IN A SUBSEQUENT TEST ALL THE PSUEDO-HEADERS ARE READ AND VERIFIED.

ALL THE FUNCTIONS ARE CHECKED OUT. 'SEEK' IS CHECKED IN THE THREE DIFFERENT VELOCITY MODES (HIGH, MEDIUM, LOW). VARIOUS ERRORS LIKE 'NXD', 'NXC', ETC. ARE SIMULATED AND CHECKED.

HARDWARE POGIC IS CHECKED USING ALL THE DRIVES THAT HAVE BEEN INDICATED.

AT THE END OF THIS SECTION, A CHECK IS MADE IF ALL INDICATED DRIVES HAVE BEEN TESTED. IF NOT, CONTROL IS TRANSFERRED TO THE BEGINNING OF THIS SECTION.

THUS ONE PASS OF THE PROGRAM INVOLVES DOING

1. SUBTEST #1 ONCE
2. DRIVE-DEPENDENT TESTS FOR ALL THE SELECTED DRIVES.

10.0 ERROR REPORTING

THE ERROR TABLE STARTING AT \$ERRTB CONTAINS INFORMATION PERTAINING TO EVERY ERROR THAT CAN OCCUR. EACH ITEM IN THE TABLE CONSISTS OF FOUR

ENTRIES.

- A. EM - THIS IS A POINTER TO THE ERROR MESSAGE TO BE TYPED OUT WHEN THE ERROR OCCURS.
- B. DH - THIS IS A POINTER TO THE DATA HEADER TO BE TYPED OUT.
- C. DT - THIS IS A POINTER TO THE DATA WHICH IS TO BE TYPED TYPED OUT UNDER THE HEADERS.
- D. 0 - THIS IS A TERMINATOR SIGNIFYING THE END OF THE ITEM.

THE ERROR CALL IS AN EMi INSTRUCTION WITH ITS LOWER BYTE ENCODED TO INDICATE THE ERROR NUMBER. THUS OR 1' WOULD BE (EMT+1) IE 104001.

EVERY ERROR CORRESPONDS TO AN ITEM IN THE ERROR TABLE. THUS "ERROR 14" WOULD CORRESPOND TO ITEM 14. AS FAR AS POSSIBLE, THE ERROR MESSAGES HAVE BEEN KEPT SHORT, BUT CLARITY IS NOT SACRIFICED FOR BREVITY. INSPITE OF THIS, IF THE USER FINDS A NEED, HE CAN LOOK UP THE ENTIRE ERROR MESSAGE IN THE ERROR ITEMS TABLE FOUND IN THE BEGINNING OF THE LISTINGS. THUS FOR "ERROR 14", "ITEM 14" IN THE ITEM TABLE CAN BE LOOKED UP. WHEN THE ERROR INSTRUCTION IS EXECUTED A TRAP OCCURS TO THE ERROR HA LOCATED AT \$ERROR WHICH PROCESSES THE ERROR CALL. SEE SEC 12.3

11.0 ERROR INTERPRETATION

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

EVERY ERROR MESSAGE CONTAINS A PC. THIS PC INDICATES THE POSITION IN PROGRAM WHERE THE ERROR CALL IS LOCATED. THE ERROR MESSAGE, BECAUSE OF PRACTICAL CONSIDERATIONS IS MADE SHORT AND MEANINGFUL. THE USER IS ADVTD 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 IN PINPOINTING THE FAULT. 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.

12.0 HANDLERS AND COMMON ROUTINES

THE COMPOSED ROUTINES USED IN THE PROGRAM ARE CALLED IN TWO WAYS.

- A. AS A SUBROUTINE THROUGH 'JSR' CALL
- B. THROUGH A 'TRAP' HANDLER

12.1 TRAP HANDLER

MANY COMMONLY USED ROUTINES IN THE PROGRAM ARE CALLED USING THE TRAP INSTRUCTION AND THE 'TRAP' HANDLER. THE LOWER BYTE OF THE TRAP INSTRUCTION IS ENCODED DIFFERENTLY FOR DIFFERENT ROUTINES. THE TRAP HANDLER IS LOCATED AT '\$TRAP'. WHEN A CALL FOR A ROUTINE IS EXECUTED, A TRAP OCCURS TO THE HANDLER 'ARAP'. THE HANDLER PICKS UP THE LOWER BYTE OF THE 'CALL INSTRUCTION' AND USES IT TO FORM THE STARTING ADDRESS OF THE ROUTINE TO GO TO FOR SERVICE.

12.2 SCOPE HANDLER

THE 'IOT' TRAP IS USED BY THE 'SCOPE' STATEMENT. WHEN 'SCOPE' IS EXECUTED, AN IOT TRAP OCCURS TO MEMORY LOCATION '\$SCOPE'. THE SCOPE HANDLER STARTS AT '\$SCOPE'. DEPENDING ON THE SWITCH SETTINGS THE HANDLER DECIDES TO LOOP ON TEXT, INHIBIT ITERATIONS ETC. THERE ARE CERTAIN POINTERS AND FLAGS WHICH ARE ADJUSTED. THUS, IT IS NOT ADVISABLE START THE PROGRAM AT ANY GIVEN LOCATION SINCE THE VARIOUS POINTERS AND FLAGS MAY NOT BE CORRECTLY ADJUSTED.

12.3 ERROR HANDLER

AN EMT TRAP INSTRUCTION IS USED BY THE ERROR CALL. THE LOWER BYTE IS ENCODED TO GIVE DIFFERENT ERROR CALLS. (EX: ERROR 1 = 104000+1; ERROR 16 = 104000+16). WHEN THE ERROR STATEMENT IS EXECUTED, A TRAP OCCURS TO MEMORY LOCATION '\$ERROR'. THE ERROR HANDLER IS LOCATED AT '\$ERROR'. THE HANDLER FORMS THE POINTER TO ERROR TABLE, WHICH IS USED IF AN ERROR MESSAGE IS TO BE TYPED DEPENDING ON THE SWITCH SETTINGS, A DECISION ABOUT HALTING ON ERROR, INHIBITING TYPEOUT, LOOPING ON ERROR ETC. IS MADE. IF AN ERROR MESSAGE IS TO BE TYPED OUT AN EXIT IS MADE TO THE ERROR MESSAGE TYPEOUT ROUTINE LOCATED AT '\$ERRTYP'.

12.4 CONTROL RESET ROUTINE

THE CALL FOR THIS ROUTINE IS 'CNT.RESET' AND IS AN ENCODED 'TRAP' INSTRUCTION. WHEN 'CNT.RESET' IS EXECUTED THE CONTROL RESET ROUTINE STARTING AT

"CN.RST" IS ENTERED. A CONTROL RESET IS ISSUED THE PROGRAM WAITS TILL THE CONTROL READY SETS, ON WHICH THE ROUTINE IS EXITED. IF CONTROL READY DOES NOT SET WITHIN A CERTAIN TIME AN ERROR IS REPORTED. THE PC TYPED OUT IS THE LOCATION WHERE THE "CNT.RESET" CALL IS LOCATED. THE WAITING TIME IS 2.8 MS FOR 11/20 AND 560 US FOR 11/45 WITH BIPOLAR MEMORY.

12.5 CONTROL READY ROUTINE

THIS ROUTINE IS CALLED BY "CNT.RDY" (AN ENCODED 'TRAP' INSTRUCTION) AND IS LOCATED AT "CN.RDY". THE ROUTINE WAITS FOR THE CONTROL READY TO SET AND WHEN IT DOES, EXITS IF CONTROL READY DOES NOT SET WITHIN A SPECIFIED TIME AN ERROR MESSAGE IS GIVEN

CNTRL RDY DIDN'T SET
PC = XXXXXX RKCS = YYYYYY

THE PC IS THE LOCATION AT WHICH THE "CNT.RDY" CALL IS LOCATED. THE WAITING TIME IS 949 MS FOR 11/20 AND 189 MS FOR 11/45 WITH BIPOLAR MEMORY.

12.6 DRIVE RESET ROUTINE

THE DRIVE - RESET ROUTINE IS LOCATED AT "DRESET" AND IS CALLED BY A "JSR". IT ISSUES A DRIVE RESET AND WAITS FOR THE R/W/S RDY TO SET, ON WHICH THE ROUTINE IS EXITED. THE WAITING TIME IS 4959 MS FOR 11/20 AND 991 MS FOR 11/45 WITH BIPOLAR MEMORY.

12.7 TIME DELAY ROUTINE

THIS ROUTINE PROVIDES A VARIABLE TIME DELAY. THE CALL IS DELAY ,N WHERE N=1 TO 177777 (OCTAL) TIME DELAY PROVIDED= 7.5 TIMES(X) N MICRO SECS FOR 11/20, 1.5N US FOR 11/45 (N CONVERTED TO DECIMAL BEFORE COMPUTING DELAY) IF THE USER WANTS TO CHANGE THE DELAY AT ANY POINT IT CAN BE DONE BY SIMPLY CHANGING VARIABLE 'N'.

12.8 WAIT FOR INTERRUPT ROUTINE

THIS ROUTINE PROVIDES A VARIABLE TIME LIMIT DURING WHICH RK11 INTERRUPT MAY OCCUR. THE IS
WAT.INT ,N N=1 TO 1777777 (OCTAL)
WAITING TIME=7.5 TIMES(X) N US FOR 11/20, 1.5N US

FOR 11/45 UPON ENTERING THE ROUTINE CPU PRIORITY IS DROPPED SO THAT RK11 CAN INTERRUPT.

12.9 OTHER ROUTINES

THERE ARE OTHER COMMONLY USED ROUTINES AS LISTED BELOW.

\$TYPE:
TYPE ROUTINE FOR TYPING OUT ASCII STRINGS.
LOCATED AT "\$TYPE"
CALLED BY "TYPE"

\$TYPOC:
ROUTINE FOR TYPING OUT OCTAL NUMBERS.
LOCATED AT "\$TYPOC"
CALLED BY "TYPOC"

\$TYPDS:
ROUTINE FOR TYPING OUT DECIMAL NUMBERS.
LOCATED AT "\$TYPDS"
CALLED BY "TYPDS"

\$RDLIN:
ROUTINE FOR INPUTTING ASCII STRINGS FROM TTY.
LOCATED AT "\$RDLIN"
CALLED BY "RDLIN"

\$ERRTYP:
ROUTINE FOR TYPING OUT ERROR MESSAGES.
LOCATED AT \$ERRTYP
CALLED BY "JSR \$ERRTYP"

\$PWDRN:
ROUTINE FOR HANDLING POWER FAILURE.
LOCATED AT \$PWDRN
CALLED WHEN THERE IS A POWER FAILURE.

\$PWRUP:
ROUTINE FOR HANDLING POWER UP AFTER A POWER FAIL.
LOCATED AT \$PWRUP
CALLED WHEN POWER RETURNS AFTER HAVING GONE DOWN.

13.0 UNEXPECTED TIMEOUTS AND RK11 INTERRUPTS

WHEN AN UNEXPECTED TIMEOUT OCCURS, THE PC AT WHICH TIME OUT OCCURED 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 BEGINING. SW 9- LOOPING CAITY IS PROVIDED AS A TROUBLE SHOOTING AID.

14.0 QUICK VERIFYING MODE

THE FIRST PASS OF THE PROGRAM IS A QUICK VERIFYING MODE. ALL THE TESTS ARE DONE ONLY ONCE, ON SUBSEQUENT PASSES THE TESTS ARE ITERATED (NORMALLY 50 TIMES, 5 IN SOME CASES). THUS THE FIRST PASS TAKES A SHORTER TIME TO COMPLETE, WHEREAS SUBSEQUENT PASSES TAKE MORE TIME.

z

852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893

```
.TITLE MD-11-CZRKKF, RK11 BASIC LOGIC TEST 2
;*COPYRIGHT (C) 1974,1977
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY JIM KAPADIA
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
;*
;*PROGRAM REVISED BY TOM SAWYER, MARCH, 1976
;*REVISED BY CHUCK HESS, AUGUST, 1976
.SBTL OPERATIONAL SWITCH SETTINGS
;*
;*      SWITCH          USE
;*      -----          -
;*      15             HALT ON ERROR
;*      14             LOOP ON TEST
;*      13             INHIBIT ERROR TYPEOUTS
;*      12             CYCLE ON ERROR TO PREVIOUS 'SCOPE' STATEMENT
;*      11             INHIBIT ITERATIONS
;*      10             TESTING ON SIMULATOR
;*      9              LOOP ON ERROR
;*      8              LOOP ON TEST IN SWR<7:0>
;*      6              DROP THE DRIVE IF MORE THAN 5 ERRORS
;*
;*****
;YOU ARE ADVISED TO READ THE DOCUMENT BEFORE USING THIS PROGRAM.
;ON GETTING AN ERROR REFER TO THE LISTINGS AT THE PC POINTED
```

```
894 :OUT IN THE ERROR MESSAGE. ADJACENT ERROR MESSAGES IF FOLLOWED
895 :CAREFULLY COULD LEAD TO AN EASY PINPOINTING OF THE FAULT
896
897 :*****
898 .SBTTL BASIC DEFINITIONS
899
900 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
901 001100 STACK= 1100
902 .EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
903 .EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
904
905 ;*MISCELLANEOUS DEFINITIONS
906 000011 HT= 11 ;;CODE FOR HORIZONTAL TAB
907 000012 LF= 12 ;;CODE FOR LINE FEED
908 000015 CR= 15 ;;CODE FOR CARRIAGE RETURN
909 000200 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
910 177776 PS= 177776 ;;PROCESSOR STATUS WORD
911 .EQUIV PS,PSW
912 177774 STKLMT= 177774 ;;STACK LIMIT REGISTER
913 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
914 177570 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
915 177570 DDISP= 177570 ;;HARDWARE DISPLAY REGISTER
916
917 ;*GENERAL PURPOSE REGISTER DEFINITIONS
918 000000 R0= X0 ;;GENERAL REGISTER
919 000001 R1= X1 ;;GENERAL REGISTER
920 000002 R2= X2 ;;GENERAL REGISTER
921 000003 R3= X3 ;;GENERAL REGISTER
922 000004 R4= X4 ;;GENERAL REGISTER
923 000005 R5= X5 ;;GENERAL REGISTER
924 000006 R6= X6 ;;GENERAL REGISTER
925 000007 R7= X7 ;;GENERAL REGISTER
926 000006 SP= X6 ;;STACK POINTER
927 000007 PC= X7 ;;PROGRAM COUNTER
928
929 ;*PRIORITY LEVEL DEFINITIONS
930 000000 PR0= 0 ;;PRIORITY LEVEL 0
931 000040 PR1= 40 ;;PRIORITY LEVEL 1
932 000100 PR2= 100 ;;PRIORITY LEVEL 2
933 000140 PR3= 140 ;;PRIORITY LEVEL 3
934 000200 PR4= 200 ;;PRIORITY LEVEL 4
935 000240 PR5= 240 ;;PRIORITY LEVEL 5
936 000300 PR6= 300 ;;PRIORITY LEVEL 6
937 000340 PR7= 340 ;;PRIORITY LEVEL 7
938
939 ;*'SWITCH REGISTER' SWITCH DEFINITIONS
940 100000 SW15= 100000
941 040000 SW14= 40000
942 020000 SW13= 20000
943 010000 SW12= 10000
944 004000 SW11= 4000
945 002000 SW10= 2000
946 001000 SW09= 1000
947 000400 SW08= 400
948 000200 SW07= 200
949 000100 SW06= 100
```

```
950      000040      SW05= 40
951      000020      SW04= 20
952      000010      SW03= 10
953      000004      SW02= 4
954      000002      SW01= 2
955      000001      SW00= 1
956      .EQUIV SW09,SW9
957      .EQUIV SW08,SW8
958      .EQUIV SW07,SW7
959      .EQUIV SW06,SW6
960      .EQUIV SW05,SW5
961      .EQUIV SW04,SW4
962      .EQUIV SW03,SW3
963      .EQUIV SW02,SW2
964      .EQUIV SW01,SW1
965      .EQUIV SW00,SW0
966
967      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
968      100000      BIT15= 100000
969      040000      BIT14= 40000
970      020000      BIT13= 20000
971      010000      BIT12= 10000
972      004000      BIT11= 4000
973      002000      BIT10= 2000
974      001000      BIT09= 1000
975      000400      BIT08= 400
976      000200      BIT07= 200
977      000100      BIT06= 100
978      000040      BIT05= 40
979      000020      BIT04= 20
980      000010      BIT03= 10
981      000004      BIT02= 4
982      000002      BIT01= 2
983      000001      BIT00= 1
984      .EQUIV BIT09,BIT9
985      .EQUIV BIT08,BIT8
986      .EQUIV BIT07,BIT7
987      .EQUIV BIT06,BIT6
988      .EQUIV BIT05,BIT5
989      .EQUIV BIT04,BIT4
990      .EQUIV BIT03,BIT3
991      .EQUIV BIT02,BIT2
992      .EQUIV BIT01,BIT1
993      .EQUIV BIT00,BIT0
994
995      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
996      000004      ERRVEC= 4          ;; TIME OUT AND OTHER ERRORS
997      000010      RESVEC= 10         ;; RESERVED AND ILLEGAL INSTRUCTIONS
998      000014      TBITVEC=14        ;; "T" BIT
999      000014      TRTVEC= 14         ;; TRACE TRAP
1000     000014      BPTVEC= 14         ;; BREAKPOINT TRAP (BPT)
1001     000020      IOTVEC= 20         ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
1002     000024      PWRVEC= 24         ;; POWER FAIL
1003     000030      EMTVEC= 30         ;; EMULATOR TRAP (EMT) **ERROR**
1004     000034      TRAPVEC=34        ;; "TRAP" TRAP
1005     000060      TKVEC= 60          ;; TTY KEYBOARD VECTOR
```

```
1006      000064      TPVEC= 64          ;;TTY PRINTER VECTOR
1007      000240      PIRQVEC=240       ;;PROGRAM INTERRUPT REQUEST VECTOR
1008
1009      .SBTTL TRAP CATCHER
1010
1011      .=0
1012      ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
1013      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1014      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1015      000174      000174      .=174
1016      000176      000000      DISPREG: .WORD 0          ;;SOFTWARE DISPLAY REGISTER
1017      000176      000000      SWREG: .WORD 0          ;;SOFTWARE SWITCH REGISTER
1018      000200      000137      002636 .SBTTL STARTING ADDRESS(ES)
1019      .SBTTL JMP @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
1020      .SBTTL ACT11 HOOKS
1021      ;*****
1022      ;HOOKS REQUIRED BY ACT11
1023      000204      $SVPC=.          ;SAVE PC
1024      000046      .=46
1025      000046      020740      $ENDAD          ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
1026      000052      000052      .=52
1027      000052      000000      .WORD 0          ;;2)SET LOC.52 TO ZERO
1028      000204      .=$SVPC          ;; RESTORE PC
```

```
1029          .SBTTL COMMON TAGS
1030
1031          ;:*****
1032          ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1033          ;*USED IN THE PROGRAM.
1034
1035          001100          .=1100
1036 001100          SCMTAG:          ;; START OF COMMON TAGS
1037 001100 000000          $PASS: .WORD 0          ;; CONTAINS PASS COUNT
1038 001102 000          $TSTNM: .BYTE 0          ;; CONTAINS THE TEST NUMBER
1039 001103 000          $ERFLG: .BYTE 0          ;; CONTAINS ERROR FLAG
1040 001104 000000          $ICNT: .WORD 0          ;; CONTAINS SUBTEST ITERATION COUNT
1041 001106 000000          $LPADR: .WORD 0          ;; CONTAINS SCOPE LOOP ADDRESS
1042 001110 000000          $LPERR: .WORD 0          ;; CONTAINS SCOPE RETURN FOR ERRORS
1043 001112 000000          $ERTTL: .WORD 0          ;; CONTAINS TOTAL ERRORS DETECTED
1044 001114 000          $ITEMB: .BYTE 0          ;; CONTAINS ITEM CONTROL BYTE
1045 001115 001          $ERMAX: .BYTE 1          ;; CONTAINS MAX. ERRORS PER TEST
1046 001116 000000          $ERRPC: .WORD 0          ;; CONTAINS PC OF LAST ERROR INSTRUCTION
1047 001120 000000          $GDADR: .WORD 0          ;; CONTAINS ADDRESS OF 'GOOD' DATA
1048 001122 000000          $BDADR: .WORD 0          ;; CONTAINS ADDRESS OF 'BAD' DATA
1049 001124 000000          $GDDAT: .WORD 0          ;; CONTAINS 'GOOD' DATA
1050 001126 000000          $BDDAT: .WORD 0          ;; CONTAINS 'BAD' DATA
1051 001130 000000          .WORD 0          ;; RESERVED--NOT TO BE USED
1052 001132 000000          .WORD 0
1053 001134 000          $AUTOB: .BYTE 0          ;; AUTOMATIC MODE INDICATOR
1054 001135 000          $INTAG: .BYTE 0          ;; INTERRUPT MODE INDICATOR
1055 001136 000000          .WORD 0
1056 001140 177570          $SWR: .WORD DSWR          ;; ADDRESS OF SWITCH REGISTER
1057 001142 177570          $DISPLAY: .WORD DDISP          ;; ADDRESS OF DISPLAY REGISTER
1058 001144 177560          $TKS: 177560          ;; TTY KBD STATUS
1059 001146 177562          $TKB: 177562          ;; TTY KBD BUFFER
1060 001150 177564          $TPS: 177564          ;; TTY PRINTER STATUS REG. ADDRESS
1061 001152 177566          $TPB: 177566          ;; TTY PRINTER BUFFER REG. ADDRESS
1062 001154 000          $NULL: .BYTE 0          ;; CONTAINS NULL CHARACTER FOR FILLS
1063 001155 002          $FILLS: .BYTE 2          ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
1064 001156 012          $FILLC: .BYTE 12          ;; INSERT FILL CHARS. AFTER A 'LINE FEED'
1065 001157 000          $TPFLG: .BYTE 0          ;; 'TERMINAL AVAILABLE' FLAG (BIT<07>=0=YES)
1066 001160 000000          $REGAD: .WORD 0          ;; CONTAINS THE ADDRESS FROM
1067          ;; WHICH ($REGO) WAS OBTAINED
1068 001162 000000          $REG0: .WORD 0          ;; CONTAINS (($REGAD)+0)
1069 001164 000000          $REG1: .WORD 0          ;; CONTAINS (($REGAD)+2)
1070 001166 000000          $REG2: .WORD 0          ;; CONTAINS (($REGAD)+4)
1071 001170 000000          $REG3: .WORD 0          ;; CONTAINS (($REGAD)+6)
1072 001172 000000          $REG4: .WORD 0          ;; CONTAINS (($REGAD)+10)
1073 001174 000000          $REG5: .WORD 0          ;; CONTAINS (($REGAD)+12)
1074 001176 000000          $REG6: .WORD 0          ;; CONTAINS (($REGAD)+14)
1075 001200 000000          $REG7: .WORD 0          ;; CONTAINS (($REGAD)+16)
1076 001202 000000          $REG10: .WORD 0          ;; CONTAINS (($REGAD)+20)
1077 001204 000000          $REG11: .WORD 0          ;; CONTAINS (($REGAD)+22)
1078 001206 000000          $TIMES: 0          ;; MAX. NUMBER OF ITERATIONS
1079 001210 000000          $ESCAPE: 0          ;; ESCAPE ON ERROR ADDRESS
1080 001212 077          $QUES: .ASCII /?/          ;; QUESTION MARK
1081 001213 015          $CRLF: .ASCII <15>          ;; CARRIAGE RETURN
1082 001214 000012          $LF: .ASCIIZ <12>          ;; LINE FEED
1083          ;:*****
1084 001216 005015 051104 053111 $MSG1: .ASCIIZ <15><12>/DRIVE PRESENT/
```

```

1085 001224 020105 051120 051505
1086 001232 052116 000
1087 001236
1088 001236 005015 047516 042516 MSG2: .EVEN
      001236 .ASCIZ <15><12>/NONE/
1089 001244 000
1090
1091 001245 015 041412 052116 MSG3: .ASCIZ <15><12>/CNT RDY DIDN'T SET/
1092 001252 051040 054504 042040
1093 001260 042111 023516 020124
1094 001266 042523 000124
1095
1096 001272 005015 051104 053111 MSG4: .ASCIZ <15><12>/DRIVE /
1097 001300 020105 000
1098
1099 001303 015 040412 046114 MSG5: .ASCII <15><12>/ALL DRVS/
1100 001310 042040 053122 123
1101
1102 001315 040 051104 050117 MSG6: .ASCIZ / DROPD/<15><12>
1103 001322 006504 000012
1104 .EVEN
1105
1106 :RK11 REGISTERS
1107 :IF FOR ANY REASON THE REGISTER ADDRESSES ARE DIFFERENT FROM THESE
1108 :(GIVEN BELOW), THE CONTENTS OF THE APPROPRIATE POINTERS SHOULD BE
1109 :MODIFIED SO THAT THE CORRECT ADDRESS IS USED.
1110 :
1111 :
1112 001326 177400 RKDS: .EVEN 177400
1113 001330 177402 RKER: 177402
1114 001332 177404 RKCS: 177404
1115 001334 177406 RKWC: 177406
1116 001336 177410 RKBA: 177410
1117 001340 177412 RKDA: 177412
1118 001342 177416 RKDB: 177416
1119
1120
1121 :TAGS AND GENERAL DATA AREA
1122 :
1123 :
1124 :
1125 001344 000000 SIMUL: 0 ;FLAG TO BE SET TO 1 WHEN ON SIMULATOR
1126 001346 000000 FTITLE: 0 ;FLAG FOR PRINTING PROGRAM TITLE
1127 001350 000000 DRIVAD: 0 ;CONTAINS ADDRESS OF THE DRIVE UNDER TEST
1128 001352 000000 DRVDON: 0 ;CONTAINS THE NUMBER OF DRIVES CHECKED.
1129 ;IT IS INCREMENTED EACH TIME THE TESTS FOR
1130 ;A DRIVE IS COMPLETED.
1131 001354 000000 DRVPTR: 0 ;CONTAINS THE POINTER TO THE DRIVE FLAG (DRIVED
1132 ;-DRIVE7) OF THE DRIVE TO BE CHECKED NEXT.
1133 001356 000000 INDX1: 0 ;GENERAL INDEX FOR KEEPING COUNT
1134 001360 000000 INDX2: 0 ;GENERAL INDEX
1135 001362 000000 COUNT: 0 ;GENERAL COUNT REGISTER
1136 001364 000000 COUNT1: 0 ;COUNT REGISTER USED FOR 'DRESET' SUBROUTINE
1137 001366 000000 TIMER: 0 ;TIMER REGISTER
1138 001370 000000 EFLG1: 0 ;SET, TO INDICATE A PARTICULAR
1139 ;ERROR CONDITION
1140

```

1141	001372	000100	SEEK0:	100	:CONTAINS ADDRESS OF CYLINDER 2
1142	001374	001000	SEEK1:	1000	:CONTAINS ADDRESS OF CYLINDER 20
1143	001376	014500	SEEK2:	14500	:CONTAINS ADDRESS OF CYLINDER 312
1144	001400	000200	RKPRI:	200	:CONTAINS THE CPU LEVEL AT WHICH
1145					:RK11 NORMALLY INTERRUPTS. THIS WORD
1146					:SHOULD BE CHANGED IF RK11 IS DESINGATED
1147					:A BR LEVEL OTHER THAN 5. E.G. IF IT IS CHANGED
1148					:TO 6, THIS WORD SHOULD BE CHANGED TO 240.
1149	001402	000220	RKVEC:	220	:CONTAINS THE NORMAL VECTOR ADDRESS TO WHICH
1150					:RK11 INTERRUPTS. IF THIS IS NOT SO, CHANGE
1151					:THIS WORD TO CONTAIN MODIFIED VECTOR ADDRESS.
1152	001404	000000	FFLAG:	0	
1153	001406	000000	ODDEVN:	0	:USED TO DETERMINE WHICH OF RK-05F DRIVES ACTIVE
1154					:0 IF EVEN DRIVE
1155					:-1 IF ODD DRIVE
1156	001410	000000	DDPCH:	0	:IF PROGRAM LOADED FROM RK05, CONTAINS
1157					:ADDRESS OF DRIVE WITH RKDP PACK
1158	001412	000000	DRIVS:	0	:CONTAINS THE NUMBER OF DRIVES PRESENT
1159					
1160					
1161					
1162					
1163					:THE FLAGS BELOW (BIT 0) ARE SET TO 1 TO INDICATE THAT A PARTICULAR DRIVE
1164					:IS PRESENT AND IS TO BE TESTED. BIT 12, IF SET, INDICATES THAT THE DRIVE
1165					:WAS DROPPED AFTER MAXIMUM ALLOWABLE NUMBER OF ERRORS OCCURED ON THAT
1166					:DRIVE (SW 6 SET).
1167					:IF MORE THAN 5 ERRORS OCCUR IN THE HARDWARE POLLING TEST (LAST)
1168					:THEN ALL DRIVES ARE DROPPED. BUT BIT 12 IS NOT SET.
1169					
1170	001414	000000	DRIV0:	0	:FLAG SET TO 1 WHEN DRIVE 0 PRESENT
1171	001416	000000	DRIV1:	0	:FOR DRIVE 1
1172	001420	000000	DRIV2:	0	:FOR DRIVE 2
1173	001422	000000	DRIV3:	0	:FOR DRIVE 3
1174	001424	000000	DRIV4:	0	:FOR DRIVE 4
1175	001426	000000	DRIV5:	0	:FOR DRIVE 5
1176	001430	000000	DRIV6:	0	:FOR DRIVE 6
1177	001432	000000	DRIV7:	0	:FOR DRIVE 7
1178					
1179	001434	000000	T56FLG:	0	
1180	001436	000000	PHYDRV:	0	
1181	001440	000000	SIZYET:	0	

1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237

001442

.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

\$ERRTB:

THE ERROR ITEMS TABLE CONSISTS OF ALL THE POSSIBLE ERROR MESSAGES
USED IN THIS PROGRAM. AN ERROR CALL IN THE PROGRAM CORRESPONDS TO
THE ITEM NUMBER IN THE ERROR TABLE. THUS 'ERROR 1' IN THE
PROGRAM CORRESPONDS TO 'ITEM 1' IN THE ERROR TABLE.
'EM###' IS THE POINTER TO THE ERROR MESSAGE WHICH WILL BE TYPED
OUT IN CASE THAT ERROR WERE TO OCCUR. THUS FOR 'ERROR 1' THE ERROR
MESSAGE TYPE OUT WILL BE 'TIME OUT ON RK11 REG'.
'DH###' IS THE POINTER TO THE HEADER BLOCK WHICH WILL BE TYPED OUT
IMMEDIATELY AFTER THE ERROR MESSAGE.
'DT###' SERVES AS A POINTER TO THE MEMORY LOCATIONS WHERE
THE INFORMATION RELEVANT TO THE ERROR TYPE OUTS (LIKE PC, CONTENTS
OF RKCS ETC.) WILL BE PICKED UP FROM.
THE LAST ROW CONTAINING '0' SERVES AS A TERMINATOR.
EXAMPLE:
IF ON RUNNING THIS PROGRAM A TIMEOUT WERE TO OCCUR ON ADDRESSING RKDS
(177400), BECAUSE OF SOME FAULT, THE FOLOWING TYPEOUT WOULD
OCCUR ON THE TELETYPE.

```
TIME OUT ON RK11 REG
PC      REG
##### 177400
```

NOTE THAT ##### WOULD BE THE ACTUAL PC WHERE 'ERROR 1' IS LOCATED.

THE ERROR HANDLER IS LOCATED AT '\$ERROR'. THE ERROR CALL IS AN 'EMT'
INSTRUCTION WITH ITS LOWER BYTE ENCODED TO PROVIDE INDEXING TO THE
ITEMS IN THE ERROR TABLE.
THUS 'ERROR 1' IS 104001
'ERROR 103' IS 104126 ETC.

;ERROR ITEMS TABLE


```
1406 ;ITEM 31
1407
1408 001742 026452 EM54 ;'CNTRL RDY' DIDN'T SET ON DOING WRITE/FMT STARTING
1409 ;FROM <DSK-ADRES>
1410 001744 032374 DH54 ;PC RKCS RKER RKDS RKDA
1411 ;DRV# CYL <DSK-ADRES> SUR SECTR
1412 001746 031774 DT54 ;$ERRPC $REG0 $REG1 $REG2 $REG3
1413 ;$REG4 $REG5 $REG6 $REG>
1414 001750 000000 0
1415
1416 ;ITEM 32
1417
1418 001752 026544 EM55 ;'HE' OR 'ERR' ON WRITE/FMT STARTING FROM
1419 ;<DSK-ADRES>
1420 001754 032374 DH54 ;PC RKCS RKER RKDS RKDA
1421 ;DRV# CYL <DSK-ADRES> SUR SECTR
1422 001756 031774 DT54 ;$ERRPC $REG0 $REG1 $REG2 $REG3
1423 ;$REG4 $REG5 $REG6 $REG7
1424 001760 000000 0
1425
1426 ;ITEM 33
1427
1428 001762 026623 EM56 ;RKDA INCREMENTED WRONG ON WRITE OR WRITE FORMAT
1429 001764 032503 DH56 ;PC EXPCT: DRV# CYL SUR SECTR
1430 ;RECV: DRV# CYL SUR SECTR
1431 001766 031774 DT54 ;$ERRPC $REG0 $REG1 $REG2 $REG3
1432 ;$REG4 $REG5 $REG6 $REG7
1433 001770 000000 0
1434
1435 ;ITEM 34
1436
1437 001772 026662 EM57 ;RKWC DIDN'T OVERFLOW ON WRITE OR WRITE FORMAT
1438 001774 032075 DH5 ;PC RECV
1439 001776 031724 DT1 ;$ERRPC $REG0
1440 002000 000000 0
1441
1442 ;ITEM 35
1443
1444 002002 026720 EM60 ;RKBA INCREMENTED WRONG ON WRITE OR WRITE FORMAT
1445 002004 032047 DH4 ;PC EXPCT RECV
1446 002006 031732 DT2 ;$ERRPC $REG0 $REG1
1447 002010 000000 0
1448
1449 ;ITEM 36
1450
1451 002012 026757 EM61 ;RKER SET, ON WRITE/READ/FORMAT
1452 002014 032155 DH30 ;PC RKCS RKER RKDS
1453 002016 031762 DT26 ;$ERRPC $REG0 $REG1 $REG2
1454 002020 000000 0
1455
1456 ;ITEM 37
1457
1458 002022 027014 EM62 ;RKDB ERROR
1459 002024 032047 DH4 ;PC EXPCT RECV
1460 002026 031732 DT2 ;$ERRPC $REG0 $REG1
1461 002030 000000 0
```


1518	002120	000000	0					
1519								
1520			:ITEM	47				
1521								
1522	002122	027422	EM72		:WRONG DRIVE ID IN RKDS AFTER SEEK			
1523	002124	032047	DH4		:PC EXPCT RECVD			
1524	002126	031732	DT2		:SERRPC \$REG0 \$REG1			
1525	002130	000000	0					
1526								
1527			:ITEM	50				
1528								
1529	002132	027464	EM73		:HARDWARE POLL, DRIVE ID BITS(13-15) SHOULD BE CLEAR			
1530	002134	032213	DH34		:PC RKDS			
1531	002136	031732	DT2		:SERRPC \$REG0			
1532	002140	000000	0					
1533								
1534			:ITEM	51				
1535								
1536	002142	027536	EM74		:HARDWARE POLL, INTERRUPTING DRIVE # NOT PRESENT			
1537	002144	032732	DH74		:PC DRIVE #			
1538	002146	031724	DT1		:SERRPC \$REG0			
1539	002150	000000	0					
1540								
1541			:ITEM	52				
1542								
1543	002152	027606	EM75		: 'DRIVE #' DID NOT INTERRUPT DURING HARDWARE POLL			
1544	002154	032732	DH74		:PC DRIVE #			
1545	002156	031724	DT1		:SERRPC \$REG0			
1546	002160	000000	0					
1547								
1548			:ITEM	53				
1549								
1550	002162	027656	EM76		:SCP DID NOT SET AFTER WAS DONE			
1551	002164	033106	DH117		:PC RKCS			
1552	002166	031724	DT1		:SERRPC \$REG0			
1553	002170	000000	0					
1554								
1555			:ITEM	54				
1556								
1557	002172	027721	EM77		:RKDA CHANGED AFTER 'DRIVE RESET'			
1558	002174	032047	DH4		:PC EXPCT RECVD			
1559	002176	031732	DT2		:SERRPC \$REG0 \$REG1			
1560	002200	000000	0					
1561								
1562			:ITEM	55				
1563								
1564	002202	027756	EM100		:DATA ERROR AT WORD#			
1565	002204	032753	DH100		:PC WORD# EXPCT RECVD			
1566	002206	031762	DT26		:SERRPC \$REG0 \$REG1 \$REG2			
1567	002210	000000	0					
1568								
1569			:ITEM	56				
1570								
1571	002212	030001	EM101		:CNTRL RDY DID NOT SET AFTER READ CHECK			
1572	002214	032327	DH44		:PC RKCS RKER RKDS RKDA			
1573	002216	031742	DT20		:SERRPC \$REG0 \$REG1 \$REG2 \$REG3			

1574	002220	000000	0	
1575				
1576			:ITEM	57
1577				
1578	002222	030043	EM102	: 'ERR' OF 'HE' SET ON READ CHECK
1579	002224	032155	DH30	:PC RKCS RKER RKDS
1580	002226	031762	DT26	: \$ERRPC \$REG0 \$REG1 \$REG2
1581	002230	000000	0	
1582				
1583			:ITEM	60
1584				
1585	002232	030067	EM103	: 'CSE' ON READ CHECK
1586	002234	033010	DH103	:PC RKER
1587	002236	031724	DT1	: \$ERRPC \$REG0
1588	002240	000000	0	
1589				
1590			:ITEM	61
1591				
1592	002242	030105	EM104	:RKWC DID NOT OVERFLOW ON READ CHECK OR WRITE CHECK
1593	002244	033024	DH104	:PC RECVD RKCS
1594	002246	031732	DT2	: \$ERRPC \$REG0 \$REG1
1595	002250	000000	0	
1596				
1597			:ITEM	62
1598				
1599	002252	030156	EM105	:RKDA INCREMENTED WRONG ON READ CHECK
1600	002254	032047	DH4	:PC EXPCT RECVD
1601	002256	031732	DT2	: \$ERRPC \$REG0 \$REG1
1602	002260	000000	0	
1603				
1604			:ITEM	63
1605				
1606	002262	030214	EM106	:RKBA CHANGED AFTER READ CHECK
1607	002264	032047	DH4	:PC EXPCT RECVD
1608	002266	031732	DT2	: \$ERRPC \$REG0 \$REG1
1609	002270	000000	0	
1610				
1611			:ITEM	64
1612				
1613	002272	030245	EM107	:MEMORY WORD CHANGED AFTER READ CHECK
1614	002274	033050	DH107	:PC LOC EXPCT RECVD
1615	002276	031762	DT26	: \$ERRPC \$REG0 \$REG1 \$REG2
1616	002300	000000	0	
1617				
1618			:ITEM	65
1619				
1620	002302	030306	EM110	:CNTRL RDY DID NOT SET AFTER WRITE CHECK
1621	002304	032327	DH44	:PC RKCS RKER RKDS RKDA
1622	002306	031742	DT20	: \$ERRPC \$REG0 \$REG1 \$REG2 \$REG3
1623	002310	000000	0	
1624				
1625			:ITEM	66
1626				
1627	002312	030351	EM111	:HE OR ERR BIT SET AFTER DOING WRITE CHECK
1628	002314	032155	DH30	:PC RKCS RKER RKDS
1629	002316	031762	DT26	: \$ERRPC \$REG0 \$REG1 \$REG2

1630	002320	000000	0					
1631								
1632			:ITEM	67				
1633								
1634	002322	030376	EM112		:WRITE CHECK ERROR			
1635	002324	032155	DH30		:PC RKCS RKER RKDS			
1636	002326	031762	DT26		:SERRPC \$REG0 \$REG1 \$REG2			
1637	002330	000000	0					
1638								
1639			:ITEM	70				
1640								
1641	002332	030417	EM113		:RKDA INCREMENTED WRONG ON WRITE CHECK			
1642	002334	032047	DH4		:PC EXPCT RECVD			
1643	002336	031732	DT2		:SERRPC \$REG0 \$REG1			
1644	002340	000000	0					
1645								
1646			:ITEM	71				
1647								
1648	002342	030456	EM114		:RKBA INCREMENTED WRONG ON WRITE CHECK			
1649	002344	032047	DH4		:PC EXPCT RECVD			
1650	002346	031732	DT2		:SERRPC \$REG0 \$REG1			
1651	002350	000000	0					
1652								
1653			:ITEM	72				
1654								
1655	002352	030515	EM115		:RKBA INCREMENTED WITH IBA SET			
1656	002354	032047	DH4		:PC EXPCT RECVD			
1657	002356	031732	DT2		:SERRPC \$REG0 \$REG1			
1658	002350	000000	0					
1659								
1660			:ITEM	73				
1661								
1662	002362	030551	EM116		:WRONG MEMORY LOCATION CHANGED WITH IBA SET			
1663	002364	032753	DH100		:PC WORD# EXPCT RECVD			
1664	002366	031762	DT26		:SERRPC \$REG0 \$REG1 \$REG2			
1665	002370	000000	0					
1666								
1667			:ITEM	74				
1668								
1669	002372	030624	EM117		:RK11 DID NOT INTERRUPT WHEN IDE WAS SET			
1670	002374	033106	DH117		:PC RKCS			
1671	002376	031724	DT1		:SERRPC \$REG0			
1672	002400	000000	0					
1673								
1674			:ITEM	75				
1675								
1676	002402	030671	EM120		:RK11 DID NOT INTERRUPT AFTER SEEK WAS INITIATED			
1677	002404	033106	DH117		:PC RKCS			
1678	002406	031724	DT1		:SERRPC \$REG0			
1679	002410	000000	0					
1680								
1681			:ITEM	76				
1682								
1683	002412	030744	EM121		:SCP SET BEFORE SEEK COMPLETED			
1684	002414	033106	DH117		:PC RKCS			
1685	002416	031724	DT1		:SERRPC \$REG0			

1686	002420	000000	0	
1687				
1688			:ITEM	77
1689				
1690	002422	031002	EM122	:RK11 DID NOT INTERRUPT AFTER SEEK COMPLETED
1691	002424	032155	DH30	:PC RKCS RKER RKDS
1692	002426	031762	DT26	:\$ERRPC \$REG0 \$REG1 \$REG2
1693	002430	000000	0	
1694				
1695			:ITEM	100
1696				
1697	002432	031051	EM123	:CNTRL RESET DID NOT CLEAR 'SCP' BIT
1698	002434	033106	DH117	:PC RKCS
1699	002436	031724	DT1	:\$ERRPC \$REG0
1700	002440	000000	0	
1701				
1702			:ITEM	101
1703				
1704	002442	031110	EM124	:RK11 DID NOT INTERRUPT AFTER READ WAS DONE
1705	002444	033106	DH117	:PC RKCS
1706	002446	031724	DT1	:\$ERRPC \$REG0
1707	002450	000000	0	
1708				
1709			:ITEM	102
1710				
1711	002452	031152	EM125	:CNTRL RESET DID NOT CLEAR REGISTER
1712	002454	032020	DH2	:PC REGADD RECVD
1713	002456	031732	DT2	:\$ERRPC \$REG0 \$REG1
1714	002460	000000	0	
1715				
1716			:ITEM	103
1717				
1718	002462	031211	EM126	:RK11 DID NOT INTERRUPT AT CPU LEVEL
1719	002464	033122	DH126	:PC LEVEL RKCS
1720	002466	031732	DT2	:\$ERRPC \$REG0 \$REG1
1721	002470	000000	0	
1722				
1723			:ITEM	104
1724				
1725	002472	031252	EM127	:RK11 INTERRUPTED AT WRONG CPU LEVEL
1726	002474	033122	DH126	:PC LEVEL RKCS
1727	002476	031732	DT2	:\$ERRPC \$REG0 \$REG1
1728	002500	000000	0	
1729				
1730			:ITEM	105
1731				
1732	002502	031314	EM130	: 'ERR BIT' DID NOT SET IN RKER
1733	002504	033150	DH130	:PC RKCS RKER ERR BIT
1734	002506	031762	DT26	:\$ERRPC \$REG0 \$REG1 \$REG2
1735	002510	000000	0	
1736				
1737				
1738			:ITEM	106
1739				
1740	002512	031351	EM131	:HE OR ERR DID NOT SET
1741	002514	033207	DH131	:PC RKCS RKER

1798	002614	032732	DH74	:PC	DRIVE #	
1799	002616	031724	DT1	:SERRPC	\$REGO	
1800	002620	000000	0			
1801						
1802			:ITEM	117		
1803						
1804	002622	025370	EM11	:RKWC	ERROR	
1805	002624	032047	DH4	:PC	EXPCT	RECVD
1806	002626	031732	DT2	:SERRPC	\$REGO	\$REG1
1807	002630	000000	0			
1808			:ITEM	120		
1809	002632	031662	EM142			
1810	002634	000000	0			
1811						
1812						
1813						

```

1814 002636 000005      START:  RESET          ;CLEAR THE BUS
1815      ;:GIVE DRIVES TIME TO LOAD HEADS IN CASE OF AN APT START.
1816 002640 023737 000042 000046      CMP      @#42,@#46      ;ARE WE IN ACT11 AUTOMATIC MODE?
1817 002646 001016      BNE      STARTA        ;NO, SKIP DELAY
1818 002650 005077 176464      CLR      @RKDA         ;SELECT UNIT 0
1819 002654 012700 000250      MOV      #250,R0       ;WAIT FOR..
1820 002660 032777 000200 176440 20$:  BIT      #200,@RKDS    ;DRIVE READY..
1821 002666 001006      BNE      STARTA        ;IN CASE..
1822 002670 005001      CLR      R1            ;OF APT..
1823 002672 005301      DEC      R1            ;START, BUT..
1824 002674 001376      BNE      .-2           ;DON'T WAIT..
1825 002676 005300      DEC      R0            ;FOREVER.
1826 002700 001367      BNE      20$          ;
1827 002702 000000      HALT                    ;RKDS BIT 7 (DRIVE READY) NEVER SET
1828 002704
1829      .SBTTL  INITIALIZE THE COMMON TAGS
1830      ;:CLEAR THE COMMON TAGS ($CMTAG) AREA
1831 002704 012706 001100      MOV      #CMTAG,R6     ;:FIRST LOCATION TO BE CLEARED
1832 002710 005026      CLR      (R6)+         ;:CLEAR MEMORY LOCATION
1833 002712 022706 001140      CMP      #SWR,R6 ;:DONE?
1834 002716 001374      BNE      .-6           ;:LOOP BACK IF NO
1835 002720 012706 001100      MOV      #STACK,SP    ;:SETUP THE STACK POINTER
1836      ;:INITIALIZE A FEW VECTORS
1837 002724 012737 022140 000020      MOV      #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
1838 002732 012737 000340 000022      MOV      #340,@IOTVEC+2 ;:LEVEL 7
1839 002740 012737 022412 000030      MOV      #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
1840 002746 012737 000340 000032      MOV      #340,@EMTVEC+2 ;:LEVEL 7
1841 002754 012737 024676 000034      MOV      #STRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
1842 002762 012737 000340 000036      MOV      #340,@TRAPVEC+2 ;:LEVEL 7
1843 002770 012737 024776 000024      MOV      #SPURDN,@PURVEC ;:POWER FAILURE VECTOR
1844 002776 012737 000340 000026      MOV      #340,@PURVEC+2 ;:LEVEL 7
1845 003004 005037 001206      CLR      $TIMES        ;:INITIALIZE NUMBER OF ITERATIONS
1846 003010 005037 001210      CLR      $ESCAPE       ;:CLEAR THE ESCAPE ON ERROR ADDRESS
1847 003014 112737 000001 001115      MOV      #1,$ERMAX     ;:ALLOW ONE ERROR PER TEST
1848 003022 012737 003022 001106      MOV      #,$SLPADR     ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
1849 003030 012737 003030 001110      MOV      #,$SLPERR     ;:SETUP THE ERROR LOOP ADDRESS
1850      ;:SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
1851      ;:EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
1852 003036 013746 000004      MOV      @ERRVEC,-(SP) ;:SAVE ERROR VECTOR
1853 003042 012737 003076 000004      MOV      #64$,@ERRVEC ;:SET UP ERROR VECTOR
1854 003050 012737 177570 001140      MOV      #DSWR,SWR     ;:SETUP FOR A HARDWARE SWICH REGISTER
1855 003056 012737 177570 001142      MOV      #DDISP,DISPLAY ;:AND A HARDWARE DISPLAY REGISTER
1856 003064 022777 177777 176046      CMP      #-1,@SWR     ;:TRY TO REFERENCE HARDWARE SWR
1857 003072 001012      BNE      66$          ;:BRANCH IF NO TIMEOUT TRAP OCCURRED
1858      ;:AND THE HARDWARE SWR IS NOT = -1
1859 003074 000403      BR      65$          ;:BRANCH IF NO TIMEOUT
1860 003076 012716 003104      64$:  MOV      #65$,(SP)   ;:SET UP FOR TRAP RETURN
1861 003102 000002      RTI
1862 003104 012737 000176 001140      65$:  MOV      #SWREG,SWR   ;:POINT TO SOFTWARE SWR
1863 003112 012737 000174 001142      MOV      #DISPREG,DISPLAY
1864 003120 012637 000004      66$:  MOV      (SP)+,@ERRVEC ;:RESTORE ERROR VECTOR
1865
1866 003124 023737 000042 000046      CMP      @#42,@#46     ;ARE WE IN ACT11 AUTOMATIC MODE?
1867 003132 001416      BEQ      69$          ;YES, SKIP TITLE
1868      .SBTTL  TYPE PROGRAM NAME
1869      ;:TYPE THE NAME OF THE PROGRAM IF FIRST PASS

```

```

1870 003134 005227 177777      INC      #-1      ;;FIRST TIME?
1871 003140 001043      BNE      67$      ;;BRANCH IF NO
1872 003142 104401 003200      TYPE     ,68$     ;;TYPE ASCIZ STRING
1873      .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
1874 003146 005737 000042      TST     @#42     ;;ARE WE RUNNING UNDER XXDP/ACT?
1875 003152 001006      BNE      69$      ;;BRANCH IF YES
1876 003154 023727 001140 000176      CMP     SWR,#SWREG ;;SOFTWARE SWITCH REG SELECTED?
1877 003162 001005      BNE      70$      ;;BRANCH IF NO
1878 003164 104406      GTSWR                ;;GET SOFT-SWR SETTINGS
1879 003166 000403      BR      70$
1880 003170 112737 000001 001134 69$:  MOVB    #1,$AUTOB  ;;SET AUTO-MODE INDICATOR
1881 003176 70$:
1882 003176 000424      BR      67$      ;;GET OVER THE ASCIZ
1883      ;;68$: .ASCIZ <CRLF>/RK11 LOGIC TEST 2/<15><12>/MAINDEC-11-CZRKKF/<CRLF>
1884      67$:
1885 003250 012700 001410      MOV     #DDPCH,R0
1886 003254 012701 177764      MOV     #-14,R1
1887 003260 005020 1$:  CLR     (R0)+
1888 003262 005201      INC     R1
1889 003264 001375      BNE     1$
1890 003266 005227 177777      INC     #-1      ;FIRST START ?
1891 003272 001020      BNE     START1    ;BR IF NOT
1892 003274 013746 000004      MOV     ERRVEC,-(SP) ;SAVE ERROR VECTOR ADDRESS
1893 003300 012737 003314 000004      MOV     #2$,ERRVEC ;NEW VECTOR ADDRESS
1894 003306 005737 177776      TST     PS        ;SEE IF PROGRAM CAN REFERENCE THE
1895      ;PROCESSOR STATUS WORD
1896 003312 000406      BR      3$        ;BR IF REFERENCE DIDN'T CAUSE TRAP
1897 003314 012737 000140 001400 2$:  MOV     #140,RKPRI ;SETUP INTERRUPTING PRIORITY TO VALUE
1898      ;WHICH WILL ALLOW INTERRUPT ON AN LSI-11
1899 003322 012716 003330      MOV     #3$, (SP) ;SETUP RETURN ADDRESS
1900 003326 000002      RTI
1901 003330 012637 000004 3$:  MOV     (SP)+,ERRVEC ;RESTORE THE ERROR VECTOR
1902      ;
1903      ;FIND OUT IF ACT11, 'XXDP' CHAIN OR DUMP MODE
1904      ;
1905 003334 012700 001410  START1: MOV     #DDPCH,R0
1906 003340 012701 177766      MOV     #-12,R1   ;CLEAR OUT DRIVE TABLE AREA
1907 003344 005020 1$:  CLR     (R0)+
1908 003346 005201      INC     R1
1909 003350 001375      BNE     1$
1910 003352 122737 000002 000041      CMPB   #2,41     ;LOADED FROM AN RK05 ?
1911 003360 001166      BNE     ST2       ;BR IF NOT
1912 003362 013737 000040 001410      MOV     40,DDPCH ;GET DEVICE INDICATOR AND DRIVE ADDRESS OF
1913      ;LOADING RK05
1914 003370 122737 000010 001410      CMPB   #10,DDPCH ;VALID DRIVE NUMBER IN BYTE 40 ?
1915 003376 101002      BHI     2$        ;BR IF YES
1916 003400 105037 001410      CLRB   DDPCH     ;MUST BE DRIVE ZERO WHICH LOADED
1917      ;THIS PROGRAM
1918 003404 005737 000042 2$:  TST     42        ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
1919 003410 001432      BEQ     4$        ;BR IF NEITHER
1920 003412 005737 001410      TST     DDPCH    ;RUNNING FROM AN RK05 ?
1921 003416 001002      BNE     3$        ;BR IF YES
1922 003420 000137 004262      JMP     ST3       ;FIND OUT NUMBER OF DRIVES
1923 003424 3$:
1924 003424 104401 003432      TYPE   ,65$     ;;TYPE ASCIZ STRING
1925 003430 000413      BR     64$      ;;GET OVER THE ASCIZ
    
```

```

1926
1927 003460
1928 003460 005046
1929 003462 113716 001410
1930 003466 104403
1931 003470 001
1932 003471 000
1933 003472 000137 004262
1934 003476 005227 177777
1935 003502 001115
1936 003504 104401 003512
1937 003510 000411
1938
1939 003534
1940 003534 005046
1941 003536 113716 001410
1942 003542 104403
1943 003544 001
1944 003545 000
1945 003546 104401 003554
1946 003552 000431
1947
1948 003636
1949 003636 104401 003644
1950 003642 000435
1951
1952 003736
1953
1954
1955
1956
1957
1958
1959 003736 012700 001412
1960 003742 012701 177765
1961 003746 005020
1962 003750 005201
1963 003752 001375
1964 003754 104401 003762
1965 003760 000415
1966
1967 004014
1968 004014 104411
1969 004016 012600
1970 004020 012701 177770
1971 004024 112002
1972 004026 042702 177400
1973 004032 012703 001414
1974 004036 012704 177770
1975 004042 012705 000060
1976 004046 020502
1977
1978 004050 001414
1979 004052 005205
1980 004054 005723
1981 004056 005204

```

```

;;65$: .ASCIZ <15><12>/NOT TESTING DRIVE /
64$:
CLR -(SP) ;CLEAR WORD ON STACK
MOVB DDPCH,(SP) ;GET DRIVE ADDRESS
TYPOS ;TYPE THE ADDRESS
.BYTE 1 ;ONLY 1 CHARACTER
.BYTE 0 ;SUPRESS LEADING ZEROS
JMP ST3 ;GET NUMBER OF DRIVES
4$: INC #-1 ;FIRST TIME THROUGH HERE ?
BNE ST2 ;BR IF NOT
TYPE ,67$ ;;TYPE ASCIZ STRING
BR 66$ ;;GET OVER THE ASCIZ
;;67$: .ASCIZ <15><12>/TO TEST DRIVE /
66$:
CLR -(SP) ;CLEAR WORD ON THE STACK
MOVB DDPCH,(SP) ;GET DRIVE ADDRESS
TYPOS ;TYPE THE DRIVE ADDRESS
.BYTE 1 ;ONLY 1 CHARACTER
.BYTE 0 ;SUPRESS LEADING ZEROS
TYPE ,69$ ;;TYPE ASCIZ STRING
BR 68$ ;;GET OVER THE ASCIZ
;;69$: .ASCIZ / HALT PROGRAM, REMOVE RKDP PACK AND REPLACE IT/<15><12>
68$:
TYPE ,71$ ;;TYPE ASCIZ STRING
BR 70$ ;;GET OVER THE ASCIZ
;;71$: .ASCIZ /WITH A WORK PACK, CLEAR LOCATION 40, AND RESTART PROGRAM/
70$:

;FIND OUT FROM USER WHICH DRIVES (LOGICAL ADDRESSES) ARE TO BE
;TESTED (DRIVES TO BE TESTED ?). IN REPLY THE USER SHOULD TYPE IN THE
;LOGICAL ADDRESSES SEPERATED BY COMMAS. THUS IF 2 DRIVES 0,1 ARE PRESENT:
; 'DRIVS TO B TSTD?'
; '0,1<CR>' A CAR. RET. SHOULD BE TYPED TO TERMINATE THE LIST.
ST2: MOV #DRIVS,R0
MOV #-13,R1
13$: CLR (R0)+
INC R1
BNE 13$
TYPE ,65$ ;;TYPE ASCIZ STRING
BR 64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <15><12>/DRIVES TO BE TESTED ?/<15><12>
64$:
RDLIN
MOV (SP)+,R0 ;GET STARTING ADRES OF ASCII STRING
MOV #-10,R1 ;SET UP COUNT
1$: MOVB (R0)+,R2 ;GET ASCII CHARACTER
BIC #177400,R2 ;MASK UNWANTED BITS
MOV #DRIV0,R3
MOV #-10,R4
MOV #60,R5
2$: CMP R5,R2 ;WAS THE TYPED IN CHARACTER
;A NUMBER BETWEEN 0-7?
BEQ 3$ ;YES, BRANCH
INC R5 ;NO, INCREMENT
TST (R3)+ ;INCREMENT POINTER TO DRV FLAG
INC R4 ;CHARACTER THAT WAS INPUT

```



```

2038                                     ;CHECK NUMBER OF DRIVES
2039 004262 012737 177777 001440 ST3:  MOV    #-1,SIZ/ET      ;CHECK FOR RK05F LATER
2040 004270 012737 004442 000004      MOV    #5$,@#4      ;SET UP ADRES FOR TIME-OUT VECTOR
2041 004276 005777 175024              TST    @RKDS        ;REFERENCE RKDS
2042 004302 005777 175032              TST    @RKDA        ;REFERENCE RKDA
2043 004306 012737 004534 000004      MOV    #BADTMO,@#4
2044 004314 104401                      TYPE
2045 004316 001216                      MSG1
2046 004320 012700 177770              MOV    #-10,R0      ;INITIALIZE COUNT FOR THE 8 DRIVES
2047 004324 005037 001412              CLR    DRIVS        ;INITIALIZE # OF DRIVES PRESENT TO 0
2048 004330 005001                      CLR    R1           ;INITIALIZE ADDRESS TO DRIVE 0
2049 004332 005004                      CLR    R4
2050 004334 012702 001414              MOV    #DRIVO,R2
2051 004340 010177 174774              1$:  MOV    R1,@RKDA   ;ADDRESS THE DRIVE
2052 004344 020177 174770              CMP    R1,@RKDA    ;CHECK, WAS IT ADDRESSED?
2053 004350 001405                      BEQ    3$          ;YES
2054 004352 012703 004356              MOV    #2$,R3
2055 004356 004737 021026              2$:  JSR    PC,TYERM ;WHILE CHECKING NUMBER OF DRIVE
2056                                     ;UNDER NON-MANUAL MODE :-
2057                                     ;RKDA HAD TO BE ADRESED BUT
2058                                     ;IT WAS FOUND THAT THE DRIVE NO
2059                                     ;THAT WAS WRITTEN COULD NOT BE READ BACK
2060                                     ;CORRECTLY.
2061
2062 004362 000413                      BR     4$
2063 004364 032777 000200 174734 3$:  BIT    #200,@RKDS  ;CHECK IF 'DRY' BIT IS SET, IF SET DRIVE IS
2064                                     ;PRESENT
2065                                     ;
2065 004372 001407                      BEQ    4$
2066 004374 104401                      TYPE
2067 004376 001213                      $CRLF
2068 004400 005237 001412              INC    DRIVS        ;IF PRESENT, INCREMENT # OF DRIVES
2069 004404 005212                      INC    (R2)         ;SET UP FLAG INDICATING THIS DRIVE PRESENT
2070 004406 010446                      MOV    R4,-(SP)
2071 004410 104402                      TYPOC
2072 004412 005722                      4$:  TST    (R2)+     ;SHIFT POINTER TO NXT DRIVE INDICATOR
2073 004414 062701 020000              ADD    #20000,R1   ;SET UP ADDRESS FOR THE NEXT DRIVE
2074 004420 005204                      INC    R4           ;HAVE U CHECKED FOR ALL 8 DRIVES
2075 004422 005200                      INC    R0
2076 004424 001345                      BNE   1$
2077 004426 005737 001412              TST    DRIVS
2078 004432 001011                      BNE   ST4
2079 004434 104401                      TYPE
2080 004436 001236                      MSG2
2081 004440 000406                      BR     ST4
2082                                     ;GO CHECK THE DRIVE INDEPENDENT
2083                                     ;CONTROLLER LOGIC
2083 004442 011603                      5$:  MOV    (SP),R3   ;GET PC WHERE TIMEOUT OCCURED
2084 004444 022626                      CMP    (SP)+,(SP)+ ;RESTORE STACK
2085 004446 062703 177776              ADD    #-2,R3
2086 004452 004737 021026              JSR    PC,TYERM    ;GO TYPE ERROR MESSAGE
2087                                     ;WHILE CHECKING FOR THE NUMBER OF
2088                                     ;DRIVES IN NON-MANUAL MODE:-
2089                                     ;RKDS AND RKDA HAD TO BE REFERENCED, TIMEOUT
2090                                     ;OCCURED ON REFERENCING.PC IN THE ERROR
2091                                     ;MESSAGE INDICATES WHERE THE TIMEOUT OCCURED.
2092
2093

```



```
2094
2095
2096 004456 005037 001434 ST4: CLR T56FLG
2097 004462 005737 001412 TST DRIVS
2098 004466 001004 BNE 1$
2099 004470 004737 021742 JSR PC,WATIME
2100 004474 000137 020652 JMP SEOP
2101 004500 012737 001414 001354 1$: MOV #DRIVO,DRVPT
2102 004506 005037 001352 CLR DRVDON ;INITIALIZE THE NO. OF DRIVES
2103 ;THAT HAVE BEEN CHECKED
2104 004512 005037 001350 CLR DRIVAD ;INITIALIZE DRIVE ADDRESS TO
2105 ;THE FIRST DRIVE
2106 004516 012737 004534 000004 MOV #BADTMO,@#4 ;SET TIME OUT VECTOR FOR UNEXPECTED
2107 ;TIME OUTS
2108 004524 012777 004600 174650 MOV #BADINT,@RKVEC ;SET UP RK11 INTERRUPT VECTOR FOR
2109 ;UNEXPECTED INTERRUPTS FROM RK11
2110 004532 000465 BR TST1 ;GO TO TEST 1
2111
2112
2113
2114
2115
2116
```

:THIS ROUTINE HANDLES UNEXPECTED TIME OUTS

```
2117 004534 011600 BADTMO: MOV (SP),RO ;SAVE PC WHERE TIME OUT OCCURED
2118 004536 005740 TST -(RO)
2119 004540 022626 CMP (SP)+,(SP)+ ;RESTORE STACK POINTER
2120 004542 104401 004550 TYPE ,65$ ;:TYPE ASCIZ STRING
2121 004546 000407 BR 64$ ;:GET OVER THE ASCIZ
2122 ;:65$: .ASCIZ <15><12>/TIMEOUT,PC=/
2123 64$:
2124 004566 010046 MOV RO,-(SP) ;SET UP FOR TYPING OUT PC
2125 004570 104402 TYPOC ;GO TYPE OUT OCTAL PC
2126 004572 000000 HALT
2127 004574 000137 002636 JMP @#START
2128
2129
2130
```

:THIS ROUTINE HANDLES UNEXPECTED INTERRUPTS FROM RK11
:SW 9 AND 10 FOR LOOPING ON ERROR
:AND LOOPING ON TEST IN WHICH TIMEOUT
:OCCURRED, ARE PROVIDED.

```
2131
2132
2133
2134
2135
2136 004600 011600 BADINT: MOV (SP),RO ;SAVE PC WHERE INTERRUPT OCCURED
2137 004602 005740 TST -(RO)
2138 004604 032777 020000 174326 BIT #20000,@SWR ;INHIBIT ERROR TYPEOUT?
2139 004612 001014 BNE 1$ ;YES, DON'T TYPE OUT
2140 004614 104401 TYPE
2141 004616 001213 $CRLF
2142 004620 104401 TYPE
2143 004622 026004 EM43 ;TYPE 'UNEXPEXED RK11 INTERRUPT'
2144 ;TYPE ' AT PC='
2145 004624 104401 004632 TYPE ,65$ ;:TYPE ASCIZ STRING
2146 004630 000403 BR 64$ ;:GET OVER THE ASCIZ
2147 ;:65$: .ASCIZ /,PC=/
2148 64$:
2149 004640 010046 MOV RO,-(SP) ;SET UP FOR TYPING OUT PC
```

```
2150 004642 104402          TYPOC          ;GO TYPE OCTAL PC WHERE BAD
2151                          ;INTERUPT OCCURED
2152 004644 032777 001000 174266 1$: BIT #1000,@SWR ;LOOP ON ERROR?
2153 004652 001403          BEQ 2$          ;NO, BRANCH
2154 004654 022626          CMP (SP)+,(SP)+ ;YES, REPOSITION STACK
2155 004656 000177 174224          JMP @SLPADR     ;GO TO THE STARTING ADDRESS OF
2156                          ;THE TEST THAT GAVE UNEXPECTED INTERRUPT
2157 004662 032777 040000 174250 2$: BIT #40000,@SWR ;LOOP ON TEST?
2158 004670 001401          BEQ 3$          ;NO, BRANCH
2159 004672 000002          RTI            ;YES, LOOP. GO BACK WHER U INTERRUPTED FROM.
2160 004674 000000          3$: HALT       ;UNEXPECTED INTERRUPT OCCURED AS
2161                          ;INDICATED IN THE TYPE OUT.U CAN LOOP
2162                          ;ON ERROR, TEST,OR INHIBIT TYPEOUT BY
2163                          ;SETTING APPROPRIATE SWITCHES.
2164 004676 000137 002636          JMP @#START     ;GO BACK TO THE START OF THE
2165                          ;PROGRAM. THUS PRESSING CONTINUE
2166                          ;AFTER THE ABOVE HALT WILL
2167                          ;RESTART THE PROGRAM
2168
2169
2170
2171                          ;RESTART AFTER POWER FAIL
2172                          ;THE PROGRAM WOULD RESTART HERE IF POWER CAME BACK AFTER A FALIURE.
2173
2174 004702 004737 021742          PFSTRT: JSR PC,WATIME ;KILL TIME
2175
2176
2177
2178                          ;:*****
2179                          ;*TEST 1 CHECK THAT THE DRIVES THAT ARE NOT SPECIFIED ARE NOT FOUND TO BE PRESENT
2180                          ;*THIS TEST CHECKS THAT THE DRIVES THAT ARE NOT SPECIFIED
2181                          ;*(IN RESPONSE TO "DRIVS TO BE TSTD?") ARE NOT FOUND TO BE PRESENT.
2182                          ;*EVERY DRIVE FROM 0 TO 7 IS ADDRESSED. IF A PARTICULAR DRIVE
2183                          ;*GIVES 'DRY' (IN RKDS), IT IS CHECKED THAT THIS DRIVE
2184                          ;*WAS SPECIFIED BY THE USER, IF IT WAS NOT AN ERROR IS
2185                          ;*REPORTED, GIVING THE DRIVE NUMBER. IT IS LIKELY THAT THE USER
2186                          ;*MAY HAVE FORGOTTEN TO PUT THE DRIVE (THAT IS NOT SPECIFIED) ON
2187                          ;*'LOAD'. IF THIS IS THE CASE THEN PUT THIS DRIVE ON 'LOAD'.
2188                          ;*IF THIS IS NOT THE CASE, THERE IS A GENUINE ERROR. (TWO DIFFERENT
2189                          ;*DRIVE ADDRESSES MAY BE RESULTING IN THE SELECTION OF THE SAME
2190                          ;*PHYSICAL DRIVE.)
2191                          ;:*****
2192 004706 000004          TST1: SCOPE
2193
2194 004710 01270C 001414          MOV #DRIVO,R0  ;INITIALIZE POINTER
2195 004714 005001          CLR R1        ;INITIALIZE DRIVE ADRES 0
2196 004716 005002          CLR R2        ;INITIALIZE DRIVE # 0
2197 004720 005737 001410          1$: TST DDPCH  ;LOADED FROM AN RK05 ?
2198 004724 001403          BEQ 2$        ;B IF NOT
2199 004726 120237 001410          CMPB R2,DDPCH ;LOADED FROM THIS DRIVE ?
2200 004732 001435          BEQ 4$        ;BR IF YES
2201 004734 010177 174400          2$: MOV R1,@RKDA ;ADRES THE DRIVE
2202 004740 105777 174362          TSTB @RKDS    ;DRIVE READY?
2203 004744 100005          BPL 3$        ;NO, THIS DRIVE NOT PRESENT
2204                          ;YES, THIS DRIVE SELECTED
2205 004746 005710          TST @R0       ;WAS THIS DRIVE SPECIFIED BY
```

```
2206                                     ;THE USER?
2207 004750 001026                       BNE      4$      ;YES, OK
2208                                     ;NO, THIS DRIVE # WAS NOT SPECIFIED
2209                                     ;BY THE USER, BUT STILL IS GIVING
2210                                     ;'DRY' WHEN ADRESED. REPORT EROR.
2211 004752 010237 001162               MOV      R2,$REGO ;GET DRIVE #
2212 004756 104116                       ERROR    116      ;THIS DRIVE # WAS NOT SPECIFIED BY
2213                                     ;THE USER, BUT WHEN ADRESED GAVE
2214                                     ;'DRY'. CHECK THAT THIS DRIVE # IF
2215                                     ;PHYSICALLY PRESENT IS ON 'LOAD'. IF
2216                                     ;THIS IS NOT THE CASE, THEN ONE DRIVE
2217                                     ;MAY BE GETTING SELECTED BY TWO DIFFERENT
2218                                     ;LOGICAL ADDRESSES.
2219 004760 005710                       3$:      TST      @R0 ;CHECK THAT THIS DRIVE WAS NOT INDICATED
2220 004762 001421                       BEQ      4$      ;IF IT WAS, & IT IS NOT FOUND TO BE
2221                                     ;PRESENT (DRY CLEAR), REPORT ERROR.
2222 004764 004737 020774               JSR      PC,GT4RG ;GET RKCS, ER, DS, DA
2223 004770 104010                       ERROR    10      ;DRIVE # (AS IN RKDA) WAS INDICATED BY
2224                                     ;THE USER, BUT WAS NOT FOUND TO BE PRESENT.
2225                                     ;CHECK THAT THE ROTARY DRIVE SELECTION
2226                                     ;SWITCH ON THE MODULE IS SET TO THE RIGHT
2227                                     ;DRIVE #.
2228
2229 004772 005010                       CLR      @R0      ;THIS DRIVE IS NOT FOUND TO BE PRESENT
2230                                     ;HENCE DROP IT FROM THE SELECTION TABLE.
2231 004774 010003                       MOV      R0,R3    ;DRIVE ADDR
2232 004776 162703 001414               SUB      #DRIVO,R3 ;MINUS OFFSET FOR TABLE
2233 005002 042703 000003               BIC      #3,R3    ;EVEN DRIVE OF PAIR
2234 005006 062703 001414               ADD      #DRIVO,R3 ;POINT TO EVEN OF PAIR IF RK05 F
2235 005012 042723 100000               BIC      #100000,(R3)+ ;NOT SPECIFIED AS F MODEL
2236 005016 042713 100000               BIC      #100000,(R3) ;SAME
2237 005022 005337 001412               DEC      DRIVS    ;DECREMENT DRIVE COUNT
2238 005026 005202                       4$:      INC      R2    ;INCRMNT DRIVE #
2239 005030 005720                       TST      (R0)+    ;INCRMNT POINTER
2240 005032 062701 020000               ADD      #20000,R1 ;INCRMNT ADRES TO NXT DRIVE
2241 005036 001330                       BNE      1$      ;LUP BAK IF NOT DONE
2242
2243
2244                                     ;THIS PART OF THE PROGRAM IS GOING TO BE REPEATED FOR
2245                                     ;EACH DRIVE PRESENT
2246                                     ;
2247                                     ;'DRIVAD' CONTAINS IN BITS 15,14,13 THE ADDRESS OF THE
2248                                     ;DRIVE BEING CURRENTLY CHECKED.
2249                                     ;
2250 005040                               NUDRV:
2251
2252
2253                                     ;*****
2254                                     ;*TEST 2      FIND OUT NEXT DRIVE TO BE CHECKED
2255                                     ;THIS CODE FINDS OUT THE NEXT DRIVE THAT IS PRESENT AND THEN SETS UP
2256                                     ;THE ADDRESS IN DRIVAD (BITS 13,14,15). THUS THROUGHOUT THE FOLLOWING TESTS
2257                                     ;THE DRIVE TESTED IS THE DRIVE WHOOSE ADDRESS IS IN 'DRIVAD'.
2258                                     ;*****
2259 005040 000004                       TST2:   SCOPE
2260 005042 012737 000001 001206         MOV      #1,$TIMES ;DO 1 ITERATION
2261 005050 012737 000002 001102         MOV      #2,$TSTNM ;RESET POINTER TO THIS TEST
```

```
2262 ;NO. CHANGE THIS (2) IN CASE THE
2263 ;TEST NO. CHANGES
2264 005056 005037 001112 CLR $ERTTL ;CLEAR TOTAL ERROR COUNT
2265 005062 005737 001412 TST DRIVS ;R THERE ANY DRIVES PRESENT?
2266 005066 001002 BNE .+6 ;YES, BRANCH
2267 005070 000137 020652 4$: JMP $EOP ;NO, JMP TO THE END
2268 005074 013701 001354 MOV DRVPTR,R1 ;GET THAT POINTER TO THE NEXT
2269 ;DRIVE FLAG
2270 005100 032721 000001 2$: BIT #BIT0,(R1)+ ;IS THIS DRIVE PRESENT?
2271 005104 001005 BNE 1$ ;YES
2272 005106 062737 020000 001350 6$: ADD #20000,DRIVAD ;FORM NXT DRIVE ADRES
2273 005114 001371 BNE 2$
2274 005116 000764 BR 4$
2275 005120 005737 001410 1$: TST DDPCH ;PROGRAM LOADED FROM AN RK05 ?
2276 005124 001413 BEQ 3$ ;NO, BRANCH
2277 005126 013746 001350 MOV DRIVAD,-(SP) ;PUT TEST DRIVE ADDRESS ON THE STACK
2278 005132 000316 SWAB (SP) ;SETUP TO RIGHT JUSTIFY THE ADDRESS
2279 005134 006216 ASR (SP) ;RIGHT JUSTIFY THE ADDRESS
2280 005136 006216 ASR (SP) ;RIGHT JUSTIFY THE ADDRESS
2281 005140 006216 ASR (SP) ;RIGHT JUSTIFY THE ADDRESS
2282 005142 006216 ASR (SP) ;RIGHT JUSTIFY THE ADDRESS
2283 005144 006216 ASR (SP) ;RIGHT JUSTIFY THE ADDRESS
2284 005146 122637 001410 CMPB (SP)+,DDPCH ;PROGRAM LOADED FROM THIS DRIVE ?
2285 005152 001755 BEQ 6$ ;BR IF YES, DON'T TEST THE DRIVE
2286 005154 010137 001354 3$: MOV R1,DRVPTR ;STORE POINTER TO THE NEXT
2287 ;DRIVE FLAG
2288 005160 104401 001272 TYPE ,MSG4
2289 005164 013746 001350 MOV DRIVAD,-(R6) ;GET THE DRIVE ADDRESS
2290 005170 004737 021200 JSR PC,SHFRT ;GO SHIFT IT TO THE RIGHT
2291 005174 005037 001404 CLR FFLAG
2292 005200 011600 MOV (R6),R0 ;DRIVE NUMBER
2293 005202 104403 TYPOS ;GO TYPE THE OCTAL # FOR THE
2294 ;DRIVE THAT IS BEING CHECKED
2295 005204 001 000 .BYTE 1,0
2296 005206 006300 ASL R0 ;INDEX TO TABLE
2297 005210 005760 001414 TST DRIV0(R0) ;SEE IF F
2298 005214 100006 BPL 5$ ;NO
2299 005216 104401 005224 TYPE ,65$ ;:TYPE ASCIZ STRING
2300 005222 000401 BR 64$ ;:GET OVER THE ASCIZ
2301 ;:65$: .ASCIZ /F/
2302 64$:
2303 005226 005237 001404 INC FFLAG ;SET F FLAG
2304 005232 104401 5$: TYPE
2305 005234 001213 $CRLF ;TYPE CR, LF
2306 ;:*****
2307 ;*TEST 3 CHECK THAT DRIVE IS SUPPLIED WITH POWER-DPL BIT
2308 ;:*****
2309 005236 000004 TST3: SCOPE
2310 005240 104413 CNT.RESET ;GO, DO CONTROL RESET
2311 ;THIS IS A CALL FOR THE 'CNTRL-
2312 ;RESET' ROUTINE. A CONTROL RESET IS
2313 ;ISSUED AND AFTER A CERTAIN TIME
2314 ;IF THE 'CNTRL RDY' DOES NOT SET
2315 ;AN ERROR IS REPORTED. NOTE THAT
2316 ;THE PC IN ERROR MESSAGE IS THE
2317 ;PC WHERE 'CNT.RESET' IS LOCATED.
```

```
2318 ;THIS IS A VERY BASIC ERR& IF IT
2319 ;OCCURS GO BACK TO TEST 10
2320 005242 013700 001326 MOV RKDS,RO
2321 005246 013777 001350 174064 MOV DRIVAD,@RKDA ;ADDRESS THE DRIVE UNDER TEST
2322 005254 005710 TST @RO ;CHECK IF ANY BIT OF RKDS IS SET?
2323 005256 001003 BNE 1$ ;IF SET, BRANCH
2324 005260 011037 001162 MOV @RO,$REGO ;GET RKDS
2325 005264 104004 ERROR 4 ;RKDS ERROR! RKDS IF ADDRESSED
2326 ;CORRECTLY SHOULD BE NON-ZERO
2327 005266 012777 000015 174036 1$: MOV #15,@RKCS ;ISSUE A DRV RESET, IF DRIVE
2328 ;POWER IS LO, DPL WILL SET
2329 005274 005001 CLR R1
2330 005276 032710 010000 2$: BIT #10000,@RO ;IS 'DPL' BIT SET?
2331 005302 001003 BNE 3$ ;DPL IS SET, BRANCH
2332 005304 005201 INC R1 ;WAIT FOR SOME TIME TO
2333 005306 001373 BNE 2$ ;SEE IF DPL WOULD SET
2334 005310 000403 BR 4$-2 ;OK, DPL NOT SET
2335 005312 004737 021002 3$: JSR PC,GT3RG ;GO, GET RKCS, ER, DS
2336 005316 104005 ERROR 5 ;DPL BIT OF RKDS IS SET, CHECK DRIVE POWER
2337
2338
2339 005320 005001 CLR R1
2340 005322 032710 000100 4$: BIT #100,@RO ;DID R/W/S RDY BIT SET?
2341 005326 001010 BNE TST4 ;:YES, EXIT
2342 005330 104417 000011 DELAY ,11 ;TIME DELAY
2343 005334 005201 INC R1 ;WAIT FOR R/W/S RDY
2344 005336 001371 BNE 4$
2345 005340 017737 173762 001162 MOV @RKDS,$REGO ;GET RKDS
2346 005346 104016 ERROR 16 ;R/W/S RDY DID NOT SET AFTER
2347 ;DRIVE RESET. DRIVE RESET WAS DONE
2348 ;TO CHECK 'DPL'BIT . THIS TEST
2349 ;IS NOT FOR CHECKING DRIVE RESET.
2350 ;U MIGHT WANT TO USE THE TEST PROVIDED
2351 ;FOR CHECKING DRIVE RESET.
2352
2353 ;*****
2354 ;*TEST 4 CHECK THAT 'DRIVE UNSAFE' IS CLEAR, 'HDEN' IS SET, 'WPS' IS CLEAR
2355 ;*****
2356 005350 000004 TST4: SCOPE
2357 005352 104413 CNT.RESET ;GO, DO CONTROL RESET
2358 ;THIS IS A CALL FOR THE 'CNTRL-
2359 ;RESET' ROUTINE. A CONTROL RESET IS
2360 ;ISSUED AND AFTER A CERTAIN TIME
2361 ;IF THE 'CNTRL RDY' DOES NOT SET
2362 ;AN ERROR IS REPORTED. NOTE THAT
2363 ;THE PC IN ERROR MESSAGE IS THE
2364 ;PC WHERE 'CNT.RESET' IS LOCATED.
2365 ;THIS IS A VERY BASIC ERR & IF IT
2366 ;OCCURS GO BACK TO TEST 10
2367 005354 013777 001350 173756 MOV DRIVAD,@RKDA ;SET DRIVE ADDRESS
2368 005362 017700 173740 MOV @RKDS,RO ;GET RKDS
2369 005366 032700 002000 BIT #2000,RO ;IS 'DRU' BIT OF RKDS SET?
2370 005372 001403 BEQ 1$ ;NO
2371 005374 004737 021002 JSR PC,GT3RG ;GO, GET RKCS, ER, DS
2372 005400 104006 ERROR 6 ;'DRU' BIT OF RKDS IS SET, CHECK
2373 ;DRIV BY PUTTING RUN/LOAD SW TO LOAD
```

```

2374
2375 005402 032700 004000 1$: BIT #4000,R0 ;THEN BACK TO RUN
2376 005406 001004 BNE 2$ ;IS 'HDEN' BIT SET?
2377 005410 017737 173712 001162 MOV @RKDS,$REGO ;YES, BRANCH
2378 005416 104007 ERROR 7 ;GET RKDS
2379 ;ERROR, 'RKOS' BIT IS NOT SET
2380 005420 032777 000040 173700 2$: BIT #40,@RKDS ;IS 'WPS' CLEAR?
2381 005426 001403 BEQ TST5 ;YES, EXIT
2382 005430 004737 020774 JSR PC,GT4RG ;GET RKCS, ER, DS, DA
2383 005434 104114 ERROR 114 ;'WPS'-WRITE PROTECT STATUS- BIT OF
2384 ;OF RKDS SHOULD BE CLEAR, IF THIS DRIVE
2385 ;IS WRITE ENABLED. CHECK & SEE IF THIS
2386 ;DRIVE IS WRITE ENABLED, IF IT IS NOT,
2387 ;WRITE ENABLE IT.
2388
2389
2390
2391
2392
2393 005436 000004
2394 005440 104413
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404 005442 013777 001350 173670 MOV DRIVAD,@RKDA ;ADDRS THE DRIVE
2405 005450 105777 173652 TSTB @RKDS ;IS 'DRY' SET?
2406 005454 100403 BMI TST6 ;YES, OK
2407 005456 004737 020774 JSR PC,GT4RG ;GO, GET RKCS, ER, DS, DA
2408 005462 104010 ERROR 10 ;'DRY' NOT SET
2409
2410
2411
2412
2413
2414
2415 005464 000004
2416 005466 013777 001350 173644
2417 005474 005001
2418 005476 032777 000400 173622 1$: CLR R1 ;INITIALIZE COUNT FOR TIMING WAIT LOOP
2419 005504 001006 BIT #400,@RKDS ;IS SOK SET?
2420 005506 005201 BNE TST7 ;EXIT
2421 005510 001372 INC R1 ;NO, WAIT
2422 005512 017737 173610 001162 BNE 1$ ;WAITED LONG?
2423 005520 104011 MOV @RKDS,$REGO ;GET RKDS
2424 ;WAITED LONG BUT 'SEC OK' BIT DID NOT
2425 ;SET
2426
2427
2428
2429
    
```

 ;*TEST 5 CHECK THAT 'DRIVE READY' IS SET IN RKDS

```

TST5: SCOPE
CNT.RESET ;GO, DO CONTROL RESET
;THIS IS A CALL FOR THE 'CNTRL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.
;THIS IS A VERY BASIC ERR & IF IT
;OCCURS GO BACK TO TEST 10
    
```

 ;*TEST 6 CHECK THAT 'SOK' BIT CAN SET
 ;* THIS TEST CHECKS THAT WITHIN A CERTAIN TIME
 ;* 'SOK' BIT CAN SET, IF IT DOES NOT AN ERROR IS REPORTED

```

TST6: SCOPE
MOV DRIVAD,@RKDA ;ADDRS THE DRIVE
CLR R1 ;INITIALIZE COUNT FOR TIMING WAIT LOOP
1$: BIT #400,@RKDS ;IS SOK SET?
BNE TST7 ;EXIT
INC R1 ;NO, WAIT
BNE 1$ ;WAITED LONG?
MOV @RKDS,$REGO ;GET RKDS
ERROR 11 ;WAITED LONG BUT 'SEC OK' BIT DID NOT
;SET
    
```

 ;*TEST 7 CHECK THAT 'SECTOR COUNTER' CAN COUNT FROM 0-13

2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485

005522 000004
005524 104413

005526 013777 001350 173604
005534 013700 001326
005540 005037 001356
005544 005005

005546 012704 177764
005552 012703 000001

005556 005037 001360

005562 005237 001356
005566 001440
005570 005237 001360
005574 001441

005576 011001
005600 032701 000400
005604 001771
005606 021001
005610 001362
005612 042701 177760
005616 001357

005620 005204
005622 001447
005624 005205
005626 001431
005630 011002
005632 032702 000400

: * THIS TEST CHECKS THAT THE SECTOR COUNTER CAN COUNT FROM
: * 0-13
: * 1) FIRST, FOR INITIALIZING PURPOSES THERE IS A TIMED LOOP
: * DURING WHICH SECTOR COUNTER SHOULD COUNT DOWN TO 0. IF THIS
: * IS NOT DONE AN ERROR IS REPORTED
: * 2) AFTER A COUNT OF 0 IS REACHED, THE PROGRAM WAITS
: * FOR A CERTAIN TIME, DURING WHICH THE SEC COUNTER
: * IS SAMPLED. IF THE COUNTER DOES NOT CHANGE WITHIN THIS
: * TIME PERIOD AN ERROR IS REPORTED.
: * 3) UPON FINDING THAT THE COUNTER HAS CHANGED, IT IS CHECKED
: * IF IT INCREMENTED CORRECTLY. IF IT DID NOT AN ERROR IS REPORTED
: * 4) IF IT INCREMENTED CORRECTLY, THE PROGRAM AGAIN WAITS IN A
: * LOOP TILL THE COUNTER CHANGES. (STEPS 2,3,4 ARE REPEATED
: * TILL THE COUNTER COUNTS UP TO 13)

TST7: SCOPE
CNT.RESET

MOV DRIVAD,@RKDA
MOV RKDS,R0
CLR INDX1
CLR R5

MOV #-14,R4
MOV #1,R3

1\$: CLR INDX2

INC INDX1
BEQ 6\$
2\$: INC INDX2
BEQ 7\$

MOV @R0,R1
BIT #400,R1
BEQ 2\$
CMP @R0,R1
BNE 1\$
BIC #177760,R1
BNE 1\$

3\$: INC R4
BEQ TST10
4\$: INC R5
BEQ 8\$
MOV @R0,R2
BIT #400,R2

:GO, DO CONTROL RESET
:THIS IS A CALL FOR THE 'CNTRL-
:RESET' ROUTINE. A CONTROL RESET IS
:ISSUED AND AFTER A CERTAIN TIME
:IF THE 'CNTRL RDY' DOES NOT SET
:AN ERROR IS REPORTED. NOTE THAT
:THE PC IN ERROR MESSAGE IS THE
:PC WHERE 'CNT.RESET' IS LOCATED.
:THIS IS A VERY BASIC ERR & IF IT
:OCCURS GO BACK TO TEST 10

:INITIALIZE
: 'COUNT' - TO TIME 'ERROR 35'
:INITIALIZE 'COUNT' - TO TIME
: 'ERROR 36' (WAIT LOOP)
:INITIALIZE 'COUNT' - FOR THE 12 SECTORS.
:R3 CONTAINS THE 'NEXT' COUNT OF SEC-CNTR
:R1 CONTAINS THE 'PREVIOUS' COUNT OF SEC-CNTR
:R2 CONTAINS THE 'PRESENT' COUNT OF SEC-CNTR
:INITIALIZE 'COUNT' - TO TIME
:(WAIT LOOP) 'ERROR 34'
:KEEP TIMING FOR 'ERROR 35'
:BRANCH & REPORT ERROR IF WAITED LONG?
:KEEP TIMING FOR 'ERROR 34'
:BRANCH & REPORT ERROR IF WAITED LONG?

:GET RKDS
:IS 'SOK' SET?
:NO, WAIT FOR IT TO SET
:MAKE SURE THAT 2 CONSECUTIVE
:READINGS OF SEC-CNTR ARE SAME
:YES, MASK OUT NON-SEC CNTR BITS
:IS IT SECTC? 0, IF NOT LOOP BACK &
:WAIT FOR SECTOR 0
:KEEP TRACK OF SECTORS CHECKED
:EXIT, IF ALL SECTORS CHKD
:KEEP TIMING FOR 'ERROR 36'
:BR & REPORT ERROR IF WAITED LONG
:GET RKDS
:IS SOK SET?

```
2486 005636 001772      BEQ      4$      ;NO, WAIT FOR SOK
2487 005640 021002      CMP      @R0,R2  ;MAKE SURE THAT 2 CONSECUTIVE
2488 005642 001370      BNE      4$      ;READINGS OF SEC-CNTR ARE SAME
2489 005644 042702 177760 BIC      #177760,R2 ;MASK NON-SEC-CNTR BITS
2490 005650 020201      CMP      R2,R1  ;HAS SEC CNTR INCREMENTED?
2491 005652 001764      BEQ      4$      ;NO, WAIT FOR IT TO CHANGE
2492 005654 020203      CMP      R2,R3  ;YES, DID IT INCREMENT CORRECTLY?
2493 005656 001023      BNE      9$      ;NO - REPORT ERROR
2494
2495 005660 005203      5$: INC      R3      ;INCREMENT 'NEXT COUNT'
2496 005662 005201      INC      R1      ;INCREMENT 'PREVIOUS COUNT'
2497 005664 005005      CLR      R5      ;INITIALIZE AGAIN FOR TIMING 'ERROR 36'
2498 005666 000754      BR       3$      ;GO & CHECK THE NEXT SECTOR COUNT
2499
2500 005670 010137 001162 6$: MOV      R1,$REGO ;GET 'SEC CNTR'
2501 005674 104012      ERROR   12      ;WAITED LONG, BUT SECTOR COUNTER
2502                                ;DID NOT COUNT TO 0
2503 005676 000421      BR       TST10  ;;EXIT
2504
2505 005700 017737 173422 001162 7$: MOV      @RKDS,$REGO ;GET RKDS
2506 005706 104011      ERROR   11      ;WAITED LONG, BUT 'SOK' BIT DID
2507                                ;NOT SET
2508 005710 000414      BR       TST10  ;;EXIT
2509
2510 005712 010237 001162 8$: MOV      R2,$REGO ;GET SEC CNTR (PRESENT COUNT)
2511 005716 010337 001164      MOV      R3,$REG1 ;GET 'NEXT COUNT'
2512 005722 104013      ERROR   13      ;WAITED LONG, BUT THE SECTOR
2513                                ;COUNTER DID NOT INCREMENT FROM
2514                                ;THE PRESENT COUNT TO THE NEXT COUNT
2515 005724 000406      BR       TST10  ;;EXIT
2516
2517 005726 010337 001162 9$: MOV      R3,$REGO ;GET 'NEXT COUNT' (SEC CNTR SHOULD BE THIS)
2518 005732 010237 001164      MOV      R2,$REG1 ;GET PRESENT COUNT (WHAT SEC CNTR WAS)
2519 005736 104014      ERROR   14      ;SEC CNTR INCREMENTED WRONG, DID
2520                                ;NOT INCREMENT FROM PRESENT COUNT
2521                                ;TO NEXT COUNT
2522 005740 000747      BR       5$
2523 ;
2524
2525 ;:*****
2526 ;*TEST 10      CHECK THAT SC=SA CAN BE GENERATED
2527 ;* THIS TEST CHECKS THAT SC=SA CAN BE GFNERATED FOR
2528 ;* EVERY SECTOR
2529 ;:*****
2530 005742 000004      TST10: SCOPE
2531 005744 104413      CNT.RESET      ;GO, DO CONTROL RESET
2532                                ;THIS IS A CALL FOR THE 'CNTRL-
2533                                ;RESET' ROUTINE. A CONTROL RESET IS
2534                                ;ISSUED AND AFTER A CERTAIN TIME
2535                                ;IF THE 'CNTRL RDY' DOES NOT SET
2536                                ;AN ERROR IS REPORTED. NOTE THAT
2537                                ;THE PC IN ERROR MESSAGE IS THE
2538                                ;PC WHERE 'CNT.RESET' IS LOCATED.
2539                                ;THIS IS A VERY BASIC ERR & IF IT
2540                                ;OCCURS GO BACK TO TEST 10
2541 005746 013704 001350      MOV      DRIVAD,R4
```



```
2542 005752 013700 001326      MOV    RKDS,R0
2543 005756 012703 177764      MOV    #-14,R3      ;INITIALIZE COUNT FOR # OF SECTORS
2544 005762 010477 173352      1$:   MOV    R4,@RKDA  ;ADDRESS THE DRIVE
2545 005766 005005              CLR    R5           ;INITIALIZE COUNT - FOR TIMING ERROR
2546 005770 005205              2$:   INC    R5           ;KEEP TIMING FOR ERROR
2547 005772 001410              BEQ    3$           ;REPORT ERROR IF WAITED LONG
2548 005774 011001              MOV    @R0,R1      ;GET RKDS
2549 005776 032701 000020      BIT    #20,R1      ;IS SC=SA SET?
2550 006002 001772              BEQ    2$           ;NO, WAIT FOR IT
2551 006004 005204              4$:   INC    R4           ;ADDRS THE NEXT SECTOR
2552 006006 005203              INC    R3           ;ARE ALL SECTORS CHECKED FOR SC=SA
2553 006010 001364              BNE    1$           ;NO, GO & CHECK NEXT
2554 006012 000406              BR     TST11       ;:YES, EXIT
2555
2556 006014 110437 001162      3$:   MOVB   R4,$REGO   ;GET SECTOR ADDRESS
2557 006020 010137 001164      MOV    R1,$REG1    ;GET RKDS
2558 006024 104015              ERROR  15          ;COULD NOT GET SC=SA FOR THIS
2559                                ;'SECTOR ADDRESS'
2560 006026 000766              BR     4$           ;GO CHK FOR THE REST
2561
2562                                ;:
2563                                ;*****
2564                                ;*TEST 11      CHECK THAT 'R/W/S RDY' IS SET & 'SIN' IS CLEAR
2565                                ;*****
```

```
TST11:  SCOPE
2566 006032 104413              CNT.RESET          ;GO, DO CONTROL RESET
2567 006034 013777 001350 173276  MOV    DRIVAD,@RKDA ;ADDRESS THE DRIVE
2568 006042 005001              CLR    R1
2569 006044 017700 173256      1$:   MOV    @RKDS,R0    ;GET RKDS
2570 006050 032700 000100      BIT    #100,R0     ;IS R/W/S RDY SET?
2571 006054 001007              BNE    2$           ;YES, BRANCH
2572 006056 005201              3$:   INC    R1           ;INCREASE LOOP TIME
2573 006060 001376              BNE    3$           ;FOR DRIVE RESET OF HEADS
2574 006062 005201              INC    R1           ;WAITED LONG ENOUGH?
2575 006064 001367              BNE    1$           ;IF NOT LUP BAK & WAIT
2576 006066 010037 001162      MOV    R0,$REGO    ;GET RKDS
2577 006072 104016              ERROR  16          ;R/W/S RDY SHOULD BE SET
2578 006074 032700 001000      2$:   BIT    #1000,R0  ;IS SIN CLEAR?
2579 006100 001403              BEQ    TST12       ;:YES, EXIT
2580 006102 004737 020774      JSR    PC,GT4RG    ;GET RKCS,ER,DS,DA
2581 006106 104001              ERROR  1           ;'SIN' SHOULD HAVE BEEN CLEAR
2582                                ;IT WAS NOT CLEAR
2583                                ;NEXT TEST IS GOING TO CHECK
2584                                ;DRIVE RESET, SIN SHOULD BE
2585                                ;CLEARED THEN. IT WILL BE CHECKED
2586                                ;THERE.
```

```
2587
2588                                ;*****
2589                                ;*TEST 12      CHECK 'DRIVE RESET'
2590                                ;*THIS TEST CHECKS THE VERY BASIC DRIVE RESET LOGIC.
2591                                ;*SINCE THE HEADS ARE AT CYLINDER 0 (GOING INTO THIS
2592                                ;*TEST) DRIVE RESET RETRACTS THEM BACK BEYOND CYLINDER 0,
2593                                ;*AFTER WHICH THEY ARE PUSHED FORWARD TO CYLINDER 0 AGAIN.
2594                                ;*IN THE LATER PART OF THIS PROGRAM THERE IS A DRIVE RESET
2595                                ;*TEST WHICH DOES THE RESET FROM LAST CYLINDER.
2596                                ;*****
2597 006110 000004      TST12:  SCOPE
```

```

2598 006112 104413          CNT.RESET          ;GO, DO CONTROL RESET
2599                                     ;THIS IS A CALL FOR THE 'CNTRL-
2600                                     ;RESET' ROUTINE. A CONTROL RESET IS
2601                                     ;ISSUED AND AFTER A CERTAIN TIME
2602                                     ;IF THE 'CNTRL RDY' DOES NOT SET
2603                                     ;AN ERROR IS REPORTED. NOTE THAT
2604                                     ;THE PC IN ERROR MESSAGE IS THE
2605                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
2606                                     ;THIS IS A VERY BASIC ERR & IF IT
2607                                     ;OCCURS GO BACK TO TEST 10
2608 006114 013700 001332      MOV      RKCS,R0
2609 006120 005004              CLR      R4          ;INITIALIZ COUNT - TO TIME ERROR
2610 006122 013777 001350 173210 MOV      DRIVAD,@RKDA ;ADDRESS THE DRIVE
2611 006130 012710 000015      MOV      #15,@R0    ;'DRIVE RESET', GO
2612 006134 104412              CHKCRDY          ;GO CHECK IF CONTROL RDY IS SET
2613                                     ;IF SO, SKIP THE EROR MESSAGE.
2614 006136 104021              ERROR    21          ;CNTRL RDY DID NOT SET AFTER
2615                                     ;SENDING CYL ADDR TO THE DRIV.
2616                                     ;'ADD ACK' SHOULD HAVE COME BACK
2617                                     ;FROM DRIVE, THEREUPON SETTING 'CN RDY'
2618 006140 012705 177776      MOV      #-2,R5      ;SET UP DELAY COUNTER
2619 006144 032777 000100 173154 6$: BIT      #100,@RKDS    ;CHECK FOR R/W/S READY
2620 006152 001402              BEQ      .+6
2621 006154 000137 006176      JMP      3$
2622 006160 005204              INC      R4
2623 006162 001370              BNE     6$
2624 006164 005205              INC      R5
2625 006166 001366              BNE     6$
2626 006170 004737 020774      JSR     PC,GT4RG    ;GO, GET RKCS, ER, DS, DA
2627 006174 104026              ERROR    26          ;R/W/S RDY DID NOT SET AFTER
2628                                     ;DRIVE RESET
2629
2630 006176 032777 001000 173122 3$: BIT      #1000,@RKDS    ;DID SIN SET?
2631 006204 001403              BEQ     5$          ;NO, BRANCH
2632 006206 004737 020774      JSR     PC,GT4RG    ;GO, GET RKCS,ER,DS,DA
2633 006212 104001              ERROR    1          ;SIN SET, AFTER A
2634                                     ;DRIVE RESET.
2635 006214 032710 140000      5$: BIT      #140000,@R0 ;WAS 'ERR' BIT OR 'HE' BIT SET?
2636 006220 001403              BEQ     4$          ;NC
2637 006222 004737 020774      JSR     PC,GT4RG    ;GO, GET RKCS, ER, DS, DA
2638 006226 104022              ERROR    22         ;'ERR' OR 'HE' BIT SET WHILE DOING
2639                                     ;DRIVE RESET
2640 006230 022710 000214      4$: CMP      #214,@R0 ;DOES RKCS STILL CONTAIN THE
2641                                     ;'DRIV RES' BITS
2642 006234 001406              BEQ     TST13       ;:YES, EXIT
2643 006236 012737 000214 001162 MOV      #214,$REG0 ;GET EXPCTD RKCS
2644 006244 011037 001164      MOV      @R0,$REG1 ;GET RKCS, RECVD
2645 006250 104024              ERROR    24         ;NO - RKCS SHOULD CONTAIN THE 'DRIV RES'
2646                                     ;FUNCTION, ERROR IF DIFFERENT.
2647
2648
2649
2650
2651
2652
2653

```

```

*****
*TEST 13      CHECK 'SEEK' TO CYLINDER 0
;* THIS TEST CHECKS THE SEEK LOGIC DOING SEEK TO CYLINDER 0.
;* NOTE THAT SINCE THE HEADS ARE ALREADY ON CYLINDER 0, NO
;* HEAD MOVEMENT IS INVOLVEDN AND THE STRESS IS ON THE BASIC SEEK
;* LOGIC.

```

```
2654  
2655 006252 000004  
2656 006254 104413  
2657  
2658  
2659  
2660  
2661  
2662  
2663  
2664  
2665  
2666 006256 104421  
2667  
2668 006260 013700 001332  
2669 006264 013777 001350 173046  
2670  
2671 006272 012710 000011  
2672 006276 104412  
2673  
2674 006300 104021  
2675  
2676  
2677  
2678 006302 005005  
2679 006304 032777 000100 173014 2$:  
2680 006312 001005  
2681 006314 005205  
2682 006316 001372  
2683 006320 004737 020774  
2684 006324 104026  
2685 006326 032777 001000 172772 3$:  
2686 006334 001403  
2687 006336 004737 020774  
2688 006342 104001  
2689  
2690  
2691  
2692  
2693 006344 032710 140000 6$:  
2694 006350 001403  
2695  
2696 006352 004737 020774  
2697 006356 104022  
2698  
2699 006360 005777 172744 4$:  
2700 006364 001403  
2701 006366 004737 021002  
2702 006372 104023  
2703  
2704 006374 022710 000210 5$:  
2705 006400 001406  
2706 006402 012737 000210 001162  
2707 006410 011037 001164  
2708 006414 104024  
2709
```

TST13: SCOPE
CNT.RESET
TST.SIN
MOV RKCS,RO
MOV DRIVAD,@RKDA
MOV #11,@RO
CHKCRDY
ERROR 21
2\$: CLR R5
BIT #100,@RKDS
BNE 3\$
INC R5
BNE 2\$+2
JSR PC,GT4RG
ERROR 26
3\$: BIT #1000,@RKDS
BEQ 6\$
JSR PC,GT4RG
ERROR 1
6\$: BIT #140000,@RO
BEQ 4\$
4\$: TST @RKER
BEQ 5\$
JSR PC,GT3RG
ERROR 23
5\$: CMP #210,@RO
BEQ TST14
MOV #210,\$REGO
MOV @RO,\$REG1
ERROR 24

;GO, DO CONTROL RESET
;THIS IS A CALL FOR THE 'CNTRL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.
;THIS IS A VERY BASIC ERR & IF IT
;OCCURS GO BACK TO TEST 10
;GO CHECK IF SIN SET. IF SET
;A DO DRIVE RESET TO CLEAR IT
;ADDRESS THE DRIVE
;'SEEK' GO
;GO CHECK IF CONTROL RDY IS SET
;IF SO, SKIP THE EROR MESSAGE.
;'CNTRL RDY' DID NOT SET AFTER SENDING
;CYL ADDR TO THE DRIVE, 'ADD ACK'
;SHOULD HAVE COME BACK FROM THE
;DRIVE, THEREUPON SETTING 'CNTRL RDY'
;DID R/W/S RDY BIT SET?
;YES, BRANCH
;WAITED LONG ENOUGH?
;IF NOT, LUP BAK & WAIT
;GO, GET RKCS, ER, DS, DA
;R/W/S RDY DID NOT SET AFTER SEEK
;DID SIN SET?
;NO, BRANCH
;GO, GET RKCS,ER,DS,DA
;SIN SET ON DOING SEEK
;TO CYL 0 NOTE THIS IS THE
;FIRST TIME THE HEADS HAVE
;BEEN MOVED
;WAS 'ERR' OR 'HE' BIT SET?
;GO, GET RKCS, ER, DS, DA
;'ERR' OR 'HE' BIT SET WHILE DOING 'SEEK'
;WAS ANY BIT IN RKER SET?
;NO
;GO, GET RKCS, ER, DS
;RKER SHOWS AN ERROR BIT, CHECK
;DOES RKCS STILL CONTAIN 'SEEK' FUNCTION
;YES, EXIT
;GET EXPCTD RKCS
;GET RKCS RECVD
;NO, RKCS SHOULD BE STILL CONTAINING
;'SEEK' FUNCTION ERROR - IF IT CHANGED

```
2710  
2711  
2712  
2713  
2714  
2715  
2716  
2717  
2718  
2719  
2720  
2721 006416 000004  
2722 006420 104413  
2723  
2724  
2725  
2726  
2727  
2728  
2729  
2730  
2731  
2732 006422 104421  
2733  
2734 006424 004737 021504  
2735 006430 104026  
2736  
2737 006432 005005  
2738 006434 013777 001350 172676  
2739 006442 052777 000100 172670  
2740 006450 013701 001326  
2741 006454 012777 000011 172650  
2742 006462 032711 000100 1$:  
2743 006466 001405  
2744 006470 005205  
2745 006472 100373  
2746 006474 004737 021002  
2747 006500 104025  
2748  
2749  
2750 006502 004737 021436 2$:  
2751 006506 104016  
2752  
2753  
2754  
2755  
2756  
2757  
2758  
2759  
2760  
2761  
2762  
2763  
2764  
2765
```

```
*****  
:TEST 14 CHECK R/W/S RDY IS CLEAR WHEN HEADS ARE IN MOTION  
:*THIS TEST CHECKS THAT R/W/S DOES GET CLEARED  
:*WHEN THE HEADS ARE IN MOTION. SINCE 'MOVE L' ON  
:*M7700 (RK05) GENERATES THIS SIGNAL, ABSENCE OF  
:*R/W/S RDY-CLEAR COULD MEAN A FAULT ON M7702  
:*WHERE 'MOVE L' IS GENERATED.  
:*NOTE THIS IS THE FIRST TIME HEADS ARE MADE TO MOVE BY SEEKING  
:*TO CYLINDER 2.  
*****  
TST14: SCOPE  
CNT.RESET ;GO, DO CONTROL RESET  
;THIS IS A CALL FOR THE 'CNTRL-  
;RESET' ROUTINE. A CONTROL RESET IS  
;ISSUED AND AFTER A CERTAIN TIME  
;IF THE 'CNTRL RDY' DOES NOT SET  
;AN ERROR IS REPORTED. NOTE THAT  
;THE PC IN ERROR MESSAGE IS THE  
;PC WHERE 'CNT.RESET' IS LOCATED.  
;THIS IS A VERY BASIC ERR & IF IT  
;OCCURS GO BACK TO TEST 10  
;GO CHECK IF SIN IS SET  
;IF SET DO DRV-RESET TO CLR IT  
;MAKE SURE HEADS R ON CYL 0  
;R/W/S RDY DIDN'T SET  
;AFTER THE ABOVE DRV RESET  
  
TST.SIN  
JSR PC,DRESET  
ERROR 26  
  
CLR R5  
MOV DRIVAD,@RKDA  
BIS #100,@RKDA ;SEEK CYLINDER 2  
MOV RKDS,R1  
MOV #11,@RKCS ;SEEK, GO  
BIT #100,@R1 ;DID R/W/S RDY CLR?  
BEQ 2$ ;YES, BRANCH  
INC R5  
BPL 1$  
JSR PC,GT3RG  
ERROR 25 ;R/W/S RDY WAS NOT CLEAR WHEN HEADS  
;WERE SEEKING TO CYLINDER 2  
  
2$: JSR PC,TSTRWS ;GO, WAIT FOR R/W/S RDY TO SET  
ERROR 16 ;R/W/S RDY DID NOT SET AFTER SEEK  
;WAS TRIED TO CYLINDER 2 (ABOVE).  
;NOTE THIS WAS THE FIRST TIME A SEEK  
;WAS TRIED TO A CYLINDER OTHER THAN  
;0.  
  
*****  
:TEST 15 CHECK 'WRITE' FORMAT FUNCTION-CYLINDER 0, SECTOR 0  
:*THIS TEST CHECKS THE LOGIC INVOLVED IN THE WRITE FMT  
:*FUNCTION. ON ISSUING A WRT FMT, THE FOLLOWING IS CHECKED  
:*1) CNTRL RDY WAS CLEARED AS GO WAS SET.  
:*2) CNTRL RDY SETS WITHIN A CERTAIN TIME ON COMPLETION OF FUNCTION
```

```

2766      ;+3) IF 'HE' OR 'ERR' BIT SET?
2767      ;+4) IF RKDA INCREMENTED CORRECTLY FROM 0 TO 1?
2768      ;+5) IF RKWC OVERFLOWED CORRECTLY TO 0?
2769      ;+6) IF RKBA INCREMENTED CORRECTLY BY 2?
2770      ;+7) IF ANY BIT IN RKR SET?
2771      ;+8) IF THE 'WRT FMT' FUNCTION BITS ARE STILL IN THE RKCS?
2772      ;*NOTE THAT ONE WORD '125252' WAS WRITTEN ON SECTOR
2773      ;*0 & IT WILL BE CHECKED IN THE NEXT TESTS.
2774      ;*****
2775      006510 000004      TST15: SCOPE
2776      006512 104413      CNT.RESET
2777
2778      ;GO, DO CONTROL RESET
2779      ;THIS IS A CALL FOR THE 'CNTRL-
2780      ;RESET' ROUTINE. A CONTROL RESET IS
2781      ;ISSUED AND AFTER A CERTAIN TIME
2782      ;IF THE 'CNTRL RDY' DOES NOT SET
2783      ;AN ERROR IS REPORTED. NOTE THAT
2784      ;THE PC IN ERROR MESSAGE IS THE
2785      ;PC WHERE 'CNT.RESET' IS LOCATED.
2786      006514 104421      TST.SIN
2787
2788      006516 012703 033342  MOV      #OUTBUF,R3
2789
2790      ;THIS CODE SETS UP A 256 WORD BUFFER
2791      ;WHICH WILL BE USED TO WRITE 1 SECTOR
2792      ;ON THE DISK
2793      ;1ST WORD      000001
2794      ;2ND WORD      177777  2'S COMPLEMENT
2795      ;3RD WORD      000002  OF ABOVE
2796      ;4TH WORD      177776
2797      ;
2798      ;253RD WORD     000177
2799      ;254TH WORD     177601
2800      ;255TH WORD     000000
2801      ;256TH WORD     125252
2802      006522 012700 000001  MOV      #1,R0      ;SET COUNT
2803
2804      006526 010023      9$:  MOV      R0,(R3)+  ;SET UP DATA WORDS
2805      006530 010013      MOV      R0,(R3)
2806      006532 095423      NEG      (R3)+
2807      006534 005200      INC      R0
2808      006536 022700 000200  CMP      #200,R0    ;DONE?
2809      006542 001371      BNE     9$
2810      006544 005023      CLR      (R3)+      ;SET 255TH WORD TO 0
2811      006546 012713 125252  MOV      #125252,@R3 ;SET 256TH WORD
2812
2813      006552 012703 033342  MOV      #OUTBUF,R3 ;RESET POINTER TO OUTBUF
2814      006556 013701 001332  MOV      RKCS,R1
2815      006562 013702 001336  MOV      RKBA,R2
2816      006566 010312      MOV      R3,@R2      ;FROM HERE-SET UP CURRENT ADDRESS
2817      006570 012777 177400 172536  MOV      #-400,@RKWC ;SET UP WORD COUNT 400 WORDS
2818      006576 013777 001350 172534  MOV      DRIVAD,@RFD ;SET UP DISK ADDR, SECTOR 0, CYLINDER 0
2819      006604 012711 002003  MOV      #2003,@R1  ;WRITE FORMAT, GO
2820
2821      006610 105711      1$:  TSTB   @R1      ;WAS 'CNTRL RDY' CLEARED AS GO WAS SET?

```

```

2822 006612 100003          BPL      2$          :YES, BRANCH
2823 006614 004737 021002  JSR      PC,GT3RG   :GO, GET RKCS, ER, DS
2824 006620 104030          ERROR    30          : 'CNTRL RDY' DIDN'T CLEAR AS GO
2825                                     : WAS SET TO 'WRITE FORMAT'
2826 006622 005000          2$: CLR      R0          :
2827 006624 105711          TSTB    @R1          : WAS 'CNTRL RDY' SET ON COMPLETION OF WRITE?
2828 006526 100411          BMI     3$          :YES, BRANCH
2829 006630 005200          INC     R0          :NO, HAVE U WAITED LONG ENOUGH?
2830 006632 001374          BNE     2$+2        :IF NOT, LOOP BACK & WAIT
2831                                     :IF YES, REPORT ERROR
2832 006634 004737 020774  JSR      PC,GT4RG   :GO, GET RKCS, ER, DS,DA
2833 006640 013737 001350 001202  MOV     DRIVAD,$REG10
2834 006646 104416          BRKDA4
2835                                     :GO TO 'BDA4' & BREAK CONTENTS OF
2836 006650 104031          ERROR    31          :$REG10 INTO DR #,CYL,SUR,SEC BITS
2837                                     : 'CNTRL RDY' DIDN'T SET ON COMPLETION
2838                                     : OF WRITE FORMAT
2839                                     : WRT FMT WAS DONE STARTING AT <DSK-ADRES>
2840 006652 004737 021234  3$: JSR      PC,CHKHE
2841                                     : INDICATED IN EROR MSGE.
2842                                     : GO CHECK IF 'HE' OR 'ERR' BIT SET,
2843 006656 104032          ERROR    32          : IF YES, SAVE RKCS, ER, DS, DA.
2844                                     : RETURN HERE IF ERROR.
2845                                     : 'HE' OR 'ERR' BIT SET WHILE DOING
2846                                     : A WRITE FORMAT
2847 006660 004737 021262  4$: JSR      PC,CHKDA
2848                                     : WRT FMT WAS DONE STARTING AT <DSK-ADRES>
2849 006664 104033          ERROR    33          : INDICATED IN EROR MSGE.
2850                                     : GO CHECK IF RKDA INCREMENTED CORRECTLY
2851 006666 004737 021316  5$: JSR      PC,CHKWC
2852                                     : IF NOT, RETURN HERE.
2853 006672 104034          ERROR    34          : RKDA SHOULD HAVE INCREMENTED BY
2854                                     : 1 SECTOR, IT DID NOT
2855 006674 022712 034342  6$: CMP     #OUTBUF+1000,@R2 : DID RKBA INCREMENT CORRECTLY?
2856 006700 001406          BEQ     7$          :YES, BRANCH
2857 006702 012737 034342 001162  MOV     #OUTBUF+1000,$REG0 : GET EXPCTD RKBA
2858 006710 011237 001164          MOV     @R2,$REG1   : GET ACTUAL RKBA
2859 006714 104035          ERROR    35          : RKBA DIDN'T INCREMENT BY 1000 AFTER
2860                                     : WRITE FORMAT OF 400 WORDS
2861 006716 004737 021342  7$: JSR      PC,CHKER
2862                                     : CHECK IOF ANY BIT IN RKER SET,
2863 006722 104036          ERROR    36          : IF YES RETURN HERE.
2864                                     : RKER BIT SET ON DOING 1 WORD
2865 006724 022711 002202  8$: CMP     #2202,@R1   : DOES RKCS STILL HAVE 'WRT FMT' BITS?
2866 006730 001406          BEQ     TST16       : :YES, EXIT
2867 006732 012737 002202 001162  MOV     #2202,$REG0 : GET EXPCTD RKCS
2868 006740 011137 001164          MOV     @R1,$REG1   : GET ACTUAL RKCS
2869 006744 104024          ERROR    24          : RKCS DIDN'T CONTAIN 'WRT FMT' BITS
2870                                     : AFTER THE FUNCTION WAS COMPLETED
2871                                     :
2872                                     :*****
2873                                     :*TEST 16 CHECK 'READ FORMAT' FUNCTION-CYLINDER 0, SECTOR 0
2874                                     :*THIS TEST CHECKS THE LOGIC INVOLVED IN THE WRITE FMT
2875                                     :*FUNCTION. ON ISSUING A WRT FMT, THE FOLLOWING IS CHECKED
2876                                     :*1) CNTRL RDY WAS CLEARED AS GO WAS SET.
2877                                     :*2) CNTRL RDY SETS WITHIN A CERTAIN TIME ON COMPLETION OF FUNCTION
    
```

2878
2879
2880
2881
2882
2883
2884
2885
2886
2887
2888
2889
2890
2891
2892
2893
2894
2895
2896
2897
2898
2899
2900
2901
2902
2903
2904
2905
2906
2907
2908
2909
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919
2920
2921
2922
2923
2924
2925
2926
2927
2928
2929
2930
2931
2932
2933

006746 000004
006750 005000
006752 104413

006754 104421

006756 013701 001332
006762 013702 001336
006766 012703 033342
006772 010312

006774 012777 177777 172332
007002 013777 001350 172330
007010 012711 002005

007014 105711
007016 100003
007020 004737 021002
007024 104030

007026 005000
007030 105711

007032 100411
007034 005200
007036 001374

007040 004737 020774
007044 013737 001350 001202
007052 104416

007054 104045

:*3) IF 'HE' OR 'ERR' BIT SET?
:*4) IF RKDA INCREMENTED CORRECTLY FROM 0 TO 1?
:*5) IF RKWC OVERFLOWED CORRECTLY TO 0?
:*6) IF RKBA INCREMENTED CORRECTLY BY 2?
:*7) IF ANY BIT IN RKER SET?
:*8) IF THE CORRECT HEADER WAS RECEIVED?
:*9) FOR RK11C, AFTER RD FMT RKDB CONTAINS THE CHECKSUM
:*FOR THAT SECTOR. (125252 IN THIS CASE, BECAUSE THE
:*FIRST WORD IN SEC 0 WAS WRITTEN AS 125252 IN
:*THE PREVIOUS TEST)
:*10) FOR RK11D, AFTER RD FMT RKDB SHOULD CONTAIN
:*A ZERO
:*11) IF THE RD FMT FUNCTION BITS ARE STILL IN
:*THE RKCS?

::*****

TST16: SCOPE
CLR RO
CNT.RESET

:GO, DO CONTROL RESET
:THIS IS A CALL FOR THE 'CNTRL-
:RESET' ROUTINE. A CONTROL RESET IS
:ISSUED AND AFTER A CERTAIN TIME
:IF THE 'CNTRL RDY' DOES NOT SET
:AN ERROR IS REPORTED. NOTE THAT
:THE PC IN ERROR MESSAGE IS THE
:PC WHERE 'CNT.RESET' IS LOCATED.
:THIS IS A VERY BASIC ERR & IF IT
:OCCURS GO BACK TO TEST 10
:GO CHECK IF SIN IS SET
:IF SET, DO DRIVE RESET TO CLR IT

TST.SIN

MOV RKCS,R1
MOV RKBA,R2
MOV #OUTBUF,R3
MOV R3,@R2

MOV #-1,@RKWC
MOV DRIVAD,@RKDA
MOV #2005,@R1

:SETUP ADRS WHERE HEADER WORD IS TO BE
:X-FERRED
:SET UP WORD COUNT
:SET UP DISK ADRS, SECTOR 0, CYLINDER 0
:READ FORMAT, GO

1\$: TSTB @R1
BPL 2\$
JSR PC,GT3RG
ERROR 30

:WAS 'CNTRL RDY' CLEARED AS GO WAS SET?
:YES, BRANCH
:GO, GET RKCS, RKER
:CNTRL RDY DIDN'T CLEAR AS GO WAS
:SET TO 'READ FORMAT'

2\$: CLR RO
TSTB @R1

:WAS 'CNTRL RDY' SET ON COMPLETION OF
:TRANSFER
:YES, BRANCH
:NO, HAVE U WAITED LONG ENOUGH?
:IF NOT, LOOP BACK & WAIT
:IF YES, REPORT ERROR
:GO, GET RKCS, ER, DS,DA

BMI 3\$
INC RO
BNE 2\$+2

JSR PC,GT4RG
MOV DRIVAD,\$REG10
BRKDA4

:GO TO 'BDA4' & BREAK CONTENTS OF
:\$REG10 INTO DR #,CYL,SUR,SEC BITS
:'CNTRL RDY' DIDN'T SET ON COMPLETION
:OF READ FORMAT

ERROR 45

```
2934 ;READ FMT WAS DONE STARTING AT <DSK-ADRES>
2935 ;INDICATED IN EROR MESGE
2936 007056 004737 021234 3$: JSR PC,CHKHE ;CHECK IF 'ERR' OR 'HE' BIT SET, IF
2937 ;YES RETURN HERE.
2938 007062 104046 ERROR 46 ;'HE' OR 'ERR' BIT SET WHILE
2939 ;DOING A 'READ FORMAT'
2940 ;READ FMT WAS DONE STARTING AT <DSK-ADRES>
2941 ;INDICATED IN EROR MESGE
2942 007064 004737 021262 4$: JSR PC,CHKDA ;CHECK IF RKDA INCREMENTED CORRECTLY
2943 ;IF NOT, RETURN HERE.
2944 007070 104040 ERROR 40 ;RKDA SHOULD HAVE INCREMENTED
2945 ;BY 1 SECTOR, IT DID NOT
2946
2947 007072 004737 021316 5$: JSR PC,CHKWC ;CHECK IF RKWC OVERFLOWED TO 0, IF
2948 ;NOT RETURN HERE.
2949 007076 104041 ERROR 41 ;RKWC DID NOT OVERFLOW TO 0
2950 ;AFTER XFER ON READ FORMAT
2951 007100 022712 033344 6$: CMP #OUTBUF+2,@R2 ;DID RKBA INCREMENT TO NXT WORD ADDRS?
2952 007104 001406 BEQ 7$ ;YES, BRANCH
2953 007106 012737 033344 001162 MOV #OUTBUF+2,$REG0 ;GET EXPCTD RKBA
2954 007114 011237 001164 MOV @R2,$REG1 ;GET ACTUAL RKBA
2955 007120 104042 ERROR 42 ;RKBA DIDN'T INCREMENT BY 2 AFTER
2956 ;'READ FORMAT' OF 1 WORD
2957 007122 004737 021342 7$: JSR PC,CHKER ;CHECK IF ANY BIT IN RKER SET, IF
2958 ;YES RETURN HERE.
2959 007126 104036 ERROR 36 ;RKER BIT SET ON DOING
2960 ;1 WORD READ FORMAT
2961 007130 005713 8$: TST @R3 ;DOES OUTBUF CONTAIN THE HEADER
2962 ;WORD-0
2963 007132 001407 BEQ 9$ ;YES, BRANCH
2964 007134 005037 001162 CLR $REG0 ;GET SECTOR NO.
2965 007140 005037 001164 CLR $REG1 ;EXPCTD HEADER
2966 007144 011337 001166 MOV @R3,$REG2 ;GET HEADER RECVD
2967 007150 104043 ERROR 43 ;CORRECT HEADER WORD-0-WAS
2968 ;NOT RECEIVED ON READ FORMAT
2969 007152 022711 002204 9$: CMP #2204,@R1 ;DOES RKCS HAVE THE 'RDFMT' BITS?
2970 007156 001406 BEQ TST17 ;:YES, BRANCH
2971 007160 012737 002204 001162 MOV #2204,$REG0 ;GET EXPCTD RKCS
2972 007166 011137 001164 MOV @R1,$REG1 ;GET ACTUAL RKCS
2973 007172 104024 ERROR 24 ;RKCS DIDN'T CONTAIN 'RD FMT'
2974 ;BITS AFTER FUNCTION WAS
2975 ;COMPLETED
2976
2977
2978
2979
```

```
*****
;*TEST 17 CHECK 'READ' FUNCTION-CYLINDER 0,SECTOR 0
;*THIS IS THE FIRST TIME A PURE READ IS PREFORMED IN THIS
;*TEST SEQUENCE. THE FOLLOWING IS CHECKED
;*1) CNTRL RDY CLEARS AS GO IS SET
;*2) CNTRL RDY SETS WITHIN A CERTAIN TIME ON COMPLETION
;*OF FUNCTION
;*3) IF 'HE' OR 'ERR' BIT SET?
;*4) IF RKDA INCREMENTED CORRECTLY?
;*5) IF RKWC OVERFLOWED TO 0?
;*6) IF RKBA INCREMENTED CORRECTLY?
```

2989


```
3046                                     ;IF NOT RETURN HERE.
3047 007316 104040                       ERROR 40                       ;RKDA DID NOT INCREMENT
3048                                     ;BY 1 (SECTOR)
3049 007320 004737 021316                5$: JSR PC,CHKWC                    ;CHECK IF RKWC OVERFLOWED TO 0,
3050                                     ;IF NOT RETURN HERE.
3051 007324 104041                       ERROR 41                       ;RKWC DID NOT OVERFLOW TO 0,
3052                                     ;AFTER X-FER ON READ
3053 007326 022712 034342                6$: CMP #OUTBUF+1000,@R2              ;DID RKBA INCREMENT CORRECTLY?
3054 007332 001406                       BEQ 7$                               ;YES, BRANCH
3055 007334 012737 034342 001162         MOV #OUTBUF+1000,$REG0              ;GET EXPCTD RKBA
3056 007342 011237 001164                 MOV @R2,$REG1                      ;GET ACTUAL RKBA
3057 007346 104042                       ERROR 42                       ;RKBA DID NOT INCREMENT BY 2
3058                                     ;AFTER 'READ' OF 1 WORD
3059 007350 004737 021342                7$: JSR PC,CHKR                    ;CHECK IF ANY BIT IN RKER SET,
3060                                     ;IF YES RETURN HERE.
3061 007354 104036                       ERROR 36                       ;RKER BIT SET ON DOING 1
3062                                     ;WORD 'READ'
3063 007356 022713 000001                8$: CMP #1,@R3                     ;DOES OUTBUF CONTAIN THE RIGHT
3064                                     ;DATA WORD
3065 007362 001411                       BEQ 9$                               ;YES BRANCH
3066 007364 012737 000001 001162         MOV #1,$REG0                       ;GET EXPCTD DATA WORD
3067 007372 011337 001164                 MOV (R3),$REG1                    ;GET RECD DATA WORD
3068 007376 013737 001350 001166         MOV DRIVAD,$REG2                  ;GET DISK ADRS FROM WHICH READ WAS DONE
3069 007404 104044                       ERROR 44                       ;DID NOT READ THE CORRECT
3070                                     ;DATA WORD--FROM DISK ADRES,
3071
3072                                     ;SEC 0, CYL 0, SUR 0
3073
3074                                     ;AFTER 1 SECTOR READ RKDB CONTAINS
3075                                     ;FOR RK11C
3076                                     ;THE CHECKSUM FOR THAT SECTOR
3077                                     ;FOR RK11D
3078                                     ;THE LAST WORD TRANSFERRED TO MEMORY
3079
3080                                     ;
3081                                     ;IT SO HAPPENS THAT WITH THE SECTOR
3082                                     ;THAT WAS READ, RKDB CONTAINS THE
3083                                     ;SAME INFORMATION FOR BOTH RK11C
3084 007406 022777 125252 171726          9$: CMP #125252,@RKDB              ;DOES RKDB CONTAIN THE EXPCTD WORD?
3085 007414 001407                       BEQ 10$                             ;YES, BRANCH
3086 007416 012737 125252 001162         MOV #125252,$REG0                ;GET EXPCTD RKDB
3087 007424 017737 171712 001164         MOV @RKDB,$REG1                  ;GET RECD RKDB
3088 007432 104037                       ERROR 37                          ;RKDB DOES NOT CONTAIN THE
3089                                     ;EXPCTD WORD AFTER A READ OF SEC 0
3090                                     ;CYL 0
3091 007434 022711 000204                10$: CMP #204,@R1                  ;DOES RKCS HAVE THE 'READ' BITS?
3092 007440 001406                       BEQ 11$                             ;YES, BRANCH
3093 007442 012737 000204 001162         MOV #204,$REG0                   ;GET EXPCTD RKCS
3094 007450 011137 001164                 MOV @R1,$REG1                    ;GET RECD RKCS
3095 007454 104024                       ERROR 24                          ;RKCS DID NOT CONTAIN 'READ'
3096                                     ;FUNCTION BITS AFTER OPERATION
3097                                     ;WAS COMPLETED
3098 007456 104413                11$: CNT.RESET                    ;GO DO CONTROL RESET
3099 007460 005777 171656                 TST @RKDB                        ;DID CONTROL RESET CLEAR RKDB?
3100 007464 001407                       BEQ TST20                          ;YES, EXIT
3101 007466 013737 001342 001164         MOV RKDB,$REG1                   ;GET ADRES OF RKDB
```

3102 007474 017737 171642 001164
3103 007502 104102
3104
3105
3106
3107
3108
3109
3110
3111
3112
3113
3114
3115
3116
3117
3118
3119 007504 000004
3120 007506 013703 001332
3121 007512 012702 177764
3122 007516 013704 001340
3123 007522 013701 001350
3124 007526 010105
3125 007530 005205
3126 007532 012737 007540 001110
3127
3128 007540 104413
3129
3130
3131
3132
3133
3134
3135
3136
3137
3138 007542 104421
3139
3140 007544 005000
3141 007546 010137 033342
3142
3143
3144
3145
3146 007552 012777 033342 171556
3147
3148 007560 012777 177777 171546
3149 007566 010114
3150 007570 012713 002003
3151
3152 007574 105777 171532
3153 007600 100410
3154 007602 005200
3155 007604 001373
3156
3157 007606 004737 020774

MOV @RKDB,\$REG1 ;GET CONTENTS OF RKDB
ERROR 102 ;CONTROL RESET DIDN'T CLR RKDB

;*TEST 20 CHECK 'WRITE FORMAT' -CYLINDER 0, SECTOR 0-13
;*THIS TEST GOES ONE STEP FURTHER & PERFORMS A WRT
;*FMT ON CYLINDER 0 & CHECKS THE FOLLOWING
;*1) IF CNTRL RDY SET WITHIN A CERTAIN TIME ON COMPLETION
;*OF THE FUNCTION
;*2) IF 'HE' OR 'ERR' BIT SET?
;*3) IF THE RKDA INCREMENTS CORRECTLY?
;*4) IF THE RKDB IS CLEAR?
;*WRT FMT IS DONE ONE SECTOR AT A TIME
;*THE FIRST WORD OF EVERY SECTOR IS WRITTEN AS A
;*PSUEDO-HEADER CONSISTING OF DRIVE #, CYLINDER #, SURFACE
;* & SECTOR #. THIS WILL BE READ & CHECKED IN THE FOLLOWING TEST.

TST20: SCOPE
MOV RKCS,R3
MOV #-14,R2 ;SET UP COUNT FOR 12 SECTORS
MOV RKDA,R4
MOV DRIVAD,R1 ;GET DRIVE ADDRESS
MOV R1,R5 ;STORE IT
INC R5
MOV #1\$, \$LPERR ;SET RETURN ADRES FOR LUPING
;ON ERROR (SW 9)
1\$: CNT.RESET ;GO, DO CONTROL RESET
;THIS IS A CALL FOR THE 'CNTRL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.
;THIS IS A VERY BASIC ERR & IF IT
;OCCURS GO BACK TO TEST 10
TST.SIN ;GO CHECK IF SIN IS SET
;IF SET, DO DRIVE RESET TO CLR IT

CLR R0
MOV R1,OUTBUF ;THIS WORD TO BE X-FERRED. FIRST
;WORD OF EACH SECTOR WILL BE THE
;ACTUAL DRIVE-ADDRS CONSISTING OF
;DRIVE NO, CYL ADDR, SURFACE
;SECTOR NO.
MOV #OUTBUF,@RKBA ;ADRS FROM WHICH DATA WORD IS TO
;X-FERRED
MOV #-1,@RKWC ;SET UP WORD COUNT
MOV R1,@R4 ;ADDRS THE DRIVE, CYL 0, & CORRECT SECTOR
MOV #2003,@R3 ;WRITE FORMAT, GO

2\$: TSTB @RKCS ;DID 'CNTRL RDY' SET?
BMI 3\$;YES, BRANCH
INC R0 ;NO, HAVE U WAITED LONG?
BNE 2\$;IF NOT, LOOP BACK & WAIT
;IF YES, REPORT ERROR
JSR PC,GT4RG ;GO, GET RKCS, ER, DS,DA

```
3158 007612 010137 0C1202      MOV      R1,$REG10      ;GET DISK ADRS (UNIT,CYL,SUR,SEC) TO WHICH
3159                                ;WRITE FORMAT WAS DONE
3160 007616 104416      BRKDA4      ;GO TO 'BDA4' & BREAK CONTENTS OF
3161                                ;$REG10 INTO DR #,CYL,SUR,SEC BITS
3162 007620 104031      ERROR 31      ;'CNTRL RDY' DID NOT SET ON COMPLETION
3163                                ;OF 'WRITE FORMAT'
3164                                ;WRT FMT WAS DONE STARTING AT <DSK-ADRES>
3165                                ;INDICATED IN EROR MSGE.
3166 007622 004737 021226      3$: JSR      PC,CHKHE1      ;CHECK IF 'ERR' OR 'HE' BIT IS SET,
3167                                ;IF YES RETURN HERE.
3168 007626 104032      ERROR 32      ;'HE' OR 'ERR' BIT SET WHILE DOING
3169                                ;WRITE FORMAT ON CYLINDER 0,
3170                                ;SECTOR IN ERROR IS AS SHOWN IN
3171                                ;DISK-ADRES BITS 0-3
3172                                ;WRT FMT WAS DONE STARTING AT <DSK-ADRES>
3173                                ;INDICATED IN EROR MSGE.
3174
3175 007630 004737 021270      4$: JSR      PC,CHKDA1      ;CHECK IF RKDA INCREMENTED CORRECTLY?
3176                                ;
3177 007634 104033      ERROR 33      ;RKDA DID NOT INCREMENT CORRECT
3178                                ;AFTER 1 WORD 'WRITE FORMAT' ON
3179                                ;CYLINDER 0, SECTOR IN ERROR IS 1
3180                                ;LESS THAN THAT SHOWN IN EXPCTD RKDA
3181 007636 005777 171500      5$: TST      @RKDB      ;CHECK THAT RKDB DOES CONTAIN A 0
3182                                ;AFTER WRT BECAUSE LAST WORD WRITTEN
3183                                ;WAS SERIALLY SHIFTED OUT TO THE DISK
3184 007642 001406      BEQ      6$      ;YES, BRANCH
3185 007644 005037 001162      CLR      $REG0      ;THIS IS WHAT RKDB SHOULD CONTAIN
3186 007650 017737 171466      MOV      @RKDB,$REG1 ;GET RKDB
3187 007656 104037      ERROR 37      ;RKDB SHOULD BE 0 AFTER WRT SINCE THE
3188                                ;LAST WORD WRITTEN WAS SERIALLY SHIFTED
3189                                ;OUT OF RKDB
3190 007660 005201      6$: INC      R1      ;INCREMENT DRIVE ADRES TO NXT SECTOR
3191 007662 005205      INC      R5
3192 007664 122705 000014      CMPB    #14,R5      ;
3193 007670 001002      BNE     .+6      ;R U GOING TO CHECK THE LAST SECTOR?
3194 007672 062705 000004      ADD     #4,R5      ;IF NOT,BRANCH
3195                                ;IF YES,INCREMENT R5 CORRECTLY TO 'EXPCTD RKDA'
3196 007676 005202      INC      R2      ;AFTER HAVING CHECKED THE LAST SECTOR
3197 007700 001317      BNE     1$      ;HAVE U FORMATTED ALL 12 SECTORS?
3198                                ;IF NOT, BRANCH BACK & LOOP
3199                                ;IF YES, EXIT
```

```
3200 *****
3201 ;*TEST 21      CHECK 'READ FORMAT'-CYLINDER 0, SECTOR 0-13
3202 ;*THIS TEST PERFORMS A RD FMT ON THE 12 SECTORS OF CYLINDER 0
3203 ;*THE FOLLOWING IS CHECKED
3204 ;*1) IF CNTRL RDY SET WITHIN A CERTAIN TIME ON COMPLETION
3205 ;*OF THE FUNCTION
3206 ;*2) IF 'HE' OR 'ERR' BIT SET?
3207 ;*3) IF THE RKDA INCREMENTS CORRECTLY?
3208 ;*4) RKBA INCREMENTED CORRECTLY BY 30 (OCTAL)
3209 ;*5) RKWC OVERFLOWED TO 0 FROM -14 (OCTAL)
3210 ;*6) CORRECT HEADER WAS RECEIVED FROM ALL 12 SECTORS.
3211 ;*7) RKCS STILL CONTAINS THE 'RD FMT' FUNCTION BITS.
3212 ;*IF THERE IS A READ ERROR IN THIS TEST OR ANY
3213 ;*OTHER TESTS THE USER SHOULD MAKE SURE THAT
```



```
3270 ;RKBA SHOULD INCREMENT BY 24 BYTES
3271 ;AT THE END OF X-FER
3272 010032 022777 033372 171276 4$: CMP #OUTBUF+30,@RKBA ;DID RKBA INCREMENT CORRECTLY?
3273 010040 001407 BEQ 5$ ;YES, BRANCH
3274 010042 012737 033372 001162 MOV #OUTBUF+30,$REGO ;GET EXPCTD RKBA
3275 010050 017737 171262 001164 MOV @RKBA,$REG1 ;GET ACTUAL RKBA
3276 010056 104042 ERROR 42 ;RKBA DID NOT INCREMENT CORRECTLY
3277 ;AFTER READ FORMAT OF 12 HEADERS
3278 010060 004737 021316 5$: JSR PC,CHKWC ;GO CHECK IF RKWC OVERFLOWED TO 0
3279 ;IF NOT RETURN HERE.
3280 010064 104041 ERROR 41 ;RKWC DID NOT OVERFLOW TO 0
3281 ;AFTER 'RD FMT' OF 12 HEADERS
3282 ;OF CYLINDER 0
3283 010066 005724 6$: TST (R4)+ ;WAS THE CORRECT HEADER RECIEVED?
3284 010070 001413 BEQ 7$ ;YES, BRANCH
3285 010072 010037 001162 MOV R0,$REGO ;GET SECTOR FOR WHICH THE HEADER
3286 010076 062737 000014 001162 ADD #14,$REGO ;COULD NOT BE READ CORRECT
3287 010104 005037 001164 CLR $REG1 ;EXPCTD HEADER=0, FOR CYL 0
3288 010110 014437 001166 MOV -(R4),$REG2 ;GET WRONG HEADER RECVD
3289 010114 104043 ERROR 43 ;HEADER WAS NOT READ RIGHT FOR
3290 ;SECTOR (AS IN ER MSGE), & CYL 0
3291 010116 005724 TST (R4)+ ;WAS THE CORRECT HEADER RECVD?
3292 010120 005200 7$: INC R0 ;YES, HAVE U CHECKED FOR ALL 12 SECTORS?
3293 010122 001361 BNE 6$ ;IF NOT, LOOP BACK & CHK HDR FRM NXT SECTR
3294
3295 010124 004737 021342 JSR PC,CHKER ;CHECK IF ANY BIT IN RKER IS SET,
3296 ;IF YES, RETURN HERE.
3297 010130 104036 ERROR 36 ;RKER BIT SET ON DOING RD FMT
3298 ;OF CYL 0, SECTORS 0-13
3299 010132 022711 002204 8$: CMP #2204,@R1 ;DOES RKCS STILL CONTAIN FUNCTION BITS?
3300 010136 001406 BEQ TST22 ;:YES, EXIT
3301 010140 012737 002204 001162 MOV #2204,$REGO ;GET EXPCTD RKCS
3302 010146 011137 001164 MOV @R1,$REG1 ;GET ACTUAL RKCS
3303 010152 104024 ERROR 24 ;RKCS DID NOT CONTAIN 'RD FMT'
3304 ;FUNCTION BITS ON COMPETION OF
3305 ;THE FUNCTION
3306
3307
3308
3309
3310 ;*****
3311 ;*TEST 22 CHECK 'READ',CYLINDER 0, SECTORS 0 TO 13
3312 ;*THIS TEST PERFORMS A READ OF ALL THE SECTORS OF CYLINDER 0
3313 ;*& CHECKS THE FOLLOWING
3314 ;*1) CNTRL RDY SETS WITHIN A CERTAIN TIME ON COMPLETION
3315 ;*OF THE FUNCTION
3316 ;*2) IF 'HE' OR 'ERR' BIT SET?
3317 ;*3) IF THE CORRECT PSUEDO-HEADER (FIRST WORD OF EVERY
3318 ;*SECTOR, WRITTEN IN A PREVIOUS TEST) WAS RECEIVED.
3319 ;*4) IF RKDS CONTAINS THE CORRECT WORD.
3320 ;*4) IF RKDA INCREMENTED CORRECTLY.
3321 ;*5) IF REST OF THE (377) WORDS IN EACH SECTOR ARE '0' , NOTE
3322 ;*PREVIOUSLY ONE WORD WAS WRITTEN PER SECTOR.
3323 ;*6) IF RKCS STILL CONTAINS THE 'READ' FUNCTION BITS
3324 ;*7) IF CONTROL RESET CLEARS RKDB.
3325 ;* IF TESTING IS BEING DONE ON A SIMULATOR ONLY LAST SECTOR(13)
3326 ;*IS READ BECAUSE THE SIMULATOR CAN STORE ONLY 1 SECTOR (256 WORDS).
```


Line	Address	Offset	Op	Op	Op	Op	Op	Op	Op	Op
3382										
3383										
3384	010312	020114	4\$:	CMP	R1,(R4)					
3385										
3386										
3387										
3388	010314	001407		BEQ	5\$					
3389	010316	010137 001162		MOV	R1,\$REG0					
3390	010322	011437 001164		MOV	(R4),\$REG1					
3391	010326	010137 001166		MOV	R1,\$REG2					
3392	010332	104044		ERROR	44					
3393										
3394										
3395	010334	004737 021270	5\$:	JSR	PC,CHKDA1					
3396										
3397	010340	104040		ERROR	40					
3398										
3399										
3400										
3401										
3402										
3403										
3404										
3405										
3406										
3407										
3408	010342	012737 177775 001370		MOV	#-3,EFLG1					
3409	010350	012700 033344		MOV	#OUTBUF+2,RO					
3410	010354	012737 177401 001362		MOV	#-377,COUNT					
3411	010362	005710	11\$:	TST	@RO					
3412	010364	001005		BNE	12\$					
3413	010366	005720		TST	(RO)+					
3414	010370	005237 001362		INC	COUNT					
3415	010374	001372		BNE	11\$					
3416	010376	000412		BR	7\$					
3417	010400	005037 001162	12\$:	CLR	\$REG0					
3418	010404	012037 001164		MOV	(RO)+,\$REG1					
3419	010410	010137 001166		MOV	R1,\$REG2					
3420										
3421	010414	104044		ERROR	44					
3422										
3423										
3424										
3425										
3426										
3427										
3428										
3429	010416	005237 001370		INC	EFLG1					
3430	010422	001357		BNE	11\$					
3431										
3432										
3433	010424	005737 001344	7\$:	TST	SIMUL					
3434	010430	001011		BNE	10\$					
3435										
3436										
3437	010432	005201		INC	R1					

; READ WAS DONE STARTING AT <DSK-ADRES>
; INDICATED IN EROR MESGE
; WAS THE DATA WORD RECVD, CORRECT?
; THE FIRST DATA WORD OF EACH SECTOR
; IS AN ADRS WORD COMPRISING OF DRIVE NO,
; CYLINDER ADRS, SUR, SECTOR ADRS
; GET EXPCTD DATA WORD FROM DISK
; GET THE DATA WORD RECVD
; GET DISK ADRES
; DID NOT RECIEVE CORRECT DATA WORD ON
; READ, OF CYLINDER 0, SECTOR AS SHOWN IN 'DSK
; ADRES' OF EXPCTD DATA WORD
; CHECK IF RKDA INCREMENTED CORRECTLY,
; IF NOT RETURN HERE.
; RKDA DID NOT INCREMENT CORRECTLY
; AFTER READ OF 1 WORD, FROM CYL 0
; SEC IN ERROR IS 1 LESS THAN THAT
; SHOWN IN EXPCTD RKDA
; AS A RESULT OF 'WRT FMT' IN A PREVIOUS TEST
; FIRST WORD OF EVERY SECTOR IS NON-
; ZERO (PSUEDO-HDR), REST 377 WORDS
; ARE ALL 0'S.
; CHECK IF THE REST OF THE 377
; WORDS ARE ALL 0'S
; ALLOW ONLY 3 ERRORS
; INITIALIZE PTR TO 2ND WRD IN BUFR
; CHECK 377 WORDS IN THE BUFFER
; IS THIS WRD 0?
; NO, ERROR
; INCRMNT PTR TO NXT WRD
; CHKD ALL 377 WRDS?
; YES, BRANCH
; GET EXPCTD WORD
; GET WORD RECVD
; GET DISK ADRES, ERROR IN THIS
; SECTOR
; DATA ERROR, THE LAST 377 WORDS
; READ FROM EACH SECTOR SHOULD BE 0
; IN A PREVIOUS TEST, FIRST WORD OF
; EVERY SEC (CYL 0) WAS WRITTEN AS A
; PSUEDO-HDR, REST OF THE WORDS IN THE
; SECTR ARE AUTOMATICALLY WRITTEN AS
; 0'S. THIS ERROR MAY MEAN THAT IT
; DIDN'T HAPPEN SO
; ALLOW ONLY 3 DATA ERORS OF THIS KIND
; TESTING ON SIMULATOR?
; YES BRANCH
; IF NOT TESTING ON SIMULATOR GO AHEAD
; & READ ALL 12 SECTORS ON CYL 0
; INCREMENT DRIV-ADRES TO NXT SECTOR


```

3438 010434 005205          INC      R5          ;INCREMENT 'EXPCTD DRIV-ADRES'
3439 010436 122705 000014  CMPB     #14,R5      ;R U GOING TO READ THE LAST SECTOR?
3440 010442 001002          BNE     .+6         ;IF NOT, BRANCH
3441 010444 062705 000004  ADD      #4,R5      ;IF YES, INCREMENT 'EXPCTD RKDA'
3442                                     ;CORRECTLY
3443 010450 005202          INC      R2          ;HAVE U READ ALL 12 SECTORS?
3444 010452 001266          BNE     1$         ;IF NOT LOOP BACK & READ THE
3445                                     ;NXT SECTOR
3446 010454 022713 000204 10$:  CMP      #204,@R3    ;DOES RKCS, STILL HAVE THE 'READ' FUNCTION
3447 010460 001406          BEQ     8$         ;YES, BRANCH
3448 010462 012737 000204 001162  MOV      #204,$REGO  ;GET EXPCTD RKCS
3449 010470 011337 001164  MOV      @R3,$REG1  ;GET RKCS RECD
3450 010474 104024          ERROR   24         ;RKCS SHOULD STILL CONTAIN THE 'READ'
3451                                     ;FUNCTION BITS
3452 010476 104413          8$:  CNT.RESET     ;GO ,DO CONTROL RESET
3453                                     ;THIS IS A CALL FOR THE 'CNTRL-
3454                                     ;RESET' ROUTINE. A CONTROL RESET IS
3455                                     ;ISSUED AND AFTER A CERTAIN TIME
3456                                     ;IF THE 'CNTRL RDY' DOES NOT SET
3457                                     ;AN ERROR IS REPORTED. NOTE THAT
3458                                     ;THE PC IN ERROR MESSAGE IS THE
3459                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
3460                                     ;THIS IS A VERY BASIC ERR & IF IT
3461                                     ;OCCURS GO BACK TO TEST 10
3462 010500 005777 170636  TST      @RKDB      ;DID CNTRL RESET CLEAR RKDB?
3463 010504 001407          BEQ     TST23      ;:YES, EXIT
3464 010506 013737 001342 001162  MOV      RKDB,$REGO ;GET ADRES OF RKDB
3465 010514 017737 170622 001164  MOV      @RKDB,$REG1 ;GET CONTENTS OF RKDB
3466 010522 104102          ERROR   102       ;CONTROL RESET DID NOT
3467                                     ;CLEAR RKDB

```

::*****

```

3470
3471 ;*TEST 23 CHECK 'WRITE FORMAT' OF THE DISK
3472 ;*THIS TEST WRITE FORMATS THE ENTIRE DISK. THE FIRST
3473 ;*WORD OF EVERY SECTOR IS WRITTEN TO BE A PSUEDO-HEADER
3474 ;*CONSISTING OF THE DRIVE #, CYLINDER #, SURFACE & SECTOR #.
3475 ;*1 SECTOR IS WRITTEN AT A TIME. THE WRITING IS DONE
3476 ;*IN THIS ORDER: CYL 0-SUR 0; CYL 0-SUR 1; CYL 1-SUR 0
3477 ;*CYL 1-SUR 1; CYL 2-SUR 0; CYL 2-SUR 1----- CYL 312-SUR 1.
3478 ;*IMPORTANCE OF THIS TEST SHOULD BE REALIZED, THIS IS
3479 ;*THE FIRST TIME EACH & EVERY SECTOR ON THE DISK IS
3480 ;*ACCESSED & WRITTEN ON. THIS IS THE FIRST TIME RKDA
3481 ;*IS BEING MADE TO INCREMENT OVER THE ENTIRE DISK (FROM
3482 ;*000000 TO 014520) IF A 'SIN' OCCURS AT ANY POINT
3483 ;*A DRIVE RESET IS DONE BEFORE DOING WRT FMT FOR THE NEXT
3484 ;*SECTOR. ANY OTHER ERROR IS CLEARED THROUGH A CONTROL RESET.
3485 ;*THE FOLLOWING CHECKING IS DONE AFTER WRITING EACH
3486 ;*CYLINDER.
3487 ;*1. CNTRL RDY SETS WITHIN A CERTAIN TIME ON COMPLETION
3488 ;*OF THE FUNCTION.
3489 ;*2. IF 'SIN' OCCURRED?
3490 ;*3. IF 'HE' OR 'ERR' BIT SET?
3491 ;*4. IF RKDA INCREMENTED CORRECTLY, INCLUDING BOUNDARY
3492 ;*CONDITIONS (SECTOR COUNTER BITS OVERFLOWING INTO SURFACE,
3493 ;*SURFACE BIT OVERFLOWING INTO CYLINDER BITS) AT THE END

```

```

3494                                     ;*OF THIS POINTERS ARE INCREMENTED ADJUSTED, ETC.
3495                                     ;*& 'WRT FMT' ON THE NEXT SECTOR IS DONE.
3496                                     ;:*****
3497 010524 000004 TST23: SCOPE
3498 010526 012737 000001 001206      MOV #1,$TIMES      ;;DO 1 ITERATION
3499 010534 012737 010564 001110      MOV #1$,$LPERR    ;SET RETURN ADRES FOR LUPING
3500                                     ;ON ERROR (SW 9)
3501 010542 005003      CLR R3             ;(R3)=0, SURFACE 0 BEING WRITTEN
3502                                     ;(R3)-1, SURFACE 1 BEING WRITTEN
3503 010544 012704 177465      MOV #-313,R4      ;SET UP COUNT FOR 203 CYLINDERS
3504 010550 012702 177764      MOV #-14,R2       ;SET UP COUNT FOR 12 SECTORS
3505 010554 013701 001350      MOV DRIVAD,R1     ;GET DRIVE ADRES
3506 010560 010105      MOV R1,R5         ;STORE IT
3507 010562 005205      INC R5
3508 010564 104413 1$: CNT.RESET      ;GO, DO CONTROL RESET
3509                                     ;THIS IS A CALL FOR THE 'CNTRL-
3510                                     ;RESET' ROUTINE. A CONTROL RESET IS
3511                                     ;ISSUED AND AFTER A CERTAIN TIME
3512                                     ;IF THE 'CNTRL RDY' DOES NOT SET
3513                                     ;AN ERROR IS REPORTED. NOTE THAT
3514                                     ;THE PC IN ERROR MESSAGE IS THE
3515                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
3516                                     ;THIS IS A VERY BASIC ERR & IF IT
3517                                     ;OCCURS GO BACK TO TEST 10
3518 010566 104421      TST.SIN          ;GO CHECK IF SIN IS SET
3519                                     ;IF SET, DO DRIVE RESET TO CLR IT
3520 010570 005037 001362 7$: CLR COUNT
3521 010574 010137 033342      MOV R1,OUTBUF     ;THIS WORD TO BE WRITTEN. THE FIRST
3522                                     ;WORD OF EACH SECTOR WILL BE THE ACTUAL
3523                                     ;DISK-ADRES, CONSISTING OF THE DRIVE NO.,
3524                                     ;CYL ADRES, SURFACE BIT SECTOR ADRES
3525 010600 012777 033342 170530      MOV #OUTBUF,@RKBA ;ADRES FROM WHICH WORD IS TO B X-FERRED
3526 010606 012777 177777 170520      MOV #-1,@RKWC     ;SET UP WORD COUNT
3527 010614 010177 170520      MOV R1,@RKDA      ;ADRES THE DRIVE, WITH CORRECT CYL
3528                                     ;& SECTOR ADRES
3529 010620 012777 002003 170504      MOV #2003,@RKCS   ;WRITE FORMAT, GO
3530
3531 010626 105777 170500 2$: TSTB @RKCS      ;DID CNTRL RDY SET
3532 010632 100411      BMI 3$           ;YES, BRANCH
3533 010634 005237 001362      INC COUNT         ;NO, HAVE U WAITED LONG ENOUGH?
3534 010640 001372      BNE 2$           ;IF NOT, LOOP BACK & WAIT
3535                                     ;IF YES, REPORT ERROR
3536 010642 004737 020774      JSR PC,GT4RG      ;GO, GET RKCS, ER, DS,DA
3537 010646 010137 001202      MOV R1,$REG10     ;GET DISK ADRES, WHERE ERROR OCCURED
3538 010652 104416      BRKDA4           ;GO TO 'BDA4' & BREAK CONTENTS OF
3539                                     ;$REG10 INTO DR #,CYL,SUR,SEC BITS
3540 010654 104031      ERROR 31        ;CNTRL RDY DID NOT SET ON COMPLETION
3541                                     ;OF 'WRITE FORMAT', ON SECTOR AS
3542                                     ;SHOWN IN <DSK-ADRES>
3543                                     ;WRT FMT WAS DONE STARTING AT <DSK-ADRES>
3544                                     ;INDICATED IN EROR MSGE.
3545 010656 032777 001000 170442 3$: BIT #1000,@RKDS      ;DID SIN BIT SET?
3546 010664 001405      BEQ 4$           ;NO, BRANCH
3547 010666 004737 021002      JSR PC,GT3RG      ;GO, GET RKCS, ER, DS
3548 010672 010137 001170      MOV R1,$REG3     ;GET, DISK-ADRES WHERE ERROR OCCURED
3549 010676 104001      ERROR 1         ;SIN SET WHILE DOING WRT FMT

```

```
3550 ;TO DISK-ADRES (AS IN $REG3)
3551
3552 010700 004737 021226 4$: JSR PC,CHKHE1 ;CHECK IF 'ERR' OR 'HE' BIT IS SET
3553 ;IF YES, RETURN HERE.
3554 010704 104032 ERROR 32 ;HE OR ERR SET WHILE DOING WRITE
3555 ;FORMAT ON SECTOR AS INDICATED IN
3556 ;<DSK-ADRES>
3557 ;WRT FMT WAS DONE STARTING AT <DSK-ADRES>
3558 ;INDICATED IN EROR MSGE.
3559 010706 004737 02127^ 5$: JSR PC,CHKDA1 ;CHECK IF RKDA INCREMENTED CORRECTLY,
3560 ;IF NOT, RETURN HERE.
3561 010712 104033 ERROR 33 ;RKDA DID NOT INCREMENT CORRECTLY
3562 ;AFTER 'WRITE FORMAT' WAS DONE
3563 ;TO THE SECTOR PREVIOUS TO THAT
3564 ;INDICATED IN 'EXPCTD' RKDA
3565 010714 005201 6$: INC R1 ;INCREMENT TO THE NXT SECTOR
3566 010716 005205 INC R5 ;INCREMENT R5, TO WHAT RKDA WILL INCREMENT
3567 010720 022702 177776 CMP #-2,R2 ;R U GOING TO FORMAT THE LAST SECTOR
3568 ;IN THE CYLINDER ?
3569 010724 001002 BNE .+6 ;IF NOT, BRANCH
3570 010726 062705 000004 ADD #4,R5 ;INCREMENT R5 CORRECTLY TO 'EXPCTD RKDA'
3571 010732 005202 INC R2 ;HAVE U FORMATTED ALL 12 SECTORS
3572 ;ON THIS CYLINDER
3573 010734 001313 BNE 1$ ;IF NOT, LOOP BACK & FORMAT THE
3574 ;NEXT SECTOR
3575 ;YES
3576 010736 012702 177764 MOV #-14,R2 ;RESET THE COUNT FOR 12 SECTORS
3577 010742 042701 000037 BIC #37,R1 ;CLEAR THE SEC ADRES BITS
3578 010746 005703 TST R3 ;SURFACE 1?
3579 010750 001006 BNE 8$ ;YES, BRANCH
3580 010752 005203 INC R3 ;NO, SET FLAG
3581 010754 062701 000020 ADD #20,R1 ;INCREMENT TO THE NXT SURFACE
3582 010760 010105 MOV R1,R5 ;THIS IS WHAT RKDA SHGULD
3583 010762 005205 INC R5 ;INCREMENT TO.
3584 010764 000677 BR 1$ ;GO, DO NXT SURFACE
3585 010766 062701 000040 8$: ADD #40,R1 ;INCREMENT TO NXT CYL
3586 010772 010105 MOV R1,R5 ;POSITION FOR
3587 010774 005205 INC R5 ;EXPCTD RKDA
3588 010776 005003 CLR R3
3589 011000 005204 INC R4 ;HAVE U FORMATTED ALL 203 CYLINDERS
3590 011002 001270 BNE 1$ ;IF NOT, LOOP BACK & FORMAT THE
3591 ;NEXT CYLINDER
3592
3593
3594
3595
```

```
::*****
:*TEST 24 CHECK 'READ FORMAT' FOR THE ENTIRE DISK
:*THIS TEST READ FORMATS THE ENTIRE DISK, WHICH WAS WRT
:*FORMATTED IN THE PREVIOUS TEST. THE FOLLOWING CHECKING
:*IS DONE
:*1. CNTRL RDY SETS WITHIN A CERTAIN TIME ON COMPLETION
:*OF FUNCTION
:*2. IF 'SIN' OCCURRED?
:*3. IF 'HE' OR 'ERR' OCCURRED?
:*4. RKDA INCREMENTED CORRECTLY.
:*5. IF THE CORRECT HEADER WAS READ.
```

3605

```
3606 ;*6. IF RKWC OVERFLOWED CORRECTLY.  
3607 ;*12 SECTORS (1 CYLINDER) ARE READ AT A TIME. IF 'SIN'  
3608 ;*OCCURS A DRIVE RESET IS DONE BEFORE READING THE NEXT  
3609 ;*SECTOR. READING IS DONE IN THIS ORDER CYL 0-SUR 0;  
3610 ;*CYL 0-SUR 1; CYL 1-SUR 0; CYL 1-SUR 1; CYL 2-SUR 0;  
3611 ;*CYL 2-SUR 1;-----CYL 312-SUR 1. IF TESTING ON SIMULATOR, ONLY  
3612 ;*THE LAST CYLINDER (312), LAST SECTOR (13), SURFACE 1 IS READ.  
3613 ;*****  
3614 011004 000004 TST24: SCOPE  
3615 011006 012737 000001 001206 MOV #1,$TIMES ;DO 1 ITERATION  
3616 011014 012737 011100 001110 MOV #1$, $LPERR ;SET RETURN ADRES FOR LUPING  
3617 ;ON ERROR (SW 9)  
3618 011022 005037 001356 CLR INDX1 ;INDX1=0, SURFACE 0 BEING READ  
3619 ;INDX1=1, SURFACE 1 BEING READ  
3620 011026 013701 001350 MOV DRIVAD,R1 ;GET DRIVE ADRES  
3621 011032 010102 MOV R1,R2  
3622 011034 005737 001344 TST SIMUL ;TESTING ON SIMULATOR?  
3623 011040 001410 BEQ 12$ ;NO, BRANCH  
3624 011042 052701 014533 BIS #14533,R1 ;SET BITS FOR CYL 312, SEC 13, SUR 1  
3625 ;ON SIMULATOR, CHECK ONLY CYL 312,  
3626 ;SECTOR 13, SURFACE 1  
3627 011046 052702 014540 BIS #14540,R2 ;RKDA SHOULD INCRMNT TO THIS AFTR  
3628 ;RD FMT OF 1 SECTOR  
3629 011052 012737 177777 001370 MOV #-1,EFLG1 ;SET COUNT FOR READING HDR  
3630 ;FROM 1 SECTOR ONLY  
3631 011060 000407 BR 1$  
3632 011062 012705 177465 12$: MOV #-313,R5 ;SET UP COUNT FOR 203 CYLINDERS  
3633 011066 012737 177764 001370 MOV #-14,EFLG1 ;SET COUNT FOR 12 HDRS TO BE  
3634 ;READ FROM EACH CYLINDER  
3635 011074 062702 000020 ADD #20,R2 ;THIS IS WHAT RKDA SHOULD INCREMENT  
3636 ;BY, AFTER 'RD FMT' OF EACH CYLINDER  
3637 011100 104413 1$: CNT.RESET ;GO, DO CONTROL RESET  
3638 ;THIS IS A CALL FOR THE 'CNTRL-  
3639 ;RESET' ROUTINE. A CONTROL RESET IS  
3640 ;ISSUED AND AFTER A CERTAIN TIME  
3641 ;IF THE 'CNTRL RDY' DOES NOT SET  
3642 ;AN ERROR IS REPORTED. NOTE THAT  
3643 ;THE PC IN ERROR MESSAGE IS THE  
3644 ;PC WHERE 'CNT.RESET' IS LOCATED.  
3645 ;THIS IS A VERY BASIC ERR & IF IT  
3646 ;OCCURS GO BACK TO TEST 10  
3647  
3648 011102 104421 TST.SIN ;CHECK IF SIN IS SET  
3649 ;IF SET DO DRV-RESET TO CLR IT  
3650  
3651 011104 012703 033342 11$: MOV #OUTBUF,R3 ;STORE ADRES OF BUFFER  
3652 011110 005037 001360 CLR INDX2  
3653 011114 010377 170216 MOV R3,@RKBA ;ADRES TO WHICH DATA IS TO BE X-FERRED  
3654 ;FROM THE DISK  
3655 011120 013777 001370 170206 MOV EFLG1,@RKWC ;SET UP WORD COUNT FOR 12 HEADERS  
3656 ;TO BE READ OFF EACH CYLINDER  
3657 ;(ONLY 1 FOR SIMULATOR)  
3658 011126 010177 170206 MOV R1,@RKDA ;ADRES THE DRIVE WITH CORRECT  
3659 ;CYLINDER & SECTOR ADRES  
3660 011132 012777 002005 170172 MOV #2005,@RKCS ;READ FORMAT, GO  
3661
```

3662	011140	105777	170166	2\$:	TSTB	@RKCS	:DID CNTR1 RDY SET?
3663	011144	100411			BMI	3\$:YES, BRANCH
3664	011146	005237	001360		INC	INDX2	:NO, HAVE U WAITED LONG ENOUGH?
3665	011152	001372			BNE	2\$:IF NOT, LOOP BACK & WAIT FOR IT
3666							:IF YES, REPORT ERROR
3667	011154	004737	020774		JSR	PC,GT4RG	:GO, GET RKCS, ER, DS,DA
3668	011160	010137	001202		MOV	R1,\$REG10	:GET DRIV-ADRES STARTING WHICH
3669							: 'READ FORMAT' WAS DONE
3670	011164	104416			BRKDA4		:GO TO 'BDA4' & BREAK CONTENTS OF
3671							:\$REG10 INTO DR #,CYL,SUR,SEC BITS
3672	011166	104045			ERROR	45	:CNTRL RDY DID NOT SET AFTER
3673							:READ FORMAT. 'RKDA' IN EROR MSGE
3674							:GIVES THE CONTENTS OF RKDA AT THE
3675							:TIME OF ERROR.
3676							:READ FMT WAS DONE STARTING AT <DSK-ADRES>
3677							:INDICATED IN EROR MSGE.
3678							
3679	011170	032777	001000 170130	3\$:	BIT	#1000,@RKDS	:DID 'SIN' SET?
3680	011176	001405			BEQ	4\$:NO, BRANCH
3681	011200	004737	021002		JSR	PC,GT3RG	:GO, GET RKCS, ER, DS
3682	011204	010137	001170		MOV	R1,\$REG3	:GET DISK-ADRES WHERE 'SIN'
3683							:OCCURED
3684	011210	104001			ERROR	1	:SIN ERROR ON DOING RD FMT
3685							:TO CYL INDICATED IN \$REG3
3686							
3687	011212	004737	021226	4\$:	JSR	PC,CHKHE1	:CHECK IF 'ERR' OR 'HE' BIT IS SET,
3688							:IF YES, RETURN HERE.
3689	011216	104046			ERROR	46	:HE OR ERR WHILE DOING A READ
3690							:FORMAT. 'RKDA' IN EROR MSGE GIVES
3691							:THE CONTENTS OF RKDA AT THE TIME OF ERROR
3692							:READ FMT WAS DONE STARTING AT <DSK-ADRES>
3693							:INDICATED IN EROR MESGE
3694	011220	020277	170114	5\$:	CMP	R2,@RKDA	:DID RKDA INCREMENT CORRECTLY BY 12 SEC
3695	011224	001410			BEQ	6\$	
3696	011226	010237	001202		MOV	R2,\$REG10	:GET EXPCTD RKDA
3697	011232	104415			BRKDAO		:GO TO 'BDAO' & BREAK CONTENTS OF
3698							:\$REG10 INTO DR #,CYL,SUR,SEC BITS
3699	011234	017737	170100 001202		MOV	@RKDA,\$REG10	:GET RECVD RDKA
3700	011242	104416			BRKDA4		:GO TO 'BDA4' & BREAK CONTENTS OF
3701							:\$REG10 INTO DR #,CYL,SUR,SEC BITS
3702	011244	104040			ERROR	40	:RKDA DID NOT INCREMENT BY 12 SECTORS
3703							:AFTER RD FMT WAS DONE. ADRES
3704							:OF CYLINDER IN ERROR CAN BE OBTAINED
3705							:FROM 'EXPCTD' RDDA
3706	011246	013700	001370	6\$:	MOV	EFLG1,R0	:SET UP COUNT FOR 12 HEADERS TO B CHKD
3707							:(ONLY 1, IF SIMULATOR)
3708	011252	010104			MOV	R1,R4	:GET DRIV-ADRES FROM WHERE RDFMT WAS DONE
3709	011254	042704	160037		BIC	#160037,R4	:GET THE CYLINDER ADRES ONLY. (HEADER)
3710	011260	020413		7\$:	CMP	R4,(R3)	:IS THE RECVD HEADER SAME AS EXPCTD?
3711	011262	001412			BEQ	8\$	
3712	011264	010437	001164		MOV	R4,\$REG1	:GET EXPCTD HEADER WORD
3713	011270	011337	001166		MOV	(R3),\$REG2	:GET HEADER WORD RECVD
3714	011274	010037	001162		MOV	R0,\$REG0	
3715	011300	062737	000014 001162		ADD	#14,\$REG0	:GET THE SECTOR (OCTAL NO) WHICH DID
3716							:NOT GIVE THE CORRECT HEADER
3717	011306	104043			ERROR	43	:DID NOT RECIEVE THE CORRECT HEADER

```
3718 ;WORD FROM 'SECTOR' AS INDICATED
3719 ;(NOTE SECTOR # IS OCTAL)
3720 011310 005723 8$: TST (R3)+ ;INCREMENT POINTER TO THE NXT WORD
3721 ;IN MEMORY WHERE THE RECD HDR IS STORED
3722 011312 005200 INC R0 ;HAVE U CHECKED ALL 12 HEADERS?
3723 011314 001361 BNE 7$ ;IF NOT, LOOP BACK & CHK THE NXT.
3724 ;YES, ALL HEADERS FOR THIS CYLINDER
3725 ;CHECKED.
3726 011316 004737 021316 JSR PC,CHKWC ;CHECK IF RKWC OVERFLOWED TO 0, IF
3727 ;NOT RETURN HERE.
3728 011322 104041 ERROR 41 ;RKWC DID NOT OVERFLOW AFTER DOING
3729 ;RDFMT OF 12 SECTORS ON THE CYLINDER
3730 ;NOTE THAT 'RKDA' IS THE INCREMENTED
3731 ;RKDA AFTER THE RDFMT
3732 011324 005737 001344 9$: TST SIMUL ;TSTING ON SIMULATOR?
3733 011330 001031 BNE TST25 ;IF YES, EXIT
3734 ;NO
3735 011332 005737 001356 TST INDX1 ;DOING SURFACE 1
3736 011336 001011 BNE 10$ ;YES, BRANCH
3737 011340 005237 001356 INC INDX1 ;NO
3738 011344 062701 000020 ADD #20,R1 ;INCREMENT DRIV ADRES TO THE NXT SURFACE
3739 011350 010102 MOV R1,R2
3740 011352 062702 000020 ADD #20,R2 ;THIS IS WHAT RKDA SHOULD INCREMENT
3741 ;TO, AFTER READ FMT OF THE CYLINDER
3742 011356 000137 011100 10$: JMP 1$ ;GO RD FMT THE NXT SURFACE
3743 011362 005037 001356 CLR INDX1 ;
3744 011366 042701 000037 BIC #37,R1 ;CLR SEC, SURFACE BITS
3745 011372 062701 000040 ADD #40,R1 ;INCREMENT TO NXT CYL
3746 011376 010102 MOV R1,R2 ;THIS IS WHAT RKDA SHOULD BE
3747 011400 062702 000020 ADD #20,R2 ;AFTER RD FMT OF CYLINDER
3748 011404 005205 INC R5 ;HAVE U DONE ALL CYLINDERS?
3749 011406 001402 BEQ TST25 ;EXIT
3750 011410 000137 011100 JMP 1$ ;IF NOT, LOOP BACK & READ FMT FROM
3751 ;THE NXT CYLINDER
3752
3753
3754
```

```
*****
;*TEST 25 CHECK 'READ' OF THE ENTIRE DISK
;*READ OF THE ENTIRE DISK (ONE WORD PER SECTOR) IS DONE
;*IN THIS TEST. IN A PREVIOUS TEST THE FIRST WORD OF
;*EVERY SECTOR WAS WRITTEN LIKE A PSUEDO-HEADER (DRIVE #,
;*CYLINDER #, SURFACE & SECTOR #). THESE PSUEDO HEADERS
;*WILL BE READ & CHECKED IN THIS TEST, PROVING THAT ANY
;*SECTOR CAN BE ACCESSED AND READ.
;*THE FOLLOWING CHECKING IS DONE
;*1. CNTRL RDY SETS WITHIN A CERTAIN TIME ON COMPLETION
;*OF FUNCTION.
;*2. IF 'SIN' OCCURRED?
;*3. IF 'HE' OR 'ERR' OCCURRED?
;*4. THE CORRECT FIRST WORD FROM EVERY SECTOR
;*WAS RECEIVED. THIS WORD REFLECTS THE ABSOLUTE
;*DISK ADDRESS (DRV #, CYL #, SUR, SEC#) OF THAT SECTOR.
;*5. IF RKDB CONTAINED THE CORRECT WORD.
;*IF 'SIN' OCCURS DRIVE RESET IS DONE BEFORE READING
;*THE NEXT SECTOR. READ IS DONE IN THIS ORDER SEC 0-11
;*CYL 0 SUR 0 -> SEC 0-11 CYL 0 SUR 1 -> SEC 0-11 CYL 1,.....
```

3751
3752
3753
3754
3755
3756
3757
3758
3759
3760
3761
3762
3763
3764
3765
3766
3767
3768
3769
3770
3771
3772
3773

```

3774                                     ;*IF TESTING ON SIMULATOR ONLY LAST CYLINDER (312), LAST
3775                                     ;*SECTOR (13), SURFACE 1 IS READ.
3776                                     ;*****
3777 011414 000004 TST25: SCOPE
3778 011416 012737 000001 001206 MOV #1,$TIMES ;;DO 1 ITERATION
3779 011424 012737 011470 001110 MOV #1$,SLPERR ;SET RETURN ADRES FOR
3780                                     ;LOOPING ON ERROR (SW9)
3781 011432 012703 033342 MOV #OUTBUF,R3
3782 011436 005004 CLR R4 ;FLAG, CLEAR WHEN READING SURFACE 0
3783                                     ;SET WHEN READING SURFACE 1
3784 011440 013701 001350 MOV DRIVAD,R1 ;GET DRIVE ADDRESS
3785 011444 005737 001344 TST SIMUL ;TSTING ON SIMULATOR?
3786 011450 001403 BEQ 10$ ;IF NOT BRANCH
3787 011452 052701 014533 BIS #14533,R1 ;SET ADRES BITS FOR LAST CYL (312)
3788 011456 000404 BR 1$ ;LAST SECTOR (13), SURFACE 1
3789 011460 012700 177764 10$: MOV #-14,R0 ;SET COUNT FOR 12 SECTORS
3790 011464 012705 177465 MOV #-313,R5 ;SET UP COUNT FOR 203 CYLINDERS
3791
3792 011470 104413 1$: CNT.RESET ;GO, DO CONTROL RESET
3793                                     ;THIS IS A CALL FOR THE 'CNTRL-
3794                                     ;RESET' ROUTINE. A CONTROL RESET IS
3795                                     ;ISSUED AND AFTER A CERTAIN TIME
3796                                     ;IF THE 'CNTRL RDY' DOES NOT SET
3797                                     ;AN ERROR IS REPORTED. NOTE THAT
3798                                     ;THE PC IN ERROR MESSAGE IS THE
3799                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
3800                                     ;THIS IS A VERY BASIC ERR & IF IT
3801                                     ;OCCURS GO BACK TO TEST 10
3802 011472 104421 TST.SIN ;GO CHECK SIN, IF SET DO
3803                                     ;DRIVE RESET TO CLR IT
3804 011474 005037 001356 8$: CLR INDX1
3805 011500 010377 167632 MOV R3,@RKBA ;ADRES TO WHICH DATA IS TO B X-FERRED
3806                                     ;FROM THE DISK
3807 011504 012777 177777 167622 MOV #-1,@RKWC ;SET UP WORD COUNT
3808 011512 010177 167622 MOV R1,@RKDA ;ADRES THE DRIVE WITH CORRECT
3809                                     ;CYLINDER & SECTOR ADRES
3810 011516 012777 000005 167606 MOV #5,@RKCS ;READ, GO
3811
3812 011524 105777 167602 2$: TSTB @RKCS ;DID CNTRL RDY SET?
3813 011530 100411 BMI 3$ ;YES, BRANCH
3814 011532 005237 001356 INC INDX1 ;NO, HAVE U WAITED LONG ENOUGH
3815 011536 001372 BNE 2$ ;IF NOT, LOOP BACK & WAIT FOR IT
3816                                     ;IF YES, REPORT ERROR
3817 011540 004737 020774 JSR PC,GT4RG ;GO, GET RKCS, ER, DS,DA
3818 011544 010137 001202 MOV R1,$REG10 ;GET DISK-ADRES WHERE ERROR OCCURED
3819 011550 104416 BRKDA4 ;GO TO 'BDA4' & BREAK CONTENTS OF
3820                                     ;$REG10 INTO DR #,CYL,SUR,SEC BITS
3821 011552 104045 ERROR 45 ;CNTRL RDY DID NOT SET AFTER DOING
3822                                     ;A 1 WORD READ FROM ADRES AS
3823                                     ;INDICATED IN <DISK-ADRES>
3824                                     ;'RKDA' IN EROR MSGE GIVES THE
3825                                     ;CONTENTS OF RKDA AT THE TIME OF ERROR
3826
3827 011554 032777 001000 167544 3$: BIT #1000,@RKDS ;DID 'SIN' SET?
3828 011562 001405 BEQ 4$ ;NO, BRANCH
3829 011564 004737 021002 JSR PC,GT3RG ;GO, GET RKCS, ER, DS

```



```
3886
3887
3888
3889
3890
3891
3892
3893
3894 011722 000004
3895 011724 012737 000005 001206
3896 011732 012703 001372
3897
3898 011736 005037 001356
3899
3900 011742 013700 001332
3901 011746 013701 001326
3902 011752 013702 001330
3903 011756 012737 011764 001110
3904
3905 011764 000240
3906 011766 104413
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916 011770 104421
3917
3918
3919
3920 011772 013704 001350
3921 011776 051304
3922 012000 010477 167334
3923 012004 012710 000011
3924
3925 012010 104412
3926
3927 012012 104021
3928
3929
3930
3931 012014 005005
3932 012016 032711 000100
3933 012022 001005
3934 012024 005205
3935 012026 001373
3936 012030 004737 020774
3937 012034 104026
3938
3939 012036 032711 001000
3940 012042 001403
3941 012044 004737 020774
```

```

*****
*TEST 26 CHECK 'SEEK' FUNCTION, WITH DIFFERENT VELOCITY MODES
* THIS TEST CHECKS SEEK IN DIFFERENT VELOCITY MODES (DIFF <3,
* 3 < DIFF < 31, DIFF > 31). FOR THESE 3 BASIC VELOCITIES SEEK IS DONE BOTH
* IN FWD AND REV DIRECTION TO CHECK THE ADDER & DIFFERENCE LOGIC. IF
* WHILE DOING A SEEK 'SIN' OCCURS, A DRIVE RESET IS DONE TO INITIALIZE
* THE POSITIONING LOGIC
*****
TST26: SCOPE
MOV #5,$TIMES ;DO 5 ITERATIONS
MOV #SEEK0,R3 ;INITIALIZE POINTER TO THE FIRST
;SEEK ADDRESS
CLR INDX1 ;INDX1, WHEN 0 INDICATES SEEK IN FWD DIRECTION
; WHEN 1 INDICATES SEEK IN REV DIRECTION
MOV RKCS,R0
MOV RKDS,R1
MOV RKER,R2
MOV #1$,$LPERR ;SET RETURN ADRES FOR LUPING ON
;EROR (SW 9)
1$: NOP
2$: CNT.RESET ;GO, DO CONTROL RESET
;THIS IS A CALL FOR THE 'CNTRL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.
;THIS IS A VERY BASIC ERR & IF IT
;OCCURS GO BACK TO TEST 10
;GO, CHECK IF SIN IS SET, IF SET
;DO DRV-RESET TO CLEAR IT
TST.SIN
MOV DRIVAD,R4 ;GET DRIV-ADRES
BIS (R3),R4 ;SET CYLINDER BITS
MOV R4,@RKDA ;ADDRS THE DRIVE
MOV #11,@R0 ;SET 'SEEK', 'GO'
CHKCRDY ;GO CHECK IF CONTROL RDY IS SET
;IF SO, SKIP THE EROR MESSAGE.
ERROR 21 ;'CNTRL RDY' DID NOT SET AFTER
;SENDING CYL ADD TO THE DRIV, 'ADD ACK'
;FROM DRIVE SHLD HAVE COME BACK
;THEREUPON SETTING 'CNTRL RDY'
4$: CLR R5
5$: BIT #100,@R1 ;DID R/W/S RDY SET?
BNE 6$ ;YES, BRANCH
INC R5 ;NO, WAIT
BNE 5$ ;WAITED LONG?
JSR PC,GT4RG ;GO, GET RKCS, ER, DS, DA
ERROR 26 ;R/W/S RDY DID NOT SET ON
;COMPLETION OF SEEK
6$: BIT #1000,@R1 ;DID SIN SET?
BEQ 7$ ;NO, BRANCH
JSR PC,GT4RG ;GO, GET RKCS, ER, DS, DA
```

```
3942 012050 104001          ERROR 1          ;SIN SET ON DOING SEEK
3943 012052 032710 140000 7$: BIT #140000,@R0 ;DID 'HE' OR 'ERR' SET?
3944 012056 001403          BEQ 8$          ;YES
3945 012060 004737 020774 JSR PC,GT4RG    ;GO, GET RKCS, ER, DS, DA
3946 012064 104022          ERROR 22        ;'ERR OF 'HE' BIT SET WHEN
3947                                ;SEEKING TO CYL AS INDICATED
3948                                ;IN RKDA
3949
3950 012066 022710 000210 8$: CMP #210,@R0   ;DOES RKCS STILL CONTAIN THE 'SEEK' FNCTION
3951 012072 001406          BEQ 9$          ;YES - EXIT
3952 012074 011037 001164 MOV @R0,$REG1   ;NO, GET RKCS RECVD
3953 012100 012737 000210 001162 MOV #210,$REG0  ;GET EXPCTD RKCS
3954 012106 104024          ERROR 24          ;RKCS SHOULD CONTAIN THE 'SEEK' BITS
3955                                ;IF NOT, ERROR
3956
3957 012110 020477 167224 C : CMP R4,@RKDA ;DID RKDA CHANGE?
3958 012114 001406          BEQ 10$         ;NO
3959 012116 010437 001162 MOV R4,$REG0    ;YES, GET EXPCTD?
3960 012122 017737 167212 001164 MOV @RKDA,$REG1 ;GET RKDA
3961 012130 104027          ERROR 27          ;RKDA CHANGED AFTER DOING SEEK
3962
3963 012132 010477 167202 10$: MOV R4,@RKDA   ;ADRES THE DRIVE,SEC 0
3964 012136 012777 033342 167172 MOV #OUTBUF,@RKBA ;READ ONE HEADER INTO THIS
3965 012144 012777 177777 167162 MOV #-1,@RKWC    ;BUS ADRES
3966 012152 012710 002005 MOV #2005,@R0    ;GO,READ FORMAT
3967 012156 104414          CNT.RDY        ;WAIT FOR CNTRL RDY
3968 012160 021337 033342 CMP (R3),OUTBUF  ;WAS THE CORRECT READE4R READ (FROM
3969 012164 001410          BEQ 11$          ;CYLINDER TO WHICH SEEK WAS DONE BEFORE)
3970 012166 005037 001162 CLR $REG0        ;STORE SEC # FROME WHERE HDR WAS RD (0)
3971 012172 011337 001164 MOV (R3),$REG1   ;GET EXPCTD HEADER
3972 012176 013737 033342 001166 MOV OUTBUF,$REG2 ;GET HDR RECVD
3973 012204 104043          ERROR 43          ;WRONG HDR WAS RECVD FROM CYLINDER (ADRES
3974                                ;IN ER MSGE). NOTE THAT A PURE SEEK WAS
3975                                ;DONE TO THIS CYL BEFORE READING HDR
3976                                ;USING READ FORMAT
3977 012206 005737 001356 11$: TST INDX1        ;SEEK IN REVRSE DIRECTION?
3978 012212 001007          BNE 12$          ;YES, BRANCH
3979 012214 005723          TST (R3)+       ;NO, INCREMENT PTR TO NXT SEEK ADRES
3980 012216 022703 001400 CMP #SEEK2+2,R3 ;DONE WITH ALL SKS IN FWD DIR?
3981 012222 001260          BNE 1$           ;NO, GO & DO NXT ONE
3982 012224 005237 001356 INC INDX1        ;SET FLAG INDICATING SK IN REVRSE
3983 012230 005743          TST -(R3)       ;
3984 012232 005743 12$: TST -(R3)        ;POSITION PTR TO NXT SK IN REV
3985 012234 022703 001370 CMP #SEEK0-2,R3 ;DONE WITH ALL?
3986 012240 001251          BNE 1$           ;IF NOT, DO NXT ONE
```

```
3987
3988
3989
3990 ;*****
3991 ;*TEST 27 CHECK DRIVE RESET FROM LAST CYLINDER
3992 ;*THE HEADS ARE POSITIONED ON THE LAST CYLINDER (DOING
3993 ;*AN IMPLIED SEEK-READ). THEN A DRIVE RESET IS ISSUED.
3994 ;*IT'S CHECKED IF THE HEADS WERE BROUGHT BACK TO 0 BY
3995 ;*DOING A 1 WORD READ & CHECKING THAT THE CORRECT WORD
3996 ;*WAS RECEIVED. IF TESTING ON SIMULATOR THIS TEST IS SKIPPED.
3997 ;*****
```

3998	012242	000004			TST27:	SCOPE				
3999	012244	012737	000005	001206		MOV	#5,\$TIMES		::DO 5 ITERATIONS	
4000	012252	005737	001344			TST	SIMUL		:R U ON A SIMULATOR?	
4001	012256	001124				BNE	TST30		::YES, EXIT	
4002	012260	013701	001332			MOV	RKCS,R1			
4003	012264	104413				CNT.RESET			:GO, DO CONTROL RESET	
4004									:THIS IS A CALL FOR THE 'CNTAL-	
4005									:RESET' ROUTINE. A CONTROL RESET IS	
4006									:ISSUED AND AFTER A CERTAIN TIME	
4007									:IF THE 'CNTRL RDY' DOES NOT SET	
4008									:AN ERROR IS REPORTED. NOTE THAT	
4009									:THE PC IN ERROR MESSAGE IS THE	
4010									:PC WHERE 'CNT.RESET' IS LOCATED.	
4011									:THIS IS A VERY BASIC ERR & IF IT	
4012									:OCCURS GO BACK TO TEST 10	
4013	012266	005000				CLR	RO			
4014	012270	012703	033342			MOV	#OUTBUF,R3		:ADRES WHERE DATA WILL BE READ INTO	
4015	012274	013704	001350			MOV	DRIVAD,R4			
4016	012300	010405				MOV	R4,R5			
4017	012302	052705	014500			BIS	#14500,R5		:SET CYL ADRES=312 (OCTAL)	
4018	012306	010577	167026			MOV	R5,@RKDA		:ADRES THE DRIVE, LAST CYLINDER	
4019	012312	012777	177777	167014		MOV	#-1,@RKWC		:READ 1 WORD	
4020	012320	010377	167012			MOV	R3,@RKBA		:INTO THIS MEMORY ADRES	
4021										
4022	012324	012711	000005			MOV	#5,@R1		:READ, GO	
4023										
4024	012330	005000				CLR	RO			
4025	012332	104414			1\$:	CNT.RDY			:THIS IS A CALL FOR CN.RDY ROUTINE	
4026									:WHICH WAITS FOR CNTRL RDY TO SET.	
4027									:A RETURN IS MADE AFTER CNTRL RDY	
4028									:SETS. IF WITHIN A CERTAIN TIME	
4029									:CNTRL RDY DOESN'T SET AN ERROR	
4030									:MESSAGE IS GIVEN. WAITING TIME	
4031									:883 MS FOR 11/20, 175 MS FOR 11/45	
4032	012334	020513			2\$:	CMP	R5,@R3		:WAS THE CORRECT WORD READ?	
4033	012336	001407				BEQ	3\$:YES, SEEK TO 312 WAS DONE CORRECTLY5,a	
4034	012340	010537	001162			MOV	R5,\$REGO		:GET EXPCTD WORD	
4035	012344	011337	001164			MOV	@R3,\$REG1		:GET WORD RECVD	
4036	012350	010537	001166			MOV	R5,\$REG2		:GET DSK-ADRES FROM WHERE WORD WAS READ	
4037	012354	104044				ERROR	44		:DID NOT READ BACK CORRECT WORD FROM	
4038									:LAST CYL, SEC 0. IF TEST 45 & 46	
4039									:WERE SUCCESSFULLY DONE THIS	
4040									:ERROR MEANS THAT IMPLIED SEEK	
4041									:TO CYL 312 COULD NOT B DONE	
4042	012356	012711	000015		3\$:	MOV	#15,@R1		:DRIVE RESET, GO	
4043	012362	104414				CNT.RDY			:THIS IS A CALL FOR CN.RDY ROUTINE	
4044									:WHICH WAITS FOR CNTRL RDY TO SET.	
4045									:A RETURN IS MADE AFTER CNTRL RDY	
4046									:SETS. IF WITHIN A CERTAIN TIME	
4047									:CNTRL RDY DOESN'T SET AN ERROR	
4048									:MESSAGE IS GIVEN. WAITING TIME	
4049									:883 MS FOR 11/20, 175 MS FOR 11/45	
4050	012364	005000				CLR	RO			
4051	012366	032777	000100	166732	4\$:	BIT	#100,@RKDS		:DID R/W/S RDY SET?	
4052	012374	001011				BNE	5\$:YES, BRANCH	
4053	012376	012702	177763			MOV	#-15,R2		:IF U R ON A SLOWER MACHINE	

4110
4111
4112
4113
4114
4115
4116
4117
4118
4119
4120
4121
4122
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133
4134
4135
4136
4137
4138
4139
4140
4141
4142
4143
4144
4145
4146
4147
4148
4149
4150
4151
4152
4153
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165

012530 000004
012532 104413

012534 104421

012536 013704 001332

012542 012700 033342
012546 012701 177401
012552 012702 177400
012556 012703 177400

012562 010320
012564 005202
012566 060103
012570 010320
012572 005202
012574 001374

012576 012777 177400 166530
012604 012777 033342 166524
012612 013777 001350 166520

012620 012714 000003

012624 105714

```
*****  
*TEST 30 'WRITE' - 256 WORD BLOCK ON SECTOR 0, CYLINDER 0  
:THE TEST BELOW SHOULD BE CONSIDERED AS A SET UP PHASE FOR  
:THE FOLLOWING TEST. IT WRITES A BLOCK OF 256 WORDS IN  
:SECTOR 0, CYLINDER 0 WITH A SPECIFIC PATTERN AND THIS WRITTEN  
:BLOCK WILL BE MADE USE OF IN THE NEXT TEST TO CHECK  
:OUT 'WRITE-CHECK' AND 'READ CHECK' FUNCTIONS.  
*****  
TST30: SCOPE  
CNT.RESET ;GO, DO CONTROL RESET  
;THIS IS A CALL FOR THE 'CNTRL-  
;RESET' ROUTINE. A CONTROL RESET IS  
;ISSUED AND AFTER A CERTAIN TIME  
;IF THE 'CNTRL RDY' DOES NOT SET  
;AN ERROR IS REPORTED. NOTE THAT  
;THE PC IN ERROR MESSAGE IS THE  
;PC WHERE 'CNT.RESET' IS LOCATED.  
;THIS IS A VERY BASIC ERR& IF IT  
;OCCURS GO BACK TO TEST 10  
;CHECK IF SIN IS SET, IF SET  
;DO DRIVE RESET TO CLEAR IT  
  
TST.SIN  
  
MOV RKCS,R4  
;THE FOLLOWING CODE IS FOR SETTING  
;UP THE I/O BUFFER IN MEMORY (STARTING AT  
;OUTBUF), WITH A PARTICULAR 256 WORD PATTERN.  
;STARTING FROM THE FIRST WORD IN THE BUFFER  
;THE LO BYTE WILL BE A COUNT PATTERN  
;FROM 0 TO 255 (DECIMAL), WHEREAS THE  
;HI-BYTE WILL BE THE COMPLEMENT OF LO BYTE,  
;A DECREASING COUNT PATTERN FROM 255 TO 0.  
;I.E.THE BUFFER WILL LOOK LIKE:  
;OUTBUF (1 111 111 1 00 000 000)  
;OUTBUF+2 (1 111 111 0 00 000 001)  
;:  
;LAST WORD (0 000 000 0 11 111 111)  
  
MOV #OUTBUF,R0  
MOV #177401,R1 ;PATTERN GENERATING NUMBER  
MOV #-400,R2 ;SET UP COUNT FOR 256 WORDS  
MOV #177400,R3 ;SET UP THE FIRST PATTERN TO B WRITTEN  
  
MOV R3,(R0)+ ;SET UP FIRST WORD IN I/O BUFFER  
INC R2 ;INCREMENT COUNT  
1$: ADD R1,R3 ;SET UP NEXT WORD PATTERN  
MOV R3,(R0)+ ;WRITE IT IN NXT I/O BUFFER WORD  
INC R2 ;HAVE U WRITTEN ALL 256 WORDS  
BNE 1$ ;IF NOT GO & WRITE NEXT PATTERN  
  
MOV #-400,@RKWC ;WRITE 256 WORDS  
MOV #OUTBUF,@RKBA ;STARTING FROM THIS BUS ADRES  
MOV DRIVAD,@RKDA ;TO THIS DISK ADRES, CYL 0, SEC 0  
  
MOV #3,@R4 ;WRITE, GO  
  
2$: TSTB @R4 ;WAS CNTRL RDY CLEARED AS GO WAS SET?
```


4222
4223
4224
4225
4226
4227
4228
4229
4230
4231
4232
4233 012764 000004
4234 012766 104413
4235
4236
4237
4238
4239
4240
4241
4242
4243
4244 012770 104421
4245
4246 012772 012700 177400
4247 012776 012701 033342
4248 013002 005021
4249 013004 005200
4250 013006 001375
4251 013010 005000
4252 013012 012777 177400 166314
4253 013020 012777 033342 166310
4254 013026 013777 001350 166304
4255
4256 013034 012777 000005 166270
4257
4258 013042 105777 166264
4259 013046 100411
4260 013050 005200
4261 013052 001373
4262
4263 013054 004737 020774
4264 013060 013737 001350 001202
4265 013066 104416
4266
4267 013070 104045
4268
4269
4270
4271
4272
4273 013072 032777 001000 166226
4274 013100 001033
4275 013102 012701 177400
4276 013106 012702 177777
4277 013112 012703 033342

```
*****  
: *TEST 31 CHECK THAT WRITE WAS DONE CORRECTLY  
: *THIS TEST CHECKS IF THE 'WRITE' OF 256 WORDS DONE IN PREVIOUS  
: *TEST IS GOOD. THE SEQUENCE OF OPERATIONS IS AS FOLLOWING:  
: *1) DO A READ OF 256 WORDS FROM SECTOR 0, CYLINDER 0  
: * INTO A BUFFER STARTING AT 'OUTBUF'.  
: *2) COMPARE & CHECK THE DATA THAT IS READ (STARTING AT 'OUTBUF')  
: * WITH THE DATA THAT WAS GENERATED PREVIOUSLY  
: *3) REPORT AN ERROR IF THE DATA READ BACK FROM DISK DOES  
: * NOT COMPARE WITH DATA THAT WAS SUPPOSE TO HAVE BEEN WRITTEN  
*****
```

```
TST31: SCOPE  
CNT.RESET  
;GO, DO CONTROL RESET  
;THIS IS A CALL FOR THE 'CNTRL-  
;RESET' ROUTINE. A CONTROL RESET IS  
;ISSUED AND AFTER A CERTAIN TIME  
;IF THE 'CNTRL RDY' DOES NOT SET  
;AN ERROR IS REPORTED. NOTE THAT  
;THE PC IN ERROR MESSAGE IS THE  
;PC WHERE 'CNT.RESET' IS LOCATED.  
;THIS IS A VERY BASIC ERR& IF IT  
;OCCURS GO BACK TO TEST 10  
;CHECK IF SIN IS SET, IF SET  
;DO DRIVE RESET TO CLEAR IT  
;SET COUNT FOR 400 WORDS  
;TO BE CLEARED IN THE BUFFER  
;CLR THE 400 WORD BUFFER  
;STARTING AT 'OUTBUF'  
TST.SIN  
MOV #-400,R0  
MOV #OUTBUF,R1  
8$: CLR (R1)+  
INC R0  
BNE 8$  
CLR R0  
MOV #-400,@RKWC ;READ 256 WORDS  
MOV #OUTBUF,@RKBA ;INTO THIS ADRES  
MOV DRIVAD,@RKDA ;STARTING FROM THIS DISK ADRES  
MOV #5,@RKCS ;READ, GO  
1$: TSTB @RKCS ;DID CNTRL RDY SET?  
BMI 2$ ;YES, BRANCH  
INC R0 ;WAITED LONG ENOUGH?  
BNE 1$ ;IF NOT, LUP BAK & WAIT  
;ERROR, IF YES  
JSR PC,GT4RG ;GO, GET RKCD, ER, DS, DA  
MOV DRIVAD,$REG10 ;GET THE STARTING ADRES  
BRKDA4 ;GO TO 'BDA4' & BREAK CONTENTS OF  
; $REG10 INTO DRV #, CYL, SUR, SEC BITS  
ERROR 45 ;CNTRL RDY DID NOT SET AFTER READ  
; OF 400 WORDS FROM CYL 0, SEC 0  
; 'RKDA' IN EROR MSGE GIVES THE  
; CONTENTS OF RKDA AT THE TIME OF EROR  
; READ WAS DONE STARTING AT <DSK-ADRES>  
; INDICATED IN EROR MESGE  
2$: * BIT #1000,@RKDS ;IS SIN SET?  
BNE TST32 ;:IF YES, EXIT  
5$: MOV #-400,R1  
MOV #177777,R2  
MOV #OUTBUF,R3
```

4278 013116 012705 177773
4279 013122 062702 177401
4280 013126 020213
4281
4282 013130 001414
4283
4284 013132 010137 001162
4285 013136 062737 000401 001162
4286 013144 010237 001164
4287
4288 013150 011337 001166
4289 013154 104055
4290
4291
4292
4293 013156 005205
4294 013160 001403
4295 013162 005723
4296
4297 013164 005201
4298 013166 001355
4299
4300
4301
4302
4303
4304
4305
4306
4307 013170 000004
4308 013172 104413
4309
4310
4311
4312
4313
4314
4315
4316
4317
4318 013174 104421
4319
4320 013176 013701 001332
4321 013202 013702 001334
4322 013206 013703 001340
4323 013212 013704 001336
4324 013216 012737 052525 033342
4325 013224 012712 177400
4326 013230 013713 001350
4327 013234 012714 033342
4328 013240 012711 000013
4329
4330 013244 105711
4331 013246 100003
4332 013250 004737 021002
4333 013254 104030

6\$: MOV #-5,R5
ADD #177401,R2
CMP R2,(R3); WAS THE READ WORD SAME AS THE WORD
; THAT WAS SUPPOSE TO BE WRITTEN
; YES, BRANCH
; NO, ERROR
BEQ 7\$
; GET THE # OF WORD
; THAT IS IN ERROR (EXAMPLE=1,2--376,377,400)
; GET EXPCTD WORD (THAT WAS SUPPOSED TO
; BE WRITTEN)
; GET WORD RECVD (THAT WAS READ BAK)
; DID NOT READ BACK WORD THAT WAS SUPPOSED
; TO HAVE BEEN WRITTEN PREVIOUSLY. POSITION
; OF WORD IN ERROR IS AS INDICATED BY
; WORD # (\$REG0), SEC 0, CYL 0
MOV R1,\$REG0
ADD #401,\$REG0
MOV R2,\$REG1
MOV (R3),\$REG2
ERROR 55
7\$: INC R5
BEQ TST32
TST (R3)+
;:EXIT
; INCREMENT POINTER TO NXT WORD (THAT
; WAS READ BACK)
; HAVE U CHKD ALL 256 WORDS?
; IF NOT, LUP BAK & CHK THE NXT WORD
; IF YES, EXIT
;:*****
; *TEST 32 CHECK 'READ CHECK' FUNCTION - CYLINDER 0, SECTOR 0
; *THIS TEST CHECKS OUT THE BASIC 'READ CHECK' LOGIC, USING THE DATA BLOCK
; *'CYLINDER, SECTOR 0) WRITTEN IN A PREVIOUS TEST. HENCE THE TEST WHICH
; *WRITES THE DATA BLOCK SHOULD BE DONE PRIOR TO THIS TEST.
;:*****
TST32: SCOPE
CNT.RESET
; GO, DO CONTROL RESET
; THIS IS A CALL FOR THE 'CNTRL-
; RESET' ROUTINE. A CONTROL RESET IS
; ISSUED AND AFTER A CERTAIN TIME
; IF THE 'CNTRL RDY' DOES NOT SET
; AN ERROR IS REPORTED. NOTE THAT
; THE PC IN ERROR MESSAGE IS THE
; PC WHERE 'CNT.RESET' IS LOCATED.
; THIS IS A VERY BASIC ERR& IF IT
; OCCURS GO BACK TO TEST 10
; CHECK IF SIN IS SET, IF SET
; DO DRIVE RESET TO CLEAR IT
TST.SIN
MOV RKCS,R1
MOV RKWC,R2
MOV RKDA,R3
MOV RKBA,R4
MOV #52525,OUTBUF
MOV #-400,@R2
MOV DRIVAD,@R3
MOV #OUTBUF,@R4
MOV #13,@R1
; READ CHECK 256 WORDS
; STARTING FROM CYL 0, SECTOR 0
; READ CHECK, GO
1\$: TSTB @R1
BPL 2\$
JSR PC,GT3RG
ERROR 30
; DID CNTRL RDY GET CLEARED AS GO WAS SET?
; YES, BRANCH
; GET RKCS, ER, DS
; CNTRL RDY DID NOT CLEAR AS GO


```
4334 013256 104412 2$: CHKCRDY ;GO CHECK IF CONTROL RDY IS SET
4335 ;IF SO, SKIP THE EROR MESSAGE.
4336 ;WAS SET TO 'READ CHECK'
4337 013260 104056 ERROR 56 ;CNTRL RDY DID NOT SET ON DOING
4338 ;'READ CHECK' FROM CYL 0, SEC 0
4339 013262 032711 140000 3$: BIT #140000,@R1 ;DID 'ERR' OR 'HE' BIT SET?
4340 013266 001403 BEQ 4$ ;NO, BRANCH
4341 013270 004737 021002 JSR PC,GT3RG ;GO, GET RKCS,ER,DS FOR ERROR MESSAGE
4342 013274 104057 ERROR 57 ;'ERR' OR 'HE' BIT SET ON DOING
4343 ;'READ CHECK' ON CYLINDER 0, SEC 0
4344 013276 032777 000002 166024 4$: BIT #2,@RKER ;DID 'CSE' BIT SET IN RKER?
4345 013304 001404 BEQ 5$ ;NO, BRANCH
4346 013306 017737 166016 001162 MOV @RKER,$REG0 ;GET RKER
4347 013314 104060 ERROR 60 ;SOFT ERROR - CSE - ON DOING 'READ
4348 ;CHECK' ON CYLINDER 0, SECTOR 0
4349 ;U SHOULD HAVE GOT ERROR 102 ALSO
4350 013316 005712 5$: TST @R2 ;DID WORD COUNT OVERFLOW TO 0?
4351 013320 001405 BEQ 6$ ;YES, BRANCH
4352 013322 011237 001162 MOV @R2,$REG0 ;GET RKWC
4353 013326 011137 001164 MOV @R1,$REG1 ;GET RKCS
4354 013332 104061 ERROR 61 ;WORD COUNT DID NOT OVERFLOW
4355 ;ON DOING 'READ CHK' ON CYL 0, SEC 0
4356 013334 013702 001350 6$: MOV DRIVAD,R2 ;RKDA SHOULD INCREMENT
4357 013340 005202 INC R2 ;TO THIS AFTER 'RD CHK' IS DONE
4358 013342 020213 CMP R2,@R3 ;DID RKDA INCREMENT CORRECTLY?
4359 013344 001405 BEQ 7$
4360 013346 010237 001162 MOV R2,$REG0 ;GET EXPCTD RKDA
4361 013352 011337 001164 MOV @R3,$REG1 ;GET RKDA RECVD
4362 013356 104062 ERROR 62 ;RKDA DID NOT INCREMENT CORRECTLY
4363 ;(BY 1) ON DOING 'READ CHK' ON
4364 ;CYL 0, SEC 0
4365 013360 022714 033342 7$: CMP #OUTBUF,@R4 ;DID RKBA GET CHANGED?
4366 013364 001406 BEQ 8$ ;NO, BRANCH (RKBA WON'T CHANGE, NO NPR'S)
4367 013366 012737 033342 001162 MOV #OUTBUF,$REG0 ;GET EXPCTD RKBA
4368 013374 011437 001164 MOV @R4,$REG1 ;GET RKBA RECVD
4369 013400 104063 ERROR 63 ;RKBA CHANGED AFTER DOING 'READ CHK'
4370 ;ON CYLINDER 0, SECTOR 0. SHOULD
4371 ;NOT CHANGE, FOR, NO NPR'S.
4372 013402 022737 052525 033342 8$: CMP #52525,OUTBUF ;'OUTBUF' SHOULD STILL CONTAIN THE
4373 ;SAME WORD AS IT DID BEFORE 'RD CHK'
4374 ;NOTE THAT AT THE BEGINING OF THIS TEST
4375 ;52525 WAS WRITTEN INTO 'OUTBUF'
4376 013410 001412 BEQ TST33 ;:YES, EXIT
4377 ;REPORT ERROR IF 'OUTBUF' CHANGED
4378 013412 012737 033342 001162 MOV #OUTBUF,$REG0 ;GET ADRES OF OUTBUF
4379 013420 012737 052525 001164 MOV #52525,$REG1 ;GET EXPCTD WORD IN 'OUTBUF'
4380 013426 013737 033342 001166 MOV OUTBUF,$REG2 ;GET WORD FOUND IN 'OUTBUF'
4381 013434 104064 ERROR 64 ;AS MENTIONED ABOVE, IF 'WRITE' OF
4382 ;256 WORD DATA BLOCK WAS DONE
4383 ;CORRECTLY BEFORE, THEN THIS ERROR
4384 ;COULD MEAN THAT AN NPR WAS DONE
4385 ;ON 'READ CHECK'.
```

```
*****
;*TEST 33 CHECK THE 'WRITE CHECK' FUNCTION - ON CYLINDER 0, SECTOR 0
;*THIS TEST CHECKS OUT THE BASIC 'WRITE CHECK' LOGIC, USING THE 256
```

```

4390
4391
4392
4393
4394
4395
4396 013436 000004
4397 013440 104413
4398
4399
4400
4401
4402
4403
4404
4405
4406
4407 013442 104421
4408
4409 013444 013701 001332
4410 013450 012700 177400
4411 013454 012702 033342
4412 013460 012703 177777
4413 013464 062703 177401
4414 013470 010322
4415 013472 005200
4416 013474 001373
4417 013476 012777 177400 165630
4418 013504 012777 033342 105624
4419 013512 013777 001350 165620
4420 013520 012711 000007
4421
4422 013524 005000
4423 013526 105711
4424 013530 100003
4425 013532 004737 021002
4426 013536 104030
4427
4428 013540 104412
4429
4430 013542 104065
4431
4432
4433 013544 032711 140000
4434 013550 001403
4435 013552 004737 021002
4436 013556 104066
4437
4438 013560 032777 000001 165542
4439 013566 001403
4440 013570 004737 021002
4441 013574 104067
4442
4443
4444
4445

```

```

: *WORD DATA BLOCK (SECTOR 0, CYLINDER 0) WRITTEN IN A PREVIOUS
: *TEST. THE BUFFER IN MEMORY, USED FOR COMPARISON OF DATA, IS THE
: *ONE STARTING AT 'OUTBUF'. HENCE THE TEST WHICH WRITES THE
: *256 WORD BLOCK ON THE DISK (AS WELL AS CREATING THE 256
: *256 WORD MEMORY BUFFER) SHOULD BE DONE BEFORE THIS TEST.
: *****
TST33: SCOPE
CNT.RESET
;GO, DO CONTROL RESET
;THIS IS A CALL FOR THE 'CNTRL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.
;THIS IS A VERY BASIC ERR& IF IT
;OCCURS GO BACK TO TEST 10
;CHECK IF SIN IS SET, IF SET
;DO DRIVE RESET TO CLEAR IT

TST.SIN
MOV RKCS,R1
MOV #-400,R0
MOV #OUTBUF,R2
MOV #177777,R3
1$: ADD #177401,R3
MOV R3,(R2)+
INC R0
BNE 1$
MOV #-400,@RKWC ;WRITE CHECK 256 WORDS
MOV #OUTBUF,@RKBA ;STARTING AT THIS BUS ADRES
MOV DRIVAD,@RKDA ;WITH THIS DISK DATA BLOCK (CYL 0, SEC 0)
MOV #7,@R1 ;WRITE CHECK, GO

2$: CLR R0 ;GIVE SOME TIME
TSTB @R1 ;DID CNTRL RDY CLEAR AS GO WAS SET?
BPL 3$ ;YES BRANCH
JSR PC,GT3RG ;GET RKCS, ER, DS
ERROR 30 ;CNTRL RDY DID NOT CLEAR AS GO WAS
;SET TO DO WRITE CHECK

3$: CHKCRDY ;GO CHECK IF CONTROL RDY IS SET
;IF SO, SKIP THE EROR MESSAGE.
ERROR 65 ;CNTRL RDY DID NOT SET AFTER
;COMPLETING WRITE CHECK ON
;CYLINDER 0, SECTOR 0

4$: BIT #140000,@R1 ;DID HE OR ERR BIT SET
BEQ 5$ ;NO, BRANCH
JSR PC,GT3RG ;GO GET RKCS ER DS FOR ERROR MESSAGE
ERROR 66 ;HE OR ERR BIT SET ON DOING WRITE
;CHK ON CYLINDER 0, SEC 0

5$: BIT #1,@RKER ;DID WCE SET IN RKER?
BEQ 6$ ;NO, BRANCH
JSR PC,GT3RG ;YES GET RKCS, ER, DS
ERROR 67 ;WCE ON WRITE CHECK OF CYL 0, SEC 0
;NOTE THAT IF A PREVIOUS TEST
; & THEN COMPARED WITH MEMORY BUFFER
; TO SEE IF IT WAS WRITTEN CORRECT WAS
; DONE RIGHT BEFORE, THIS ERROR SHOULD NOT

```

```

4446                                     ;HAPPEN UNLESS THERE IS A FAULT IN THE
4447                                     ;COMPARING LOGIC OF 'WRT CHK'
4448 013576 005777 165532 6$:   TST    @RKWC    ;DID RKWC OVERFLOW?
4449 013602 001406          BEQ    7$         ;YES, BRANCH
4450 013604 017737 165524 001162  MOV    @RKWC,$REG0 ;NO, GET RKWC
4451 013612 011137 001164      MOV    @R1,$REG1   ;GET RKCS
4452 013616 104061          ERROR  61         ;RKWC DID NOT OVERFLOW AFTER
4453                                     ;WRITE CHECK ON CYL 0, SEC 0
4454 013620 013704 001350 7$:   MOV    DRIVAD, R4 ;RKDA SHOULD INCREMENT
4455 013624 005204          INC     R4         ;TO THIS AFTER WRT CHK
4456 013626 020477 165506      CMP    R4,@RKDA   ;DID RKDA INCREMENT CORRECTLY?
4457 013632 001406          BEQ    8$         ;YES, BRANCH
4458 013634 010437 001162      MOV    R4,$REG0   ;NO, GET EXPCTD RKDA
4459 013640 017737 165474 001164  MOV    @RKDA,$REG1 ;GET RKDA RECVD
4460 013646 104070          ERROR  70         ;RKDA DID NOT INCREMENT CORRECTLY
4461                                     ;(BY 1 SECTOR) AFTER WAT CHK ON SEC 0, CYL 0
4462 013650 022777 034342 165460 8$:   CMP    #OUTBUF+1000,@RKBA ;DID RKBA INCREMENT CORRECTLY?
4463 013656 001407          BEQ    9$         ;YES, EXIT
4464 013660 012737 034342 001162  MOV    #OUTBUF+1000,$REG0 ;GET EPCTD RKBA
4465 013666 017737 165444 001164  MOV    @RKBA,$REG1   ;GET RKBA RECVD
4466 013674 104071          ERROR  71         ;RKBA DID NOT INCREMENT CORRECTLY
4467                                     ;(BY 1000 BYTES) AFTER A WRT CHK
4468                                     ;OF 256 WORDS ON CYL 0, SEC 0
4469 013676 022711 000206 9$:   CMP    #206,@R1    ;DOES RKCS STILL CONTAIN THE WRT CHK BITS?
4470 013702 001406          BEQ    TST34       ;YES, BRANCH
4471 013704 012737 000206 001162  MOV    #206,$REG0   ;NO, GET EXPCTD RKCS
4472 013712 011137 001164      MOV    @R1,$REG1   ;GET RKCS RECVD
4473 013716 104024          ERROR  24         ;RKCS BITS CHANGED AFTER WRT CHK
4474                                     ;WAS DONE
4475                                     ;*****
4476                                     ;*TEST 34 CHECK THAT IBA INHIBITS INCREMENTING OF RKBA
4477                                     ;*THIS TEST CHECKS THAT THE BUS ADDRESS DOES NOT INCREMENT WHEN
4478                                     ;*THE IBA BIT IS SET. SEQUENCE OF OPERATIONS:
4479                                     ;*1) CLEAR OUT 256 WORD BUFFER IN MEMORY (OUTBUF)
4480                                     ;*2) READ FROM SECTOR 0, CYLINDER 0 THE 256 WORD BLOCK THAT WAS
4481                                     ;*WRITTEN IN A PREVIOUS TEST (NOTE: THAT TEST SHOULD HAVE BEEN
4482                                     ;*DONE BEFORE THIS). IBA BIT IS SET DURING READ BACK.
4483                                     ;*3) CHECK THAT RKBA DID NOT INCREMENT
4484                                     ;*4) CHECK THAT THE ENTIRE BLOCK WAS READ INTO THE SAME MEMORY
4485                                     ;*WORD (OUTBUF) & THE REST OF THE WORDS IN THAT BUFFER ARE 0
4486                                     ;*AS PREVIOUSLY CLEARED OUT.
4487                                     ;*****
4488 013720 000004          TST34: SCOPE
4489 013722 104413          CNT.RESET ;GO, DO CONTROL RESET
4490                                     ;THIS IS A CALL FOR THE 'CNTRL-
4491                                     ;RESET' ROUTINE. A CONTROL RESET IS
4492                                     ;ISSUED AND AFTER A CERTAIN TIME
4493                                     ;IF THE 'CNTRL RDY' DOES NOT SET
4494                                     ;AN ERROR IS REPORTED. NOTE THAT
4495                                     ;THE PC IN ERROR MESSAGE IS THE
4496                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
4497                                     ;THIS IS A VERY BASIC ERR& IF IT
4498                                     ;OCCURS GO BACK TO TEST 10
4499 013724 104421          TST.SIN ;CHECK IF SIN IS SET, IF SET
4500                                     ;DO DRIVE RESET TO CLEAR IT
4501 013726 013701 001332      MOV    RKCS,R1
    
```

```

4502 013732 012700 177400          MOV    #-400,R0          ;SET UP COUNT FOR 256 WORDS
4503 013736 012702 033342          MOV    #OUTBUF,R2
4504 013742 010203          MOV    R2,R3
4505
4506 013744 005023          1$:   CLR    (R3)+          ;CLEAR OUT THE 256
4507 013746 005200          INC    R0              ;WORD MEMORY BUFFER STARTING
4508 013750 001375          BNE    1$              ;AT 'OUTBUF'
4509 013752 012777 177400 165354  MOV    #-400,@RKWC     ;READ BACK 256 WORDS
4510 013760 010277 165352  MOV    R2,@RKBA        ;INTO THIS BUS ADRES (IBA WILL B SET)
4511 013764 013777 001350 165346  MOV    DRIVAD,@RKDA    ;FROM THIS DSK ADRES (SEC 0, CYL 0)
4512                                     ;NOTE: SEC 0 HAS BEEN WRITTEN IN A
4513                                     ;PREVIOUS TEST WITH A UNIQUE PATTERN
4514 013772 012711 004005          MOV    #4005,@R1      ;READ, GO, IBA SET
4515
4516 013776 005037 001362          2$:   CLR    COUNT
4517 014002 105711          TSTB   @R1             ;DID CNTRL RDY SET?
4518 014004 100412          BMI    3$              ;YES, BRANCH
4519 014006 005237 001362          INC    COUNT          ;WAITED LONG ENOUGH?
4520 014012 001373          BNE    2$              ;IF NOT, LUP BAK & WAIT
4521 014014 004737 020774          JSR    PC,GT4RG        ;GO, GET RKCS, ER, DS, DA
4522 014020 013737 001350 001202  MOV    DRIVAD,$REG10   ;GET THE STARTING ADRES
4523 014026 104416          BRKDA4                 ;BREAK CONTENTS OF $REG10
4524                                     ;INTO DR #, CYL, SUR, SEC
4525 014030 104045          ERROR  45              ;CNTRL RDY DID NOT SET AFTER DOING
4526                                     ;READ
4527 014032 004737 021234          3$:   JSR    PC,CHKHE     ;CHECK IF 'ERR' OR 'HE' BIT IS SET,
4528                                     ;IF YES, RETURN HERE.
4529 014036 104046          ERROR  46              ;ERR BIT SET ON DOING READ FROM SEC 0,
4530                                     ;CYL 0 (INDICATED IN <DSK-ADRES>)
4531                                     ;'RKDA' IN EROR MSGE GIVES THE
4532                                     ;CONTENTS OF RKDA AT THE TIME OF EROR
4533
4534 014040 020277 165272          4$:   CMP    R2,@RKBA     ;DID RKBA INCREMENT?
4535 014044 001406          BEQ    5$              ;OK IF NOT, BRANCH
4536 014046 010237 001162          MOV    R2,$REG0        ;GET EXPCTD RKBA
4537 014052 017737 165260 001164  MOV    @RKBA,$REG1     ;GET RKBA RECVD
4538 014060 104072          ERROR  72              ;RKBA INCREMNTED WHEN IBA BIT WAS
4539                                     ;SET, SHOULD NOT HAVE
4540 014062 032777 001000 165236  5$:   BIT    #1000,@RKDS    ;IS SIN SET?
4541 014070 001042          BNE    TST35           ;:IF YES, EXIT
4542 014072 012700 177400          MOV    #-400,R0
4543 014076 022712 000377          CMP    #377,@R2
4544                                     ;CHECK THAT THE FIRST WORD IN
4545                                     ;'OUTBUF' IS 377 (LAST WORD OF SEC 0,
4546                                     ;CYL 0). NOTE THAT READ WAS DONE
4547 014102 001411          BEQ    6$              ;INTO THIS SAME WRD WITH IBA SET
4548 014104 012737 000377 001162  MOV    #377,$REG0      ;GET EXPCTD WORD (LAST WORD OF THE BUFFER
4549 014112 011237 001164          MOV    (R2),$REG1     ;GET WORD RECVD (LAST WRD FROM SEC 0)
4550 014116 013737 001350 001166  MOV    DRIVAD,$REG2   ;DISK ADRES WHERE ERROR OCCURED
4551                                     ;(SEC 0, CYL 0 LAST WORD)
4552                                     ;DATA ERROR
4553 014124 104044          ERROR  44              ;THE FIRST WORD IN MEM BUFFER (OUTBUF)
4554                                     ;SHOULD BE NON-ZERO & SHOULD CONTAIN
4555                                     ;THE LAST WORD READ BACK FROM SEC 0
4556 014126 005722          6$:   TST    (R2)+
4557 014130 012705 177773          MOV    #-5,R5         ;INCREMENT POINTER TO THE NXT WORD
                                     ;ALLOW ONLY 5 MESAGES FOR ERR 116
    
```

4558	014134	005200		7\$:	INC	R0		:CHKD ALL 256 WORDS IN THE BUFFER?
4559	014136	001417			BEQ	TST35		:::YES, EXIT
4560	014140	005722			TST	(R2)+		:IS THIS WORD 0?
4561	014142	001774			BEQ	7\$:YES, LUP BAK & CHK THE NXT WORD?
4562	014144	005037	001164		CLR	\$REG1		:ERROR. GET EXPCTED WORD - 0
4563	014150	014237	001166		MOV	-(R2), \$REG2		:GET WORD THAT WAS FOUND IN THE BUFFER
4564	014154	010004			MOV	R0, R4		
4565	014156	062704	000401		ADD	#401, R4		
4566	014162	010437	001162		MOV	R4, \$REG0		:THIS 'WORD #' IN MEMORY BUFFER
4567								:SHOULD HAVE BEEN ZERO
4568	014166	104073			ERROR	73		:THE 256 WORD BUFER (STARTING AT
4569								:OUTBUF) WAS CLEARED BEFORE READING
4570								:BAK SEC 0 INTO IT. SINCE THE IBA
4571								:BIT WAS SET DURING THE READ, ONLY
4572								:THE FIRST WORD OF (OUTBUF) SHOULD
4573								:HAVE CHANGED, THE REST OF THE WORDS
4574								:SHOULD BE STILL 0. IF THIS ERROR
4575								:OCCURS, 'WORD #' (OF THE BUFFER) AS
4576								:INDICATED IN THE EROR MESSAGE) GOT
4577								:CHANGED WHEN READ WAS DONE FROM
4578								:THE DISK, INDICATING THAT WITH IBA
4579								:SET X-FER WAS NOT DONE INTO THE
4580								:SAME MEMORY LOCATION. 'WORD #'
4581								:IS OCTAL & SPECIFIES THE POSITION
4582								:IN THE BUFFER (FIRST WORD IS 'WORD #' 1)
4583	014170	005205			INC	R5		
4584	014172	001401			BEQ	TST35		:::EXIT
4585	014174	000757			BR	7\$		
4586								
4587								
4588								
4589								
4590								
4591								
4592								
4593								
4594								
4595	014176	000004						
4596	014200	104413						
4597								
4598								
4599								
4600								
4601								
4602								
4603								
4604								
4605								
4606	014202	104421						
4607								
4608	014204	012746	000340					
4609	014210	012746	014216					
4610	014214	000002						
4611	014216							
4612	014216	013701	001332					
4613	014222	013700	001402					

```
7$: INC R0 ;CHKD ALL 256 WORDS IN THE BUFFER?
    BEQ TST35 ;:YES, EXIT
    TST (R2)+ ;IS THIS WORD 0?
    BEQ 7$ ;YES, LUP BAK & CHK THE NXT WORD?
    CLR $REG1 ;ERROR. GET EXPCTED WORD - 0
    MOV -(R2), $REG2 ;GET WORD THAT WAS FOUND IN THE BUFFER
    MOV R0, R4
    ADD #401, R4
    MOV R4, $REG0 ;THIS 'WORD #' IN MEMORY BUFFER
                    ;SHOULD HAVE BEEN ZERO
                    ;THE 256 WORD BUFER (STARTING AT
                    ;OUTBUF) WAS CLEARED BEFORE READING
                    ;BAK SEC 0 INTO IT. SINCE THE IBA
                    ;BIT WAS SET DURING THE READ, ONLY
                    ;THE FIRST WORD OF (OUTBUF) SHOULD
                    ;HAVE CHANGED, THE REST OF THE WORDS
                    ;SHOULD BE STILL 0. IF THIS ERROR
                    ;OCCURS, 'WORD #' (OF THE BUFFER) AS
                    ;INDICATED IN THE EROR MESSAGE) GOT
                    ;CHANGED WHEN READ WAS DONE FROM
                    ;THE DISK, INDICATING THAT WITH IBA
                    ;SET X-FER WAS NOT DONE INTO THE
                    ;SAME MEMORY LOCATION. 'WORD #'
                    ;IS OCTAL & SPECIFIES THE POSITION
                    ;IN THE BUFFER (FIRST WORD IS 'WORD #' 1)
    ERROR 73
    INC R5
    BEQ TST35 ;:EXIT
    BR 7$
:
:*****
:*TEST 35 CHECK THAT RK11 INTERRUPTS WHEN IDE IS SET
:*THIS TEST CHECKS IF RK11 INTERRUPTS TO ITS DESIGNATED VECTOR
:*ADDRESS WHEN IDE BIT IS SET, WITH CONTROL READY SET & GO CLEAR.
:* IT IS NORMALLY 220, UNLESS IT HAS BEEN CHANGED. IF IT HAS BEEN
:*CHANGED RK11 WILL INTERRUPT TO 'RKVEC'. NOTE 'RKVEC' HAS
:*TO BE SET UP BY THE USER.
:*****
TST35: SCOPE
        CNT.RESET ;GO, DO CONTROL RESET
                    ;THIS IS A CALL FOR THE 'CNTRL-
                    ;RESET' ROUTINE. A CONTROL RESET IS
                    ;ISSUED AND AFTER A CERTAIN TIME
                    ;IF THE 'CNTRL RDY' DOES NOT SET
                    ;AN ERROR IS REPORTED. NOTE THAT
                    ;THE PC IN ERROR MESSAGE IS THE
                    ;PC WHERE 'CNT.RESET' IS LOCATED.
                    ;THIS IS A VERY BASIC ERR& IF IT
                    ;OCCURS GO BACK TO TEST 10
                    ;CHECK IF SIN IS SET. IF SET
                    ;DO DRIVE RESET TO CLEAR IT
        TST.SIN
        MOV #340, -(SP)
        MOV #64$, -(SP)
        RTI
64$: MOV RKCS, R1
     MOV RKVEC, R0 ;GET POINTER TO RK VECTOR ADRES
```



```

4670                                     ;THIS IS A VERY BASIC ERR& IF IT
4671                                     ;OCCURS GO BACK TO TEST 10
4672 014342 013700 001332                MOV    RKCS,R0
4673 014346 013777 001350 164764        MOV    DRIVAD,@RKDA
4674 014354 004737 021504                JSR    PC,DRESET
4675 014360 104026                        ERROR  26
4676                                     ;ADRES THE DRIVE
4677 014362 013701 001402                2$:   MOV    RKVEC,R1
4678 014366 012721 014432                MOV    #3$, (R1)+
4679 014372 012711 000340                MOV    #340, (R1)
4680 014376 052777 000040 164734        BIS    #40,@RKDA
4681 014404 012710 000111                MOV    #111,@RO
4682 014410 104420 000300                WAT.INT ,300
4683                                     ;SET UP VECTOR ADRES FOR RK11 INTERUPT
4684                                     ;SET UP PSW ON INTERRUPT
4685                                     ;ADRES CYLINDER #1
4686                                     ;SEEK, GO WITH IDE SET
4687                                     ;WAIT FOR THE DRIVE TO
4688 014414 012777 004600 164760        MOV    #BADINT,@RKVEC
4689 014422 011037 001162                MOV    @RO,$REGO
4690 014426 104075                        ERROR  75
4691                                     ;RESTORE UNEXPECTED RK11 INTERRUPT
4692 014430 000402                        BR     3$+4
4693 014432 022626                        3$:   CMP    (SP)+,(SP)+
4694                                     ;OK, IF RK11 INTERRUPTED TO THIS
4695 014434 022626                        CMP    (SP)+,(SP)+
4696                                     ;RESTORE STACK POINTER (FROM RK11 INTERRUPT)
4697 014436 012777 014502 164736        MOV    #5$,@RKVEC
4698 014444 032710 020000                BIT    #20000,@RO
4699 014450 001403                        BEQ    4$
4700 014452 011037 001162                MOV    @RO,$REGO
4701 014456 104076                        ERROR  76
4702                                     ;GET RKCS
4703 014460 104420 056700                4$:   WAT.INT ,56700
4704                                     ;SCP SET BEFORE SEEK TO LAST
4705                                     ;CYLINDER WAS DONE
4706                                     ;WAIT FOR DRIVE TO INTERRUPT
4707 014464 012777 004600 164710        MOV    #BADINT,@RKVEC
4708                                     ;AFTER SEEK WAS COMPLETED
4709 014472 004737 021002                JSR    PC,GT3RG
4710 014476 104077                        ERROR  77
4711                                     ;WAITING TIME=180 MS FOR 11/20
4712 014500 000401                        BR     5$+2
4713 014502 022626                        5$:   CMP    (SP)+,(SP)+
4714                                     ;OK, IF RK11 INTERUPTED TO THIS AFTER
4715                                     ;SEEK WAS COMPLETED. RESTORE
4716 014504 022626                        CMP    (SP)+,(SP)+
4717                                     ;STACK POINTER (FROM RK11 INTERRUPT)
4718 014506 012777 004600 164666        MOV    #BADINT,@RKVEC
4719                                     ;RESTORE RK11 INTERRUPT VECTOR ADRES
4720 014514 032710 020000                BIT    #20000,@RO
4721 014520 001003                        BNE    6$
4722 014522 011037 001162                MOV    @RO,$REGO
4723 014526 104053                        ERROR  53
4724                                     ;FOR UNEXPECTED INTERUTS
4725 014530 017701 164572                6$:   MOV    @RKDS,R1
                                     ;DID SCP BIT SET?
                                     ;YES, BRANCH
                                     ;GET RKCS
                                     ;SCP DID NOT SET AFTER RK11 INTERRUPTED
                                     ;INDICATING SEEK WAS DONE
                                     ;GET RKDS

```



```

4782 014662 013700 001332      MOV      RKCS,R0
4783 014666 013702 001340      MOV      RKDA,R2
4784 014672 013704 001336      MOV      RKBA,R4
4785 014676 013701 001350      MOV      DRIVAD,R1
4786 014702 052701 000013      BIS      #13,R1          ;SET BITS FOR SEC 13
4787 014706 012777 177600 164420  MOV      #-200,@RKWC      ;READ 200 (OCTAL WORDS)
4788 014714 010112          MOV      R1,@R2          ;FROM THIS DISK ADRES (CYL 0, SEC 13)
4789 014716 012714 033342      MOV      #OUTBUF,@R4     ;INTO THIS BUS ADRES
4790 014722 013705 001402      MOV      RKVEC,R5
4791 014726 012725 014764      MOV      #1$, (R5)+      ;SET UP VECTOR ADRES FOR RK11 TO INTRUPT
4792 014732 012715 000340      MOV      #340, (R5)      ;SET PSW ON INTERUPT
4793 014736 012710 000105      MOV      #105,@R0        ;READ, GO, IDE SET
4794 014742 104420 127710      WAT.INT ,127710         ;WAIT FOR RK11 TO INTERRUPT ON
4795          ;COMPLETION OF READ
4796          ;WAITING TIME= 337 MS FOR 11/20
4797          ;67 MS FOR 11/45
4798 014746 012777 004600 164426  MOV      #BADINT,@RKVEC  ;RESTORE UNEXPTED INTERRUPT VECTOR ADRES
4799 014754 011037 001162      MOV      @R0,$REG0       ;GET RKCS
4800 014760 104101          ERROR    101             ;RK11 DID NOT INTERRUPT AFTER READ
4801          ;WAS DONE, IDE BIT SET.
4802 014762 000404          BR      1$+10
4803 014764 022626 1$:      CMP      (SP)+,(SP)+     ;OK, IF RK11 INTERRUPTED TO THIS
4804          ;RESTORE STACK POINTER (FROM RK11 INTERRUPT)
4805 014766 022626          CMP      (SP)+,(SP)+     ;RESTORE STACK POINTER (FROM WAT.INT)
4806 014770 012777 004600 164404  MOV      #BADINT,@RKVEC  ;RESTORE UNEXPECTED RK11 INTERRUPT
4807          ;VECTOR ADRES
4808 014776 004737 021342      JSR      PC,CHKER        ;CHECK IF ANY BIT IN RKER IS SET,
4809          ;IF YES, RETURN HERE.
4810 015002 104036          ERROR    36             ;RKER SET ON DOING READ FROM SEC 0,
4811          ;CYL 13 IN INTERRUPT MODE
4812 015004 062701 000005 4$:      ADD      #5,R1           ;RKDA SHOULD HAVE INCREMENTED TO THIS
4813 015010 020112          CMP      R1,@R2         ;DID RKDA INCREMENT CORRECTLY?
4814 015012 001405          BEQ      2$             ;YES BRANCH
4815 015014 010137 001162      MOV      R1,$REG0       ;GET EXPCTD RTDA
4816 015020 011237 001164      MOV      @R2,$REG1      ;GET RKDA RECVD
4817 015024 104040          ERROR    40             ;RKDA INCREMENTED WRONG ON DOING
4818          ;A READ ON CYL 0, SEC 13
4819 015026 004737 021316 2$:      JSR      PC,CHKWC        ;CHECK THAT RKWC OVERFLOWED TO 0,
4820          ;IF NOT RETURN HERE.
4821 015032 104041          ERROR    41             ;RKWC DIDN'T OUFLO AFTER
4822          ;A READ OF 200 WORDS
4823
4824 015034 3$:
4825 015034 012746 000340      MOV      #340,-(SP)
4826 015040 012746 015046      MOV      #64$,-(SP)
4827 015044 000002          RTI
4828 015046 64$:
4829 015046 022714 033742      CMP      #OUTBUF+400,@R4 ;DID RKBA INCREMENT CORRECTLY?
4830 015052 001406          BEQ      TST40          ;:YES, EXIT
4831 015054 012737 033742 001162  MOV      #OUTBUF+400,$REG0 ;GET EXPCT RKBA
4832 015062 011437 001164      MOV      @R4,$REG1      ;GET RKBA RECVD
4833 015066 104042          ERROR    42             ;RKBA DID NOT INCREMENT CORRECTLY
4834          ;AFTER A READ OF 200 WORDS
4835
4836          ;*****
4837          ;*TEST 40      CHECK THAT RK11 INTERRUPTS AT BR5 ONLY
    
```

L 7

4838
4839
4840
4841
4842
4843
4844
4845
4846 015070 000004
4847 015072 104413
4848
4849
4850
4851
4852
4853
4854
4855
4856
4857 015074 104421
4858
4859 015076 012737 015132 001110
4860
4861 015104 013700 001332
4862 015110 013777 001350 164222
4863 015116 012701 000007
4864 015122 012702 000340
4865 015126 013703 001400
4866
4867
4868
4869 015132 013704 001402
4870 015136 012724 015244
4871 015142 012714 000340
4872 015146 010246
4873 015150 012746 015156
4874 015154 000002
4875 015156
4876 015156 012710 000100
4877 015162 012705 177760
4878 015166 005205
4879 015170 001376
4880 015172 020203
4881 015174 003005
4882
4883
4884 015176 010137 001162
4885 015202 011037 001164
4886 015206 104103
4887
4888
4889 015210 005010
4890 015212 062702 177740
4891
4892 015216 005301
4893 015220 001344

:*THIS TEST CHECKS THAT RK11 CAN INTERRUPT AT BR5 ONLY. IF IT
:*INTERRUPTS AT A LEVEL HIGHER THAN BR5 AN ERROR IS INDICATED.
:*IF IT DOES NOT INTERRUPT AT BR5 OR LOWER THEN ALSO AN
:*ERROR IS INDICATED. IF FOR SOME REASON THE INTERRUPT
:*LEVEL IS CHANGED FROM BR5, THEN CONTENTS OF RKPRI WILL
:*HAVE TO BE CHANGED ACCORDINGLY AND STILL TEXT WILL
:*CHECK FOR THIS BR LEVEL.

::*****

TST40: SCOPE
CNT.RESET

;GO, DO CONTROL RESET
;THIS IS A CALL FOR THE 'CNTRL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.
;THIS IS A VERY BASIC ERR& IF IT
;OCCURS GO BACK TO TEST 10
;CHECK IF SIN IS SET, IF SET
;DO DRIVE RESET TO CLEAR IT
;SET RETURN ADRES FOR LUPING
;ON ERROR (SW 9)

TST.SIN

MOV #1\$, \$LPERR
MOV RKCS, R0
MOV DRIVAD, @RKDA
MOV #7, R1
MOV #340, R2
MOV RKPRI, R3

;PRIORITY LEVEL 7
;BR LEVEL 7 FOR PSW
;NOTE, IF RK11 INTERRUPT LEVEL IS
;CHANGED FROM 5 TO ANY OTHER LEVEL
;THEN CHANGE CONTENTS OF 'RKPRI'
; ACCORDINGLY

1\$: MOV RKVEC, R4
MOV #3\$, (R4)+
MOV #340, (R4)
MOV R2, -(SP)
MOV #4\$, -(SP)
RTI

;SET UP ADRES FOR RK11 TO INTERUPT
;SET UP PSW ON INTERUPT
;SET PROCESSOR PRIORITY LEVEL AS

4\$: MOV #100, @R0
MOV #-20, R5
INC R5
BNE .-2
CMP R2, R3
BGT 2\$

;INDICATED BY R2
;SET THE IDE BIT
;WAIT FOR THE RK11 INTERRUPT
;WAITING TIME=78 US FOR 11/20
;13 US FOR 11/45
;WAS THE CPU PRIORITY LEVEL LESS THAN
;THE RK11 LEVEL? IF YES, RK11
;SHOULD HAVE INTERRUPTED. ERROR,
;IF IT DID NOT

MOV R1, \$REG0
MOV @R0, \$REG1
ERROR 103

;GET CPU BR LEVEL
;GET RKCS
;THOUGH CPU LEVEL WAS LESS THAN
;THE RK11 LEVEL (5), RK11 DID NOT
;INTERRUPT

2\$: CLR @R0
ADD #-40, R2

;CLEAR RKCS
;DECREASE THE PRIORITY LEVEL (FOR
;CPU) BY 1
;CPU WILL B AT THIS LEVEL
;LUP BAK & CHK FOR THIS BR LEVEL.

```

4894
4895 015222 012777 004600 164152      MOV      #BADINT,@RKVEC      ;DONE WITH CHKING FOR ALL LEVELS.
4896                                     ;RESTORE UNEXPECTED RK11 INTERRUPT
4897 015230 012746 000340              MOV      #340,-(SP)          ;VECTOR
4898 015234 012746 015242              MOV      #648,-(SP)
4899 015240 000002                      RTI
4900 015242                                64$:
4901 015242 000414                      BR       TST41              ;;EXIT,TO NXT TST
4902
4903 015244 022626                                3$:
4904 015246 012777 004600 164126      CMP      (SP)+,(SP)+        ;RESTORE STACK POINTER
4905                                     MOV      #BADINT,@RKVEC      ;RESTORE UNEXPECTED RK11 INTERRUPT
4906 015254 020203                                *      CMP      R2,R3          ;VECTOR
4907 015256 003754                                *      BLE     2$             ;IF THIS INTERRUPT OCCURED WHEN
4908                                     ;CPU LEVEL WAS LESS THAN THE
4909                                     ;RK11 PRIORITY LEVEL (5) THEN IT IS
4910 015260 010137 001162              MOV      R1,$REG0           ;OK. IF NOT SO, ERROR
4911 015264 011037 001164              MOV      @R0,$REG1          ;GET CPU BR LEVEL
4912 015270 104104              ERROR  104                  ;GET RKCS
4913                                     ;RK11 INTERRUPTED WHEN THE CPU
4914                                     ;LEVEL (AS POINTED BY R1) WAS
4915                                     ;HIGHER OR SAME AS THE RK11
4916 015272 000746              BR       2$                  ;LEVEL (5)
4917                                     ;GO BACK & CHK THE NXT LEVEL
4918
4919                                     ;*****
4920                                     ;*TEST 41      SIMULATE & CHECK 'OVR' ERROR
4921                                     ;*THIS TEST SIMULATES OVERRUN ERROR AND CHECKS IF THE OVR
4922                                     ;*BIT IN RKER GETS SET. THEN IT IS CLEARED USING CNTRL RESET
4923                                     ;*& CHECKED THAT IT WAS CLEARED. OVR CONDITION IS SIMULATED
4924                                     ;*BY TRYING TO READ 401(OCTAL) WORDS FROM LAST CYLINDER(312),
4925                                     ;*LAST SECTOR (13), SURFACE 1.
4926 015274 000004                                TST41: SCOPE
4927 015276 104413                                CNT.RESET
4928                                     ;GO, DO CONTROL RESET
4929                                     ;THIS IS A CALL FOR THE 'CNTRL-
4930                                     ;RESET' ROUTINE. A CONTROL RESET IS
4931                                     ;ISSUED AND AFTER A CERTAIN TIME
4932                                     ;IF THE 'CNTRL RDY' DOES NOT SET
4933                                     ;AN ERROR IS REPORTED. NOTE THAT
4934                                     ;THE PC IN ERROR MESSAGE IS THE
4935                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
4936                                     ;THIS IS A VERY BASIC ERR& IF IT
4937 015300 104421              TST.SIN          ;OCCURS GO BACK TO TEST 10
4938                                     ;CHECK IF SIN IS SET, IF
4939 015302 013701 001350              MOV      DRIVAD,R1          ;SET, DO DRIVE RESET TO CLR IT
4940 015306 052701 014533              BIS      #14533,R1         ;GET ADRES OF DRIVE
4941                                     ;SET BITS FOR LAST CYLINDER (312),
4942 015312 012777 177377 164014      MOV      #-401,@RKWC       ;SUR 1, LAST SECTOR (13)
4943 015320 012777 033342 164010      MOV      #OUTBUF,@RKBA     ;READ 401 WORDS
4944 015326 010177 164006              MOV      R1,@RKDA          ;INTO THIS MEMORY BUFFER
4945                                     ;FROM THIS DSK ADRES, LAST CYL.
4946 015332 012777 000005 163772      MOV      #5,@RKCS          ;LAST SEC, SURFACE 1
4947                                     ;READ, GO
4948 015340 005002                                1$:
4949 015342 105777 163764              CLR      R2
4949                                     TSTB    @RKCS              ;DID CNTRL RDY SET?

```

```

4950 015346 100410 BMI 2$ :YES, BRANCH
4951 015350 005202 INC R2 :NO, WAIT FOR IT
4952 015352 001373 BNE 1$ :IF WAITED LONG, REPORT ERROR MESSAGE BECAUSE
4953 :OVR SHOULD HAVE SET HE CAUSING
4954 :CNTRL RDY TO SET BY NOW
4955 015354 017737 163754 001166 MOV @RKWC,$REG2
4956 015362 004737 021010 JSR PC,GT2RG :GO, GET RKCS, ER
4957 015366 104002 ERROR 2 :CNTRL RDY DID NOT SET AFTER DOING
4958 :AN OVR READ. HE SHOULD HAVE OCCURED
4959 :SETTING CNTRL RDY (HE BECAUSE OF
4960 :OVR CONDITIONS)
4961 015370 032777 040000 163732 2$: BIT #40000,@RKER :DID OVR BIT SET IN RKER?
4962 015376 001006 BNE 3$
4963 015400 004737 021010 JSR PC,GT2RG :GET RKCS, ER
4964 015404 012737 040000 001166 MOV #40000,$REG2 :THIS BIT (OVR) DID NOT SET.
4965 015412 104105 ERROR 105 :OVR ERROR BIT DID NOT SET IN RKER
4966 :ON SIMULATING OVR CONDITIONS
4967 015414, 022777 140204 163710 3$: CMP #140204,@RKCS :DID HE & ERR SET WHEN OVR SET IN RKER?
4968 015422 001403 BEQ 4$ :YES, BRANCH
4969 015424 004737 021010 JSR PC,GT2RG :GET RKCS, ER
4970 015430 104106 ERROR 106 :HE OR ERR BIT DID NOT SET IN RKCS WHEN
4971 :AN OVR ERROR WAS SIMULATED
4972 :CLEAR OVER, ERR, HE BITS
4973 015432 104413 4$: CNT.RESET :GO, DO CONTROL RESET
4974 :THIS IS A CALL FOR THE 'CNTRL-
4975 :RESET' ROUTINE. A CONTROL RESET IS
4976 :ISSUED AND AFTER A CERTAIN TIME
4977 :IF THE 'CNTRL RDY' DOES NOT SET
4978 :AN ERROR IS REPORTED. NOTE THAT
4979 :THE PC IN ERROR MESSAGE IS THE
4980 :PC WHERE 'CNT.RESET' IS LOCATED.
4981 :THIS IS A VERY BASIC ERR& IF IT
4982 :OCCURS GO BACK TO TEST 10
4983 015434 004737 021356 JSR PC,CHKECLR :CHECK IF 'OVR' BIT WAS CLEARED BY
4984 :CON.RESET, IF NOT RETURN HERE.
4985 015440 104102 ERROR 102 :CNTRL RESET DID NOT CLEAR OVR
4986 :BIT IN RKER
4987 015442 004737 021402 5$: JSR PC,CHKCCLR :CHECK IF 'ERR' & 'HE' BIT GOT CLEARED BY
4988 :CON.RESET, IF NOT RETURN HERE.
4989 015446 104102 ERROR 102 :CNTRL RESET DID NOT CLEAR
4990 :HE OR ERR BIT IN RKCS.
4991 015450 004737 021504 6$: JSR PC,DRESET :GO DO DRIVE RESET
4992 015454 104026 ERROR 26 :R/W/S RDY DIDN'T SET
4993 :AFTER THE ABOVE DRIVE RESET
4994
4995 ;*****
4996 ;*TEST 42 SIMULATE & CHECK PGE ERROR
4997 ;*THIS TEST SIMULATES 'PROGRAMMING ERROR' & CHECKS IF IT IS
4998 ;*DETECTED BY PGE BIT IN RKER. THEN A CNTRL RESET IS DONE &
4999 ;*IT IS CHECKED IF PGE BIT WAS CLEARED. IT IS ALSO CHECKED IF
5000 ;*THE SETTING & CLEARING OF PGE BIT SETS & CLEARS HE, ERR
5001 ;*BITS IN RKCS.
5002 ;*****
5003 015456 000004 TST42: SCOPE
5004 015460 104413 CNT.RESET :GO, DO CONTROL RESET
5005 :THIS IS A CALL FOR THE 'CNTRL-

```

```
5006 ;RESET' ROUTINE. A CONTROL RESET IS
5007 ;ISSUED AND AFTER A CERTAIN TIME
5008 ;IF THE 'CNTRL RDY' DOES NOT SET
5009 ;AN ERROR IS REPORTED. NOTE THAT
5010 ;THE PC IN ERROR MESSAGE IS THE
5011 ;PC WHERE 'CNT.RESET' IS LOCATED.
5012 ;THIS IS A VERY BASIC ERR& IF IT
5013 ;OCCURS GO BACK TO TEST 10
5014 015462 104421 TST.SIN ;GO CHECK IF SIN IS SET, IF
5015 ;SET DO DRIVE RESET TO CLR IT
5016 015464 013701 001330 MOV RKER,R1
5017 015470 013777 001350 163642 MOV DRIVAD,@RKDA ;ADRES THE DRIVE, CYLINDER 0
5018
5019 015476 012777 002011 163626 MOV #2011,@RKCS ;SEEK, GO WITH FMT SET
5020 ;THIS IS A PGE SIMULATION
5021 015504 104414 CNT.RDY ;THIS IS A CALL FOR 'CN.RDY'
5022 ;ROUTINE WHICH WAITS FOR CNT
5023 ;RDY TO SET. IF CNTRL RDY DOES
5024 ;NOT SET WITHIN 883 MS/ 11-20
5025 ;(176 MS FOR 11-45 WITH BIPOLAR)
5026 ;AN ERROR IS REPORTED
5027 015506 032711 004000 BIT #4000,@R1 ;DID PGE BIT IN RKER SET?
5028 015512 001006 BNE 1$ ;YES, BRANCH
5029 015514 012737 004000 001166 MOV #4000,$REG2 ;THIS BIT IN RKER (PGE) DID NOT SET
5030 015522 004737 021010 JSR PC,GT2RG ;GO GET RKCS, ER FOR MESSAGE
5031 015526 104105 ERROR 105 ;PGE BIT DID NOT SET IN RKER
5032 ;ON SIMULATION OF PGE CONDITION
5033 ;$REG2 CONTAINS THE RKER BIT (PGE)
5034 ;THAT SHOULD HAVE SET.
5035 015530 022777 142210 163574 1$: CMP #142210,@RKCS ;DID HE & ERR BITS SET?
5036 015536 001403 BEQ 2$ ;YES, BRANCH
5037 015540 004737 021010 JSR PC,GT2RG ;GO, GET RKCS, ER
5038 015544 104106 ERROR 106 ;HE OR ERR BIT DID NOT SET WHEN
5039 ;PGE SET IN RKER.
5040 ;CLEAR PGE, HE, ERR BITS
5041 015546 104413 2$: CNT.RESET ;GO, DO CONTROL RESET
5042 ;THIS IS A CALL FOR THE 'CNTRL-
5043 ;RESET' ROUTINE. A CONTROL RESET IS
5044 ;ISSUED AND AFTER A CERTAIN TIME
5045 ;IF THE 'CNTRL RDY' DOES NOT SET
5046 ;AN ERROR IS REPORTED. NOTE THAT
5047 ;THE PC IN ERROR MESSAGE IS THE
5048 ;PC WHERE 'CNT.RESET' IS LOCATED.
5049 ;THIS IS A VERY BASIC ERR& IF IT
5050 ;OCCURS GO BACK TO TEST 10
5051 015550 004737 021356 JSR PC,CHKECLR ;CHECK IF 'PGE' BIT GOT CLEARED BY
5052 ;CONTROL RESET, IF NOT RETURN HERE.
5053 015554 104102 ERROR 102 ;CNTRL RESET DID NOT CLEAR
5054 ;PGE BIT IN RKER
5055 015556 004737 021402 3$: JSR PC,CHKCCLR ;CHECK IF 'ERR' BITGOT CLEARED BY
5056 ;CON.RESET, IF NOT RETURN HERE.
5057 015562 104102 ERROR 102 ;RKCS BITS HE OR ERR DID NOT
5058 ;GET CLEARED BY CNTRL RESET
5059
5060 ;*****
5061 ;*TEST 43 SIMULATE & CHECK NXM ERROR
```

5062
5063
5064
5065
5066
5067
5068
5069 015564 000004
5070 015566 104413
5071
5072
5073
5074
5075
5076
5077
5078
5079
5080 015570 104421
5081
5082 015572 005002
5083 015574 013700 001332
5084 015600 012777 177777 163526
5085 015606 012777 160000 163522
5086 015614 013777 001350 163516
5087 015622 012710 000067
5088 015626 105777 163500 1\$:
5089 015632 100410
5090 015634 005202
5091 015636 001373
5092 015640 004737 021010
5093 015644 017737 163464 001166
5094 015652 104113
5095
5096
5097
5098 015654 032777 002000 163446 2\$:
5099 015662 001006
5100 015664 004737 021010
5101 015670 012737 002000 001166
5102 015676 104105
5103
5104 015700 022710 140266 3\$:
5105 015704 001403
5106 015706 004737 021010
5107 015712 104106
5108
5109
5110 015714 104413 4\$:
5111
5112
5113
5114
5115
5116
5117

TST43: SCOPE
CNT.RESET
;GO, DO CONTROL RESET
;THIS IS A CALL FOR THE 'CNTRL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.
;THIS IS A VERY BASIC ERR& IF IT
;OCCURS GO BACK TO TEST 10
;GO CHECK IF SIN IS SET
;IF SET DO DRIVE RESET TO CLR IT
TST.SIN
CLR R2
MOV RKCS,R0
MOV #-1,@RKWC ;WRITE CHECK 1 WORD
MOV #160000,@RKBA ;AT THIS BUS ADRES
MOV DRIVAD,@RKDA ;WITH THIS DISK ADRES (CYL 0, SEC 0)
MOV #67,@R0 ;WRT CHK, GO, MEX BITS SET
1\$: TSTB @RKCS ;DID CNTRL RDY SET AS A RESULT OF HE?
BMI 2\$;YES, BRANCH
INC R2 ;WAITED LONG ENOUGH?
BNE 1\$;IF NOT LUP BAK & WAIT
JSR PC,GT2RG ;GET RKCS, ER
MOV @RKWC,\$REG2 ;GET RKWC
ERROR 113 ;CNTRL RDY DID NOT SET ON DOING
;A WRT CHK WITH A NXM LOCATION.
;THIS HE SHOULD HAVE SET THE
;CNTRL RDY BIT IN RKCS
;DID NXM BIT IN RKER SET?
2\$: BIT #2000,@RKER ;YES, BRANCH
BNE 3\$;GO GET RKCS, RKER
JSR PC,GT2RG ;THIS BIT (NXM) DID NOT SET IN RKER
MOV #2000,\$REG2 ;NXM BIT DID NOT SET IN RKER ON
ERROR 105 ;SIMULATING NXM CONDITION.
;DID HE & ERR BIT SET?
3\$: CMP #140266,@R0 ;YES, BRANCH
BEQ 4\$;GO, SET RKCS, RKER
JSR PC,GT2RG ;HE OR ERR BIT DID NOT SET WHEN
ERROR 106 ;NXM ERROR WAS SIMULATED
;CLEAR NXM, HE, ERR BITS
4\$: CNT.RESET ;GO, DO CONTROL RESET
;THIS IS A CALL FOR THE 'CNTRL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.

5230							: IF THE 'CNTRL RDY' DOES NOT SET
5231							: AN ERROR IS REPORTED. NOTE THAT
5232							: THE PC IN ERROR MESSAGE IS THE
5233							: PC WHERE 'CNT.RESET' IS LOCATED.
5234							: THIS IS A VERY BASIC ERR& IF IT
5235							: OCCURS GO BACK TO TEST 10
5236	016132	004737	021436			JSR	PC,TSTRWS : GO CHECK & WAIT FOR R/W/S RDY
5237							: TO SET. IF SET SKIP ERROR BELOW
5238	016136	104016				ERROR	16 : R/W/S RDY IS NOT SET
5239	016140	104421				TST.SIN	: CHECK IF SIN IS SET, IF SET
5240							: DO DRIVE RESET TO CLR IT
5241	016142	010277	163172			MOV	R2,@RKDA : ADRES DRIVE, NXC CYLINDER
5242	016146	012710	000011			MOV	#11,@R0 : SEEK, GO TO NXC CYL
5243	016152	104412				CHKCRDY	: GO CHECK IF CONTROL RDY IS SET
5244							: IF SO, SKIP THE EROR MESSAGE.
5245	016154	104021				ERROR	21 : SEEK WAS TRIED TO A NON EXISTENT
5246							: CYLINDER, NXC SHOULD HAVE OCCURED
5247							: SETTING CNTRL RDY. BUT CNTRL RDY
5248							: DID NOT SET.
5249	016156	032777	000100	163144	9\$:	BIT	#100,@RKER : DID NXC SET?
5250	016164	001020				BNE	4\$: YES, BRANCH
5251	016166	004737	021010			JSR	PC,GT2RG : GO GET RKCS, ER
5252	016172	017737	163142	001166		MOV	@RKDA,\$REG2 : GET RKDA
5253	016200	104110				ERROR	110 : NXC DID NOT SET WHEN SEEK
5254							: WAS TRIED TO CYLINDER AS INDICATED
5255							: IN RKDA
5256	016202	004737	021436			JSR	PC,TSTRWS : CHECK & WAIT FOR R/W/S RDY.
5257							: IF SET SKIP ERROR
5258	016206	104016				ERROR	16 : R/W/S SHOULD BE SET
5259	016210	104413				CNT.RESET	: GO DO CONTROL RESET
5260	016212	004737	021504			JSR	PC,DRESET : GO DO DRIVE RESET
5261	016216	104026				ERROR	26 : NXC DID NOT SET AND DRIVE MAY
5262							: HAVE TRIED TO DO A SEEK, AFTER
5263							: WHICH R/W/S RDY DID NOT SET
5264	016220	005237	001362			INC	COUNT : ALLOW ONLY 5 MESSAGES FOR
5265	016224	001405				BEQ	5\$: ERROR 133
5266	016226	062702	000040	4\$:		ADD	#40,R2 : ADRES THE NXT CYL(IN NON-EXISTENT ZONE)
5267	016232	032702	017740			BIT	#17740,R2 : CHKD FOR ALL NXC'S?
5268	016236	001334				BNE	3\$: IF NOT, LUP BAK & CHK THE NXT NXC
5269							
5270	016240	032710	140000	5\$:		BIT	#140000,@R0 : DID HE & ERR BIT SET WHEN NXC BIT SET?
5271	016244	001003				BNE	6\$: YES, BRANCH
5272	016246	004737	021010			JSR	PC,GT2RG : GET RKCS, ER
5273	016252	104106				ERROR	106 : HE OR ERR BIT DID NOT SET IN RKCS
5274							: WHEN NXC ERROR WAS SIMULATED
5275							: CLEAR HE, ERR, NXC BITS
5276	016254	104413		6\$:		CNT.RESET	: GO, DO CONTROL RESET
5277							: THIS IS A CALL FOR THE 'CNTRL-
5278							: RESET' ROUTINE. A CONTROL RESET IS
5279							: ISSUED AND AFTER A CERTAIN TIME
5280							: IF THE 'CNTRL RDY' DOES NOT SET
5281							: AN ERROR IS REPORTED. NOTE THAT
5282							: THE PC IN ERROR MESSAGE IS THE
5283							: PC WHERE 'CNT.RESET' IS LOCATED.
5284							: THIS IS A VERY BASIC ERR& IF IT
5285							: OCCURS GO BACK TO TEST 10

5286 016256 004737 021356
5287
5288 016262 104102
5289
5290 016264 032710 140000
5291 016270 001405
5292 016272 010037 001162
5293 016276 011037 001164
5294 016302 104102
5295
5296
5297
5298
5299
5300
5301
5302
5303
5304 016304 000004
5305 016306 104413
5306
5307
5308
5309
5310
5311
5312
5313
5314
5315 016310 013700 001332
5316 016314 013777 001350 163016
5317 016322 052777 000014 163010
5318 016330 012777 177777 162776
5319 016336 012777 033342 162772
5320 016344 012710 000005
5321 016350 104414
5322
5323
5324
5325
5326
5327
5328 016352 017702 162752
5329 016356 032702 000040
5330 016362 001006
5331 016364 004737 021010
5332 016370 012737 000040 001166
5333 016376 104105
5334
5335 016400 042702 000040
5336 016404 001407
5337
5338 016406 012737 000040 001162
5339 016414 017737 162710 001164
5340 016422 104107
5341

JSR PC,CHKECLR ;CHECK IF 'NXC' BIT WAS CLEARED BY
;CON.RESET. IF NOT, RETURN HERE.
ERROR 102 ;CNTRL RESET DID NOT CLEAR
;NXC BIT IN RKER.
7\$: BIT #140000,@R0 ;DID HE & ERR BITS GET CLEARED?
BEQ TST46 ;:YES, EXIT
MOV R0,\$REG0 ;GET ADRES OF RKCS
MOV @R0,\$REG1 ;GET RKCS CONTENTS
ERROR 102 ;CNTRL RESET DID NOT CLEAR
;HE OR ERR BIT IN RKCS
:*****
:*TEST 46 SIMULATE & CHECK NXS ERROR
;*THIS TEST SIMULATES NON-EXISTENT SECTOR ERROR & CHECKS THAT
;*IT IS DETECTED BY NXS BIT OF RKER. IT IS CHECKED THAT
;*WHEN NXS SETS HE & ERR OF RKER ALSO SETS, AND ALL THREE
;*CAN BE CLEARED BY CONTROL RESET.
:*****
TST46: SCOPE ;GO, DO CONTROL RESET
CNT.RESET ;THIS IS A CALL FOR THE 'CNTRL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.
;THIS IS A VERY BASIC ERR& IF IT
;OCCURS GO BACK TO TEST 10
MOV RKCS,R0 ;GET ADRES OF DRIVE
MOV DRIVAD,@RKDA ;SET BITS FOR SECTOR 12 (DECIMAL)
BIS #14,@RKDA ;READ 1 WORD
MOV #-1,@RKWC ;INTO THIS BUS ADRES
MOV #OUTBUF,@RKBA ;READ, GO (FROM NX SECTOR)
MOV #5,@R0 ;THIS IS A CALL FOR 'CN.RDY'
CNT.RDY ;ROUTINE WHICH WAITS FOR CNT
;RDY TO SET. IF CNTRL RDY DOES
;NOT SET WITHIN 883 MS/ 11-20
;(176 MS FOR 11-45 WITH BIPOLAR)
;AN ERROR IS REPORTED
;NXS ERROR SHOULD OCCUR NOW
MOV @RKER,R2 ;DID NXS BIT SET IN RKER?
BIT #40,R2 ;YES, BRANCH
BNE 1\$;GO GET RKCS, RKER
JSR PC,GT2RG ;THIS BIT (NXS) IN RKER DID NOT SET
MOV #40,\$REG2 ;NXS BIT DID NOT SET ON SIMULATING
ERROR 105 ;NYS ERROR
1\$: BIC #40,R2 ;MASK NXS BIT
BEQ 2\$;CHECK IF ANY OTHER
;RKER BIT SET
MOV #40,\$REG0 ;GET EXPCTD RKER
MOV @RKER,\$REG1 ;GET RKER RECVD
ERROR 107 ;ONLY 'NXS' SHOULD BE SET
;IN RKER, ANOTHER RKER BIT

```

5342                                     ;WAS SET. (NOTE 'NXS' WAS
5343                                     ;SIMULATED)
5344 016424 022710 140204                2$:  CMP      #140204,R0      ;DID HE & ERR BITS SET?
5345 016430 001403                        BEQ      3$          ;YES, BRANCH
5346 016432 004737 021010                JSR      PC,GT2RG   ;GO GET RKCS, RKER
5347 016436 104106                        ERROR    106        ;HE OR ERR BIT DID NOT SET WHEN
5348                                     ;NXS ERROR OCCURED
5349                                     ;CLEAR NXS, HE, ERR BITS
5350 016440 104413                        3$:  CNT.RESET     ;GO, DO CONTROL RESET
5351                                     ;THIS IS A CALL FOR THE 'CNTRL-
5352                                     ;RESET' ROUTINE. A CONTROL RESET IS
5353                                     ;ISSUED AND AFTER A CERTAIN TIME
5354                                     ;IF THE 'CNTRL RDY' DOES NOT SET
5355                                     ;AN ERROR IS REPORTED. NOTE THAT
5356                                     ;THE PC IN ERROR MESSAGE IS THE
5357                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
5358                                     ;THIS IS A VERY BASIC ERR& IF IT
5359                                     ;OCCURS GO BACK TO TEST 10
5360 016442 004737 021356                JSR      PC,CHKECLR ;CHECK IF 'NXS' BIT WAS CLEARED BY
5361                                     ;CON.RESET. IF NOT, RETURN HERE.
5362 016446 104102                        ERROR    102        ;CNTRL RESET DID NOT CLEAR
5363                                     ;NXS BIT IN RKER
5364 016450 004737 021402                4$:  JSR      PC,CHKCCLR ;CHECL IF 'HE' & 'ERR' BITS WERE CLEARED
5365                                     ;BY CON.RESET. IF NOT, RETURN HERE.
5366 016454 104102                        ERROR    102        ;RKCS BITS ERR OR HE WERE NOT
5367                                     ;CLEARED BY CNTRL RESET
5368
5369                                     ;:*****
5370                                     ;*TEST 47      SIMULATE & CHECK WCE
5371                                     ;*THIS TEST SIMULATES A WRITE CHECK ERROR AND CHECKS THAT IT
5372                                     ;*IS DETECTED BY WCE BIT OF RKER. FOR COMPARISON IT USES
5373                                     ;*THE 256 WORDS DATA BLOCK WRITTEN ON SECTOR 0, CYLINDER 0
5374                                     ;*IN A PREVIOUS TEST. THIS BLOCK IS COMPARED WITH THE 256 WORDS
5375                                     ;*MEMORY BUFFER STARTING AT 'OUTBUF'. WCE IS SIMULATED BY
5376                                     ;*DROPPING A BIT FROM ONE OF THE WORDS IN THE MEMORY BUFFER.
5377                                     ;:*****
5378 016456 000004                        TST47: SCOPE
5379 016460 013700 001332                MOV      RKCS,R0
5380 016464 104413                        CNT.RESET     ;GO, DO CONTROL RESET
5381                                     ;THIS IS A CALL FOR THE 'CNTRL-
5382                                     ;RESET' ROUTINE. A CONTROL RESET IS
5383                                     ;ISSUED AND AFTER A CERTAIN TIME
5384                                     ;IF THE 'CNTRL RDY' DOES NOT SET
5385                                     ;AN ERROR IS REPORTED. NOTE THAT
5386                                     ;THE PC IN ERROR MESSAGE IS THE
5387                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
5388                                     ;THIS IS A VERY BASIC ERR& IF IT
5389                                     ;OCCURS GO BACK TO TEST 10
5390 016466 104421                        TST.SIN
5391                                     ;CHECK IF SIN IS SET, IF
5392 016470 012701 033342                MOV      #OUTBUF,R1 ;SET DO DRV-RESET TO CLR IT
5393 016474 012702 177400                MOV      #-400,R2   ;THIS CODE SETS UP A MEMORY
5394 016500 012703 177777                MOV      #177777,R3 ;BUFFER OF 256 WORDS STARTING
5395                                     ;AT OUTBUF
5396                                     ;FIRST WORD 177400
5397 016504 062703 177401                1$:  ADD      #177401,R3 ;SECOND 177001
    
```



```
5454 ;THE PC IN ERROR MESSAGE IS THE
5455 ;PC WHERE 'CNT.RESET' IS LOCATED.
5456 ;THIS IS A VERY BASIC ERR& IF IT
5457 ;OCCURS GO BACK TO TEST 10
5458 016640 104421 TST.SIN ;CHECK IF SIN IS SET, IF
5459 ;SET DO DRIVE RESET TO CLR IT
5460 016642 013700 001332 MOV RKCS,R0
5461 016646 012737 170007 033360 MOV #170007,OUTBUF+16 ;WCE IS SIMULATED BY DROPPING A BIT
5462 ;IN THE EIGHTH WORD (WHICH IS ACTUALLY
5463 ;174007). NOTE THAT 256 WORD MEMORY
5464 ;BUFFER IS CREATED IN THE PREVIOUS TEST.
5465 016654 013701 001350 MOV DRIVAD,R1
5466 016660 012777 177000 162446 MOV #-1000,@RKWC ;WRT CHK 1000 (OCTAL) WORDS, 2 SECTORS
5467 016666 012777 033342 162442 MOV #OUTBUF,@RKBA ;FROM THIS BUS ADRES
5468 016674 010177 162440 MOV R1,@RKDA ;WITH THIS DISK ADRES, SEC 0, CYL 0
5469 016700 012710 000407 MOV #407,@R0 ;WRT CHK, GO, SSE
5470 016704 104412 CHKCRDY ;GO CHECK IF CONTROL RDY IS SET
5471 ;IF SO, SKIP THE EROR MESSAGE.
5472 016706 104065 ERROR 65 ;CNTRL RDY DID NOT SET AFTER WRT
5473 ;CHK. A SOFT ERROR (WCE) IN
5474 ;SECTOR 0 SHOULD HAVE STOPPED
5475 ;ALL CONTROL ACTION.
5476 016710 022777 000001 162412 2$: CMP #1,@RKER ;CHECK ONLY 'WCE' BIT SHOULD
5477 ;BE SET?
5478 016716 001407 BEQ 3$ ;YES, BRANCH
5479 016720 012737 000001 001162 MOV #1,$REG0 ;GET EXPCTD RKER
5480 016726 017737 162376 001164 MOV @RKER,$REG1 ;GET RKER RECVD
5481 016734 104107 ERROR 107 ;ONLY BIT 'WCE' OF RKER
5482 ;SHOULD BE SET (WCE WAS
5483 ;SIMULATED ABOVE). ERROR
5484 ;IF IT'S NOT
5485 016736 005201 3$: INC R1 ;CHECK THAT RKDA INCREMENTED BY
5486 016740 020177 162374 CMP R1,@RKDA ;1 SECTOR ONLY IMPLYING THAT
5487 ;CNTRL ACTION DID STOP AFTER
5488 ;SOFT ERROR IN SECTOR 0
5489 016744 001406 BEQ TST51 ;:YES, EXIT
5490 016746 010137 001162 MOV R1,$REG0 ;GET EXPCTD RKDA
5491 016752 017737 162362 001164 MOV @RKDA,$REG1 ;GET RKDA RECVD
5492 016760 104070 ERROR 70 ;RKDA SHOULD HAVE INCRMNTD
5493 ;BY 1 SECTOR ONLY, IT DIDN'T.
5494 ;WCE WAS SIMULATED IN THE
5495 ;FIRST SECTOR & A WRT CHK
5496 ;OF 2 SECTORS WAS ISSUED.
5497 ;CONTROLLER SHOULD STOP AFTER
5498 ;DETECTING WCE IN THE FIRST
5499 ;SECTOR. HENCE RKDA SHOULD
5500 ;INCREMENT BY 1 SECTOR ONLY
5501
5502
5503 ;*****
5504 ;*TEST 51 CHECK THAT RK11 INTERRUPTS ON SOFT ERROR WHEN SSE & IDE ARE SET
5505 ;*THIS TEST CHECKS WHEN SSE BIT IS SET WITH IDE SET AND A SOFT
5506 ;*ERROR OCCURS, THEN ALL CONTROL ACTION WILL STOP AND A BUS
5507 ;*REQUEST (INTERRUPT) WILL OCCUR AT THE END OF THE CURRENT
5508 ;*SECTOR. SOFT ERROR IS SIMULATED BY WCE AS IN PREVIOUS
5509 ;*TEST. PREREQUISITES FOR THIS TEST ARE THE, SAME AS THOSE
```

```

5510                                     :*FOR THE PREVIOUS TEST.
5511                                     :*****
5512 016762 000004 ' TST51: SCOPE
5513 016764 104413      CNT.RESET
5514                                     ;GO, DO CONTROL RESET
5515                                     ;THIS IS A CALL FOR THE 'CNTRL-
5516                                     ;RESET' ROUTINE. A CONTROL RESET IS
5517                                     ;ISSUED AND AFTER A CERTAIN TIME
5518                                     ;IF THE 'CNTRL RDY' DOES NOT SET
5519                                     ;AN ERROR IS REPORTED. NOTE THAT
5520                                     ;THE PC IN ERROR MESSAGE IS THE
5521                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
5522                                     ;THIS IS A VERY BASIC ERR& IF IT
5523 016766 104421      TST.SIN
5524                                     ;OCCURS GO BACK TO TEST 10
5525 016770 012737 170007 033360      MOV #170007,OUTBUF+16
5526                                     ;CHECK IF SIN IS SET, IF
5527                                     ;SET DO DRIVE RESET TO CLR IT
5528                                     ;WCE IS SIMULATED BY DROPPING A BIT
5529                                     ;IN THE EIGHTH WORD (WHICH IS 174007)
5530 016776 013701 001350      MOV DRIVAD,R1
5531 017002 012777 177000 162324      MOV #-1000,@RKWC
5532 017010 012777 033342 162320      MOV #OUTBUF,@RKBA
5533 017016 010177 162316      MOV R1,@RKDA
5534 017022 013700 001402      MOV RKVEC,R0
5535 017026 012720 017060      MOV #1$(,R0)+
5536 017032 012710 000340      MOV #340,@R0
5537 017036 012777 000507 162266      MOV #507,@RKCS
5538 017044 104420 177777      WAT.INT,177777
5539                                     ;WRT CHK 1000 (OCTAL) WORDS, 2 SECTORS
5540                                     ;FROM THIS BUS ADRES
5541 017050 004737 021010      JSR PC,GT2RG
5542 017054 104111      ERROR 111
5543                                     ;WITH THIS DISK ADRES, SEC 0, CYL 0
5544 017056 000417      BR 2$
5545                                     ;SET UP INTERRUPT VECTOR FOR RK11
5546 017060 022626 1$:      CMP (SP)+,(SP)+
5547 017062 022626      CMP (SP)+,(SP)+
5548 017064 012777 004600 162310      MOV #BADINT,@RKVEC
5549                                     ;SET PSW ON INTERRUPT
5550 017072 005201      INC R1
5551 017074 020177 162240      CMP R1,@RKDA
5552                                     ;WRT CHK, GO. SSE, IDE SET
5553                                     ;WAIT FOR INTERRUPT FROM RK11
5554 017100 001406      BEQ 2$
5555 017102 010137 001162      MOV R1,$REG0
5556 017106 017737 162226 001164      MOV @RKDA,$REG1
5557 017114 104003      ERROR 3
5558                                     ;WAIT FOR INTERRUPT FROM RK11
5559                                     ;TIME=485 MS FOR 11/20,
5560                                     ;97 MS FOR 11/45
5561 017116 2$:      MOV #340,-(SP)
5562 017116 012746 000340      MOV #64$,-(SP)
5563 017122 012746 017130      RTI
5564 017126 000002
5565 017130 64$:
  
```

!

5566 017130 005077 162176
5567
5568
5569
5570
5571
5572
5573
5574
5575
5576
5577
5578 017134 000004
5579 017136 013700 001332
5580 017142 012701 177774
5581 017146 005002
5582 017150 012737 017156 001110
5583
5584 017156 104417 000142
5585 017162 004737 021436
5586 017166 104016
5587 017170 104413
5588
5589
5590
5591
5592
5593
5594
5595
5596
5597 017172 010210
5598 017174 012777 177777 162132
5599 017202 013777 001350 162130
5600 017210 012777 177776 162120
5601
5602 017216 052710 000007
5603
5604
5605
5606 017222 104412
5607
5608 017224 104065
5609 017226 010205
5610 017230 062705 000020
5611 017234 042705 000100
5612 017240 011004
5613 017242 042704 177717
5614 017246 020504
5615 017250 001405
5616 017252 010537 001162
5617 017256 010437 001164
5618 017262 104112
5619
5620
5621

CLR @RKCS ;CLEAR THE IDE BIT

: *TEST 52 CHECK THE MEX BITS IN RKCS
: *THIS TEST CHECKS OUT THE EXTENDED MEMORY BITS OF THE RKCS.
: *THE RKBA IS SET TO 177776 AND A ONE WORD WRITE CHECK IS TRIED.
: *THIS COULD GIVE RISE TO NXM ERROR, BUT EVEN THEN THE RKBA
: *SHOULD OVERFLOW INTO THE MEX BITS. SIMILIARLY IT IS CHECKED
: *THAT THE OVERFLOWING BIT CAN MAKE THE MEX BITS COUNT
: *01,10,11,00.

TST52: SCOPE
MOV RKCS,R0 ;SET UP THE COUNT
MOV #-4,R1 ;INITIALIZE MEX BITS TO B SET IN RKCS
CLR R2 ;SET RETURN ADRES FOR
MOV #1\$, \$LPERR ;LUPING ON EROR (SW9)

1\$: DELAY ,142 ;TIME DELAY
JSR PC,TSTRWS ;WAIT FOR R/W/S RDY
ERROR 16 ;R/W/S RDY IS NOT SET
CNT.RESET ;GO, DO CONTROL RESET
;THIS IS A CALL FOR THE 'CNTPL-
;RESET' ROUTINE. A CONTROL RESET IS
;ISSUED AND AFTER A CERTAIN TIME
;IF THE 'CNTRL RDY' DOES NOT SET
;AN ERROR IS REPORTED. NOTE THAT
;THE PC IN ERROR MESSAGE IS THE
;PC WHERE 'CNT.RESET' IS LOCATED.
;THIS IS A VERY BASIC ERR& IF IT
;OCCURS GO BACK TO TEST 10
MOV R2,@R0 ;SET MEX BITS (AS IN R2) IN RKCS
MOV #-1,@RKWC ;WRT CHK 1 WORD
MOV DRIVAD,@RKDA ;THIS DISK ADRES, SEC 0, CYL 0
MOV #177776,@RKBA ;THIS BUS ADRES. NOTE THIS BA
;IN CONJUCTION WITH MEX BITS OF RKCS
BIS #7,@R0 ;WRT CHK, GO
;THERE MAY BE A NXM OR WCE BUT
;WHATEVER THE CASE RKBA SHOULD
;OVERFLOW MAKING THE MEX BITS COUNT
;GO CHECK IF CONTROL RDY IS SET
;IF SO, SKIP THE EROR MESSAGE.
;CNTRL RDY DID NOT SET AFTER WRT CHK

3\$: ERROR 65
MOV R2,R5 ;MEX BITS SHOULD INCREMENT BY 1 TO THIS
ADD #20,R5 ;MASK OUT IDE BIT POSITION, IF SET
BIC #100,R5 ;GET RKCS
MOV @R0,R4 ;MASK OUT ALL BITS EXCEPT MEX
BIC #177717,R4 ;DID MEX BITS INCREMENT CORRECTLY?
CMP R5,R4 ;YES, BRANCH
BEQ 4\$;GET EXPCTD MEX BITS
MOV R5,\$REGO ;GET MEX BITS RECVD
MOV R4,\$REG1 ;MEX BITS DID NOT INCREMENT AS
ERROR 112 ;'EXPCTD' WHEN RKBA OVERFLOWED.
;NOTE THAT BIT POSITION 4 & 5
;REFLECT MEX BITS 0 & 1 IN THE

```
5622                                     ;ERROR MESSAGE.
5623 017264 017703 162040 4$: MOV @RKER,R3 ;GET RKER
5624 017270 010305 MOV R3,R5
5625 017272 042703 003001 BIC #3001,R3 ;MASK WCE,DLT,NXM BIT, IF SET
5626 017276 001410 BEQ 5$ ;BRANCH IF REST OF RKER CLR
5627 017300 042705 177776 BIC #177776,R5 ;MASK NON-WCE BITS
5628 017304 010537 001162 MOV R5,$REGO ;THIS IS THE EXPCTD RKER
5629 017310 017737 162014 001164 MOV @RKER,$REG1 ;GET RKER RECVD
5630 017316 104107 ERROR 107 ;ERROR IN RKER. IT SHOULD
5631                                     ;BE AS EXPECTED IN
5632                                     ;ERROR MESSAGE
5633 017320 062702 000020 5$: ADD #20,R2 ;INCREMENT TO NXT MEX BIT
5634 017324 005201 INC R1 ;HAVE U CHKD THE MEX BITS 4 TIMES?
5635 017326 001313 BNE 1$ ;IF NOT, LUP BACK
5636
5637 ::*****
5638 ;*TEST 53 TRANSFER FROM DISK TO TTY
5639 ;* THIS TEST CHECKS THE HIGH ORDER BITS OF THE ADDRESS
5640 ;* LINES. FIRST A ONE WORD (100) IS WRITTEN ON SECTOR,
5641 ;* 2, CYL 0. THEN IT IS READ BACK, BUT THE NPR IS DONE
5642 ;* NOT TO THE MEMORY, BUT THE TELETYPE BUFFER (TKS 177560)
5643 ;* AND IT CHECKED THAT THE WORD WAS RECIEVED CORRECTLY.
5644 ;*IF IT IS NOT, AN ERROR IS REPORTED. THIS TEST IS
5645 ;*SKIPPED ON AN 11/05.
5646 ::*****
5647 017330 000004 TST53: SCOPE
5648 017332 012737 000001 001206 MOV #1,$TIMES ;:DO 1 ITERATION
5649                                     ;THIS CODE FINDS OUT IF THE CPU
5650                                     ;IS AN 11/05 OR ELSE.
5651                                     ;ON AN 11/05, RO (177700) CAN BE
5652                                     ;ADDRESSED AS A MEMORY LOCATION, BUT
5653                                     ;ON ANY OTHER CPU IF 177700 IS REFERENCED
5654                                     ;A TIME OUT WILL OCCUR.
5655 017340 012737 017362 000004 MOV #5$,@#4 ;SET UP TIME OUT VECTOR
5656 017346 005737 177700 TST @#177700 ;REFERENCE RO
5657 017352 012737 004534 000004 MOV #BADTMO,@#4 ;RO WAS REFERENCED W/O TIMEOUT
5658                                     ;HENCE 11/05
5659 017360 000520 BR TST54 ;:SKIP THIS TEST
5660 017362 022626 5$: CMP (SP)+,(SP)+ ;RESTORE STACK POINTER
5661 017364 012737 004534 000004 MOV #BADTMO,@#4 ;RESTORE TIMEOUT VECTOR
5662 017372 012746 000340 MOV #340,-(SP)
5663 017376 012746 017404 MOV #64$,-(SP)
5664 017402 000002 RTI
5665 017404 64$:
5666 017404 013700 001332 MOV RKCS,RO
5667 017410 104413 CNT.RESET ;GO, DO CONTROL RESET
5668                                     ;THIS IS A CALL FOR THE 'CNTRL-
5669                                     ;RESET' ROUTINE. A CONTROL RESET IS
5670                                     ;ISSUED AND AFTER A CERTAIN TIME
5671                                     ;IF THE 'CNTRL RDY' DOES NOT SET
5672                                     ;AN ERROR IS REPORTED. NOTE THAT
5673                                     ;THE PC IN ERROR MESSAGE IS THE
5674                                     ;PC WHERE 'CNT.RESET' IS LOCATED.
5675                                     ;THIS IS A VERY BASIC ERR& IF IT
5676                                     ;OCCURS GO BACK TO TEST 10
5677 017412 012701 033342 MOV #OUTBUF,R1
```



```

5678 017416 013704 001336      MOV      RKBA,R4
5679 017422 012711 000100      MOV      #100,@R1      ;WRITE THIS WORD
5680 017426 012777 177777 161700  MOV      #-1,@RKWC     ;WRITE 1 WORD
5681 017434 013702 001350      MOV      DRIVAD,R2
5682 017440 052702 000002      BIS      #2,R2        ;ON CYL 0, SEC 2
5683 017444 010277 161670      MOV      R2,@RKDA
5684 017450 010114      MOV      R1,@R4      ;FROM THIS MEMORY LOC
5685 017452 012710 000003      MOV      #3,@R0      ;WRITE, GO
5686 017456 005003      CLR      R3
5687 017460 105710 1$:      TSTB    @R0
5688 017462 100410      BMI      2$
5689 017464 005203      INC      R3
5690 017466 001374      BNE      1$
5691 017470 004737 020774      JSR      PC,GT4RG     ;GET RKCS, ER, DS
5692 017474 010237 001202      MOV      R2,$REG10   ;GET THE STARTING ADRES
5693 017500 104416      BRKDA4   ;BREAK IT INTO DRV #, CYL, SUR, SEC #
5694 017502 104031      ERROR    31         ;CNTRL RDY DID NOT SET AFTER
5695                                     ;WRITE OF 1 WORD ON CYL 0, SEC 2
5696 017504 012777 177777 161622 2$:      MOV      #-1,@RKWC     ;READ 1 WORD
5697 017512 010277 161622      MOV      R2,@RKDA     ;FROM SEC 2, CYL 0
5698 017516 013714 001144      MOV      $TKS,@R4     ;INTO TTY STAU$ REGISTER
5699 017522 005077 161416      CLR      @$TKS        ;CLEAR TTY KEY BRD STATUS REG
5700
5701 017526 012710 000065      MOV      #65,@R0     ;READ, MEX BITS SET
5702 017532 005003      CLR      R3
5703 017534 105710 3$:      TSTB    @R0
5704 017536 100410      BMI      4$
5705 017540 005203      INC      R3
5706 017542 001374      BNE      3$
5707 017544 004737 020774      JSR      PC,GT4RG     ;GET THE STARTING ADRES
5708 017550 010237 001202      MOV      R2,$REG10   ;BREAK IT INTO DR#, CYL, SUR, SEC#
5709 017554 104416      BRKDA4   ;CNTRL RDY DIDN'T SET AFTER
5710 017556 104045      ERROR    45         ;READ OF 1 WORD FROM CYL 0, SEC 2.
5711                                     ;IN EROR MSGE, <DSK-ADRES> GIVES
5712                                     ;ADRES WHERE READ BEGAN. 'RKDA'
5713                                     ;GIVES CONTENTS OF RKDA AT TIME OF EROR
5714                                     ;WAS THE CORRECT WORD READ INTO
5715 017560 032737 000100 001144 4$:      BIT.     #100,$TKS   ;THE TTY STATUS REGISTER?
5716                                     ;:YES, EXIT
5717 017566 001015      BNE      TST54
5718 017570 017705 161350      MOV      @$TKS,R5     ;GET THE WORD RECVD FROM DISK
5719 017574 010537 001164      MOV      R5,$REG1
5720 017600 052705 000100      BIS      #100,R5     ;THIS WORD WAS EXPCTD
5721 017604 010537 001162      MOV      R5,$REG0     ;STORE EXPCTD WORD
5722 017610 011437 001166      MOV      @R4,$REG2   ;GET RKBA
5723 017614 011037 001170      MOV      @R0,$REG3   ;GET RKCS
5724 017620 104115      ERROR    115        ;DATA ERROR. A ONE WORD (100)
5725                                     ;NPR WAS TRIED FROM DISK TO
5726                                     ;TTY KEYBOARD STATUS REGISTER
5727                                     ;(17756) . BIT 6 SHOULD HAVE BEEN
5728                                     ;SET AS RESULT OF THIS
5729                                     ;BUT IT WAS NOT
5730
5731
5732
5733
;*****
;*TEST 54      CHECK THAT RKBA CAN COUNT CORRECTLY

```

```
5734                                     : *THIS TEST CHECKS THAT RKBA CAN COUNT CORRECTLY. IT IS SET
5735                                     : *TO THE DESIRED INITIAL VALUE. THEN A ONE WORD WRITE CHECK
5736                                     : *IS TRIED, WITH MEX (MEMORY EXTENSION) BITS SET. IF THERE IS
5737                                     : *NO MEMORY PRESENT (FOR CERTAIN BUS ADDRESSES), THERE
5738                                     : *WILL BE AN NXM ERROR STOPPING CONTROLLER ACTION. BUT RKBA
5739                                     : *SHOULD HAVE INCREMENTED BY 1 FROM ITS INITIAL VALUE. IF IT
5740                                     : *HAS NOT, AN ERROR IS REPORTED.
5741                                     ;*****
5742 017622 000004 TST54: SCOPE
5743 017624 012737 000005 001206 MOV #5,$TIMES ;DO 5 ITERATIONS
5744 017632 104421 TST.SIN ;CHECK IF SIN SET, IF SET DRV RESET
5745 017634 005001 CLR R1 ;INITIALIZE (VALUE OF RKBA)
5746 017636 012702 000002 MOV #2,R2 ;INITIALIZE (INCMNTD VALUE OF RKBA)
5747
5748 017642 012737 017654 001110 MOV #18,$LPERR ;SET RETURN ADRES FOR LUPING
5749 ;ON EROR
5750
5751 017650 013705 001336 MOV RKBA,R5
5752 017654 004737 021436 1$: JSR PC,TSTRWS ;WAIT FOR R/W/S RDY
5753 017660 104016 ERROR 16 ;R/W/S RDY IS NOT SET
5754 017662 104413 CNT.RESET ;DO CONTROL RESET
5755 017664 012777 177777 161442 MOV #-1,@RKWC ;WRITE CHK 1 WORD
5756 017672 010115 MOV R1,@R5 ;THIS BUS ADRES
5757 017674 013777 001350 161436 MOV DRIVAD,@RKDA ;SET DISK ADRES
5758 017702 012777 000067 161422 MOV #67,@RKCS ;WRITE CHECK, GO, MEX BITS SET
5759 017710 104412 CHKCRDY ;GO CHECK IF CONTROL RDY IS SET
5760 ;IF SO, SKIP THE EROR MESSAGE.
5761 017712 104065 ERROR 65 ;CNTRL RDY DID NOT SET AFTER
5762 ;WRT CHK WAS TRIED TO NXM LOC
5763 ;U MIGHT WANT TO USE TESTS
5764 ;CHECKING MEX BITS & NXM.
5765 017714 005237 001356 INC INDX1 ;ALLOW ONLY 5 ERRORS OF ABOVE KIND
5766 017720 001417 BEQ 5$
5767
5768 017722 020215 3$: CMP R2,@R5 ;DID RKBA INCREMENT BY 1 FROM
5769 ;ITS INITIAL VALUE?
5770 017724 001410 BEQ 4$ ;YES, BRANCH
5771 017726 010137 001162 MOV R1,$REGO ;GET EXPCTD RKBA
5772 017732 011537 001164 MOV @R5,$REG1 ;GET RKBA RECVD
5773 017736 104017 ERROR 17 ;RKBA DID NOT INCREMENT BY
5774 ;1 FROM ITS INITIAL VALUE.
5775 ;ONE WORD WRT CHK WAS TRIED
5776 ;TO A NXM LOCATION. THERE
5777 ;WILL BE AN NXM ERROR,
5778 ;BUT STILL RKBA SHOULD
5779 ;INCREMENT BY 1 FROM ITS
5780 ;INITIAL VALUE.
5781 017740 005237 001360 INC INDX2 ;ALLOW ONLY 5 ERRORS OF
5782 017744 001405 BEQ 5$ ;THE ABOVE KIND
5783 017746 060201 4$: ADD R2,R1 ;SET NXT VALUE OF RKBA
5784 017750 010102 MOV R1,R2
5785 017752 062702 000002 ADD #2,R2 ;SET EXPCTD VALUE OF RKBA
5786 017756 001336 BNE 1$ ;ALL DONE?
5787
5788 017760 5$: ;DUMMY EXIT POINT
5789
```

5790
5791
5792
5793
5794
5795
5796
5797 017760 000004
5798 017762 012737 000001 001206
5799 017770 005737 001404
5800 017774 001403
5801 017776 004537 025160
5802 020002 104120
5803
5804 020004
5805
5806
5807
5808
5809
5810
5811
5812
5813
5814 020004 000004
5815 020006 012737 000001 001206
5816 020014 005237 001352
5817
5818 020020 004737 021504
5819 020024 104026
5820 020026 023737 001412 001352
5821
5822 020034 001405
5823 020036 062737 020000 001350
5824 020044 000137 005040
5825
5826 020050 005037 001112
5827
5828
5829
5830
5831
5832
5833
5834
5835
5836
5837
5838
5839
5840
5841
5842
5843
5844
5845 020054 000004

*TEST 55 CHECK FOR RK-05F
;*THIS TEST CHECKS RK-05F TYPE DRIVES
;*TO INSURE THAT IF SEEKS ARE ISSUED ON ONE
;*DRIVE, THE OTHER DRIVE BECOMES BUSY

TST55: SCOPE
MOV #1,\$TIMES ;;DO 1 ITERATION
TST FFLAG ;SEE IF RK-05F
BEQ 1\$;NOT F
JSR R5,FCHECK ;SEE IF OTHER GOES BUSY
ERROR 120

1\$:

*TEST 56 END OF PROGRAM
;*THIS IS NOT A TEST, BUT A LINKAGE PROVIDED TO PERFORM
;*THE ABOVE SUB-TESTS FOR ALL DRIVES THAT ARE PRESENT.
;*NOTE THAT THE NEXT TEST- HARDWARE POLLING LOGIC-
;*IS DONE USING ALL THE DRIVES THAT ARE INDICATED PRESENT.
;*DO NOT LOOP ON THIS 'TEST'.

TST56: SCOPE
MOV #1,\$TIMES ;;DO 1 ITERATION
INC DRVDON ;INCREMENT THE COUNT FOR THE NUMBER
;OF DRIVES THAT ARE CHECKED
JSR PC,DRESET ;RESET THE DRIVE
ERROR 26 ;R/W/S DIDN'T SET AFTER DRIVE RESET
BTECP: CMP DRIVS,DRVDON ;HAVE U TESTED ALL THE DRIVES
;THAT ARE PRESENT?
BEQ 1\$;IF YES, EXIT
ADD #20000,DRIVAD ;ADRES THE NXT POSSIBLE DRIVE
JMP NUDRV ;GO BACK AND TEST THE NEXT
;DRIVE PRESENT
1\$: CLR \$ERTTL

*TEST 57 CHECK HARDWARE POLLING LOGIC
;*THIS TEST CHECKS THE HARDWARE POLL LOGIC, USING ALL THE DRIVES
;*PRESENT ON THE RK11. ATLEAST TWO DRIVES SHOULD BE PRESENT
;*TO DO A MEANINGFUL HARDWARE POLL. SEQUENCE OF OPERATIONS IS
;*AS FOLLOWING:
;*1) NUMBER OF DRIVES ON THE RK11 IS ASCERTAINED.
;*2) HAVING LOCKED OUT ALL INTERRUPTS (CPU PR 7), SEEK IS INITIATED
;*FOR ONE DRIVE AT A TIME, ONLY WHEN 'CNTRL RDY' IS SET.
;*3) CPU PRIORITY IS DROPPED TO 4 SO THAT RK11 CAN INTERRUPT, THE INCOMING
;*INTERRUPT IS PROCESSED TO CHECK IF IT WAS DUE TO 'SEEK DONE' BY
;*ONE OF THE DRIVES.
;*4) IF BY THE END OF THE SET TIME A DRIVE HAS NOT INTERRUPTED
;*AN ERROR MESSAGE IS GIVEN INDICATING WHICH DRIVE DID NOT
;*INTERRUPT AFTER SEEK WAS DONE.

TST57: SCOPE

```

5846 020056 012737 000005 001206      MOV      #5,$TIMES      ;;DO 5 ITERATIONS
5847 020064 005237 001440              INC      SIZYET        ;;FOUNR RK05F YET?
5848 020070 001002              BNE     25$           ;;YES
5849 020072 004737 025304              JSR     PC,SIZF       ;;FIND WHICH ARE RK-05F
5850 020076 005037 001436      25$:    CLR      PHYDRV     ;;NUMBER OF ACTUAL DRIVES
5851 020102 012700 001414              MOV     #DRIVO,RO     ;;TABLE
5852 020106 005710      23$:    TST      (RO)       ;;DRIVE HERE+?
5853 020110 001405              BEQ     22$           ;;NO
5854 020112 005237 001436              INC     PHYDRV        ;;COUNT DRIVE
5855 020116 005710              TST     (RO)         ;;RK05F?
5856 020120 100001              BPL     22$           ;;NO
5857 020122 005720              TST     (RO)+        ;;DONT COUNT F TWICE
5858 020124 005720      22$:    TST     (RO)+ ;NEXT DRIVE
5859 020126 020027 001433              CMP     RO,#DRIV7+1  ;;ALL YET
5860 020132 002765              BLT     23$           ;;NO
5861 020134 005037 001406              CLR     ODDEVN        ;;EVEN DRIVES FIRST IF F
5862 020140 005737 001412      156$:   ST        DRIVS    ;;ANY DRIVES PRESENT?
5863 020144 001002              BNE     20$           ;;YES
5864 020146 000137 020652              JMP     $EOP          ;;NO
5865 020152 005237 001434      20$:    INC     T56FLG
5866 020156 013700 001332              MOV     RKCS,RO
5867 020162 005037 001356              CLR     INDX1        ;;FLAG TO INDICATE:
5868                                ;;(INDX1)=0 POLLING DONE AFTER ALL
5869                                ;;DRIVES SEEK TO CYL 0
5870                                ;;(INDX1)=1 POLLING DONE AFTER ALL
5871                                ;;DRIVES SEEK TO CYL 4
5872 020166 005037 001360      15$:    CLR     INDX2        ;;FLAG INDICATING TYPE OF INTERRUPT
5873                                ;;SET TO NON-ZERO TO INDICATE
5874                                ;;THAT THE INTERRUPT IS DUE TO
5875                                ;;SEEK DONE
5876 020172 104413              CNT.RESET           ;;GO, DO CONTROL RESET
5877                                ;;THIS IS A CALL FOR THE 'CNTRL-
5878                                ;;RESET' ROUTINE. A CONTROL RESET IS
5879                                ;;ISSUED AND AFTER A CERTAIN TIME
5880                                ;;IF THE 'CNTRL RDY' DOES NOT SET
5881                                ;;AN ERROR IS REPORTED. NOTE THAT
5882                                ;;THE PC IN ERROR MESSAGE IS THE
5883                                ;;PC WHERE 'CNT.RESET' IS LOCATED.
5884                                ;;THIS IS A VERY BASIC ERR& IF IT
5885                                ;;OCCURS GO BACK TO TEST 10
5886 020174 005737 001356              TST     INDX1        ;;PERFORMING SEEKS TO CYL 4
5887 020200 001002              BNE     .+6          ;;YES, BRANCH
5888 020202 005002              CLR     R2           ;;NO
5889 020204 000402              BR      .+6
5890 020206 012702 000200              MOV     #200,R2      ;;SET ADRES FOR FOURTH CYLINDER
5891 020212 012701 001414              MOV     #DRIVO,R1    ;;INITIALIZE POINTER
5892 020216 012703 177770              MOV     #-10,R3      ;;SET COUNT FOR 8 DRIVES
5893 020222 012705 033342              MOV     #OUTBUF,R5   ;;INITIALIZE POINTER TO INDICATOR AREA
5894 020226 005025              CLR     (R5)+        ;;CLEAR OUT THE 8-WORD INDICATOR
5895 020230 005203              INC     R3           ;;AREA WHICH IS USED FOR DOING
5896 020232 001375              BNE     .-4          ;;SOFTWARE POLLING LATER ON
5897 020234 012703 177770              MOV     #-10,R3      ;;SET COUNT FOR 8 POSSIBLE DRIVES
5898 020240 012705 033342              MOV     #OUTBUF,R5   ;;INITIALIZE POINTER TO INDICATOR AREA
5899 020244      1$:
5900 020244 012746 000340              MOV     #340,-(SP)
5901 020250 012746 020256              MOV     #64$,-(SP)
    
```



```
6014
6015 020560 105715
6016 020562 001006
6017 020564 011546
6018 020566 004737 021200
6019 020572 012637 001162
6020
6021 020576 104052
6022
6023 020600 062705 000002
6024 020604 005303
6025 020606 001364
6026
6027 020610 005737 001356
6028 020614 001004
6029 020616 005237 001356
6030 020622 000137 020166
6031
6032
6033
6034
6035
6036
6037
6038
6039
6040
6041
6042
6043
6044
6045
6046
6047
6048
6049
6050
6051
6052
6053
6054
6055 020626 005237 001406
6056 020632 022737 000002 001406
6057 020640 001402
6058 020642 000137 020140
6059 020646 005037 001434
6060
6061
6062
6063
6064
6065
6066
6067
6068
6069

14$: TSTB (R5) ;DID THIS DRIVE INTERRUPT?
      BNE 13$ ;YES, BRANCH
      MOV (R5),-(R6) ;GET THIS DRIVE #
      JSR PC,SHFTRT ;SHIFT IT TO THE RIGHT
      MOV (R6)+,$REGO ;THIS DRIVE # DID NOT INTERRUPT
                        ;DURING H'WARE POLL
                        ;DRIVE # (AS IN $REGO) DID NOT
15$: ADD #2,R5 ;INTERRUPT DURING HARDWARE POLL
      DEC R3 ;INCREMENT POINTER TO THE NEXT FLAG
      BNE 14$ ;CHKD FOR ALL DRIVES?
                        ;IF NOT LUP BACK

      TST INDX1 ;DONE POLLING FOR SEEKS TO CYL 312?
      BNE TSTEND ;IF YES, EXIT
      INC INDX1 ;IF NOT, INCREMENT FLAG
      JMP 15$ ;GO DO IT

;INDICATOR TABLE
;THE 8-WORD INDICATOR TABLE USED IN
;THE FORMER PART OF THIS SUB-TEST
;IS LOCATED STARTING AT 'OUTBUF'.
;WORDS ARE SET UP TO INDICATE
;PRESENCE OF A DRIVE EG: IF
;DRIVES 0,1,2 ARE PRESENT, IT WILL
;LOOK LIKE
;OUTBUF: 000000 BITS 13,14,15
;         020000 CONTAIN THE
;         040000 DRIVE NO.
;         000000 REST 0'S
;WHEN A DRIVE INTERRUPTS AFTER SEEK
;IS DONE BIT 0 OF THE CORRESPONDING
;INDICATOR WORD IS SET. THUS FOR THE
;ABOVE EXAMPLE IF ALL DRIVES INTERRUPTED
;CORRECTLY THEN IT WILL LOOK LIKE:
;         12$: 000001 BIT 0 SET
;         020001 TO INDICATE
;         040001 DR INTERRUPTED
;         000000 REST 0'S

TSTEND: INC ODDEVN ;NOW ODD IF RK05F
        CMP #2,ODDEVN ;SEE IF DONE
        BEQ 21$ ;ALL DONE
        JMP T56 ;TEST AGAIN
21$: CLR T56FLG

.SBTTL END OF PASS ROUTINE

;*****
;*INCREMENT THE PASS NUMBER ($PASS)
;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
;*TYPE 'END PASS #XXXXX' (WHERE XXXXX IS A DECIMAL NUMBER)
;*IF THERES A MONITOR GO TO IT
;*IF THERE ISN'T JUMP TO ST4
```

6070
 6071 020652
 6072 020652 000004
 6073 020654 005037 001102
 6074 020660 005037 001206
 6075 020664 005237 001100
 6076 020670 042737 100000 001100
 6077 020676 005327
 6078 020700 000001
 6079 020702 003022
 6080 020704 012737
 6081 020706 000001
 6082 020710 020700
 6083 020712 104401 020757
 6084 020716 013746 001100
 6085 020722 104405
 6086 020724 104401 020754
 6087 020730 013700 000042
 6088 020734 001405
 6089 020736 000005
 6090 020740 004710
 6091 020742 000240
 6092 020744 000240
 6093 020746 000240
 6094 020750
 6095 020750 000137
 6096 020752 004456
 6097 020754 377 377 000
 6098 020757 015 042412 042116
 6099 020764 050040 051501 020123
 6100 020772 000043

```

$EOP:
  SCOPE
  CLR $STNM      ;;ZERO THE TEST NUMBER
  CLR $TIMES     ;;ZERO THE NUMBER OF ITERATIONS
  INC $PASS      ;;INCREMENT THE PASS NUMBER
  BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
  DEC (PC)+      ;;LOOP?
$EOPCT: .WORD 1
  BGT $DOAGN     ;;YES
  MOV (PC)+,@(PC)+ ;;RESTORE COUNTER
$ENDCT: .WORD 1
  $EOPCT
  TYPE , $SENDMG ;;TYPE 'END PASS #'
  MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
  TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
  TYPE , $ENULL  ;;TYPE A NULL CHARACTER
$GET42: MOV @#42,R0 ;;GET MONITOR ADDRESS
  BEQ $DOAGN     ;;BRANCH IF NO MONITOR
  RESET         ;;CLEAR THE WORLD
$ENDAD: JSR PC,(R0) ;;GO TO MONITOR
  NOP          ;;SAVE ROOM
  NOP          ;;FOR
  NOP          ;;ACT11
$DOAGN: JMP @(PC)+ ;;RETURN
$RTNAD: .WORD ST4
$ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
$SENDMG: .ASCIZ <15><12>/END PASS #/
  
```

```

6101
6102
6103
6104
6105
6106
6107
6108      .SBTTL GT2RG: ROUTINE FOR GETTING RKCS,RKER
6109
6110      ;SUBROUTINE FOR TRANSFERRING THE CONTENTS OF RKCS, RKER
6111      ;TO $REG0, $REG1 RESPECTIVELY BEFORE TYPING OUT AN ERROR MESSAGE.
6112      ;CALL: JSR PC,GT2RG
6113
6114      .SBTTL GT3RG: ROUTINE FOR GETTING RKCS, RKER, RKDS
6115
6116      ;GT3RG
6117      ;SUBROUTINE FOR TRANSFERRING THE CONTENTS OF RKCS, RKER, RKDS
6118      ;TO $REG0, $REG1, $REG2 RESPECTIVELY BEFORE TYPING OUT AN
6119      ;ERROR MESSAGE.
6120      ;CALL: JSR PC,GT3RG
6121
6122      .SBTTL GT4RG: ROUTINE FOR GETTING RKCS, RKER, RKDS, RnDA
6123
6124      ;GT4RG
6125      ;SUBROUTINE FOR TRANSFERRING CONTENTS OF RKCS, RKER, RKDS
  
```


6126
6127
6128
6129
6130 020774 017737 160340 001170
6131 021002 017737 160320 001166
6132 021010 017737 160314 001164
6133 021016 017737 160310 001162
6134 021024 000207
6135
6136
6137
6138
6139
6140
6141
6142
6143
6144
6145
6146
6147
6148 021026
6149 021026 104401 021034
6150 021032 000406
6151
6152 021050
6153 021050 010346
6154 021052 104402
6155 021054 000207
6156
6157
6158
6159
6160
6161
6162
6163
6164
6165
6166
6167
6168
6169
6170
6171
6172
6173
6174
6175
6176
6177
6178
6179
6180 021056 010046
6181 021060 012700 001172

;RKDA TO \$REG0, \$REG1, \$REG2, \$REG3 RESPECTIVELY BEFORE
;TYPING OUT AN ERROR MESSAGE.
;CALL: JSR PC,GT4RG
GT4RG: MOV @RKDA,\$REG3 ;GET RKDA
GT3RG: MOV @RKDS,\$REG2 ;GET RKDS
GT2RG: MOV @RKER,\$REG1 ;GET RKER
MOV @RKCS,\$REG0
RTS PC

.SBTTL TYERM: SPECIAL ERROR MESSAGE ROUTINE

;TYERM
;THIS ROUTINE TYPES OUT 'EROR AT PC=X'
;X IS THE PC WHERE THE EXPLANATION AS TO WHAT HAPPENED IS GIVEN. THIS ROUTINE
;IS USED ONLY FOR NON-MANUAL MODE OF THE PROGRAM.
;CALL: JSR TYERM

TYERM:
TYPE ,65\$;;TYPE ASCIZ STRING
BR 64\$;;GET OVER THE ASCIZ
;;65\$: .ASCIZ <15><12>/EROR,PC=/
64\$:
MOV R3,-(SP)
TYPOC
RTS PC

.SBTTL BDAO, BDA4: BREAK DISK ADDRESS INTO SEC, SUR, CYL, DRIVE

;BDAO, BDA4
;THIS ROUTINE BREAKS A DISK ADDRESS (BITS 0-15) INTO DRIVE #,
;CYLINDER #, SURFACE, SECTOR #. THE ROUTINE IS CALLED BY USING EITHER
;BRKDAO OR BRKDA4, BOTH BEING 'TRAP' INSTRUCTIOS WITH THEIR LOWER BYTES
;ENCODED TO PROVIDE INDEXING TO 'BDAO' OR 'BDA4'. BEFORE CALLING
;THE ROUTINE THE DISK ADDRESS WHICH IS TO BE BROKEN AS ABOVE
;IS DEPOSITED IN \$REG10.
;'BRKDAO' PUTS THE
;DRIVE # INTO \$REG0
;CYLINDER # INTO \$REG1
;SURFACE # INTO \$REG2
;SECTOR # INTO \$REG3
;CALL: BRKDAO
BRKDA4 PUTS THE
DRIVE # INTO \$REG4
CYLINDER # INTO \$REG5
SURFACE # INTO \$REG6
SECTOR # INTO \$REG7
BRKDA4

BDAO: MOV R0,-(SP) ;PUSH R0 ONTO THE STACK
MOV # \$REG3+2,R0 ;SET UP POINTER

```

6182 021064 000403          BR      BDAR
6183
6184 021066 010046          BDA4:  MOV    R0,-(SP)      ;PUSH R0 ONTO THE STACK
6185 021070 012700 001202    MOV    #SREG7+2,R0      ;SET UP POINTER
6186
6187 021074 032777 020000 160036 BDAR:  BIT    #20000,@SWR      ;INHIBIT TYPEOUT?
6188 021102 001034          BNE    2$              ;YES, BRANCH TO EXIT POINT
6189
6190 021104 010146          MOV    R1,-(SP)      ;PUSH R1 ON STACK
6191 021106 010246          MOV    R2,-(SP)      ;PUSH R2 ON STACK
6192 021110 013701 001202    MOV    $REG10,R1     ;GET THE ADDRESS WHICH
6193                          ;HAS TO BE BROKEN
6194 021114 042701 177760    BIC    #177760,R1     ;EXTRACT SECTOR BITS 0-3
6195 021120 010140          MOV    R1,-(R0)      ;MOVE SECTOR BITS TO $REG3 OR $REG7
6196 021122 013701 001202    MOV    $REG10,R1     ;GET THE DSK-ADRES TO BE BROKEN
6197 021126 006201          ASR    R1              ;SHIFT RIGHT 4 TIMES
6198 021130 006201          ASR    R1
6199 021132 006201          ASR    R1
6200 021134 006201          ASR    R1
6201 021136 010102          MOV    R1,R2          ;STORE THIS
6202 021140 042702 177776    BIC    #177776,R2     ;EXTRACT THE SURFACE BIT
6203 021144 010240          MOV    R2,-(R0)      ;MOVE SURFACE BIT TO $REG3 OR $REG6
6204 021146 006201          ASR    R1
6205 021150 010102          MOV    R1,R2          ;STORE IT
6206 021152 042702 177400    BIC    #177400,R2     ;EXTRACT THE CYLINDER BITS
6207 021156 010240          MOV    R2,-(R0)      ;MOVE CYLINDER BITS TO $REG1 OR $REG5
6208 021160 000301          SWAB   R1              ;SWAB HI-LO BYTES
6209 021162 042701 177770    BIC    #177770,R1     ;EXTRACT THE DRIVE #
6210 021166 010140          MOV    R1,-(R0)      ;MOVE DRIVE # TO $REG0 OR $REG4
6211
6212 021170 012602          MOV    (SP)+,R2      ;RESTORE R2
6213 021172 012601          MOV    (SP)+,R1      ;RESTORE R1
6214 021174 012600          2$:   MOV    (SP)+,R0      ;RESTORE R0 FROM THE STACK
6215 021176 000002          RTI                    ;RETURN FROM INTERRUPT, EXIT THIS
6216                          ;ROUTINE
6217
6218
6219
6220                          .SBTTL  SHFTRT: SHIFT RIGHT ROUTINE
6221
6222                          ;SHFTRT
6223                          ;THIS ROUTINE SHIFTS A WORD TO THE RIGHT 13 TIMES. THE WORD TO BE SHIFTED
6224                          ;IS PUT ON THE STACK BEFORE ENTERING THIS ROUTINE AND IT IS POPPED UP
6225                          ;FROM THE STACK AFTER THE SHIFT HAS BEEN DONE.
6226                          ;CALL:  JSR    PC,SHFTRT
6227
6228 021200 012737 177763 021224 SHFTRT: MOV    #-15,2$      ;SET UP A COUNT OF 13
6229 021206 000241          CLC                    ;CLEAR THE C BIT
6230 021210 006066 000002    1$:   ROR    2(R6)      ;ROTATE RIGHT THE WORD TO B SHFTD
6231 021214 005237 021224    INC    2$              ;SHIFTED 13 TIMES?
6232 021220 001373          BNE    1$              ;IF NOT LUP BAK & SHIFT
6233 021222 000207          RTS    PC              ;EXIT FROM THIS SUBROUTINE
6234 021224 000000          2$:   0
6235
6236
6237
    
```

```

6238
6239
6240
6241
6242
6243
6244
6245
6246
6247
6248
6249
6250
6251
6252
6253
6254
6255
6256
6257
6258
6259
6260
6261 021226 010137 001202
6262 021232 000403
6263
6264 021234 013737 001350 001202
6265 021242 032777 140000 160062
6266 021250 001467
6267 021252 004737 020774
6268 021256 104416
6269
6270 021260 000207
6271
6272
6273
6274
6275
6276
6277
6278
6279
6280
6281
6282 021262 013705 001350
6283 021266 005205
6284 021270 020577 160044
6285 021274 001455
6286
6287 021276 010537 001202
6288 021302 104415
6289
6290 021304 017737 160030 001202
6291 021312 104416
6292
6293 021314 000207

        .SBTTL  CHKHE:  CHECK FOR 'ERR'OR
        .SBTTL  CHKHE1: CHECK FOR 'ERR'OR

::CHKHE
:THIS ROUTINE CHECKS IF 'HE' OR 'ERR' BITS IN RKCS ARE SET. IF ANY OF THE
:TWO BITS ARE SET, THE CONTENTS OF RKCS, ER, DS, AND DA ARE SAVED AND A
:RETURN IS MADE TO THE ERROR MESSAGE FOLLOWING THE 'JSR' CALL.
:AT THE TIME OF ENTRY 'DRIVAD' CONTAINS THE DISK ADDRESS WHICH IS TO
:BE BROKEN DOWN INTO DRIVE #, CYLINDER, SURFACE AND SECTOR #. THIS INFORMATION
:IS SAVED TO BE USED LATER FOR ERROR REPORTING. IF THE BITS ARE NOT SET,
:RETURN IS MADE TO SKIP THE ERROR MESSAGE.

:CHKHE1
:THIS ROUTINE CHECKS IF 'HE' OR 'ERR' BITS IN RKCS ARE SET. IF ANY OF THE
:TWO BITS ARE SET, THE CONTENTS OF RKCS, ER, DS, AND DA ARE SAVED AND A
:RETURN IS MADE TO THE ERROR MESSAGE FOLLOWING THE 'JSR' CALL.
:AT THE TIME OF ENTRY R1 CONTAINS THE DISK ADDRESS WHICH IS TO BE BROKEN
:DOWN INTO DRIVE #, CYLINDER, SURFACE AND SECTOR #. THIS INFORMATION IS
:SAVED TO BE USED LATER FOR ERROR REPORTING. IF THE BITS ARE NOT SET,
:RETURN IS MADE TO SKIP THE ERROR MESSAGE.

CHKHE1: MOV    R1,$REG10      ;SAVE THE DISK ADRES
        BR     CHE1

CHKHE:  MOV    DRIV/D,$REG10 ;SAVE THE DISK ADRES
CHE1:   BIT    #140000,@RKCS ;IS 'HE' OR 'ERR' BIT SET?
        BEQ    CRETRN        ;NO
        JSR    PC,GT4RG      ;GET RKCS,ER,DS, DA
        BRKDA4              ;GO TO 'BDA4' & BREAK CONTENTS O
                                ;$REG10 INTO DR#, CYL, SUR, SEC BITS
        RTS    PC           ;RETURN TO THE ERROR MESSAGE

        .SBTTL  CHKDA:  CHECK IF RKDA INCREMENTED CORRECTLY

:CHKDA
:THIS ROUTINE CHECKS IF RKDA INCREMENTED CORRECTLY. IF RKDA INCREMENTED
:CORRECTLY RETURN IS MADE TO SKIP THE ERROR MESSAGE.
:IF RKDA DID NOT INCREMENT CORRECTLY, THE EXPECTED AND RECIEVED VALUES
:OF RKDA ARE SAVED AND A RETURN IS MADE TO THE ERROR MESSAGE FOLLOWING THE
: 'JSR' CALL.
CHKDA:  MOV    DRIVAD,R5      ;RKDA SHOULD INCREMENT TO THIS
        INC    R5           ;AFTER DATA TRANSFER IS DONE
CHKDA1: CMP    R5,@RKDA      ;DID RKDA INCREMENT CORRECTLY?
        BEQ    CRETRN        ;IF YES, BRANCH
                                ;IF NOT, REPORT ERROR
        MOV    R5,$REG10     ;GET EXPCTD RKDA
        BRKDA0              ;GO TO 'BDA0' & BREAK CONTENTS OF
                                ;$REG10 INTO DR #,CYL,SUR,SEC BITS
        MOV    @RKDA,$REG10  ;GET ACTUAL RKDA
        BRKDA4              ;GO TO 'BDA4' & BREAK CONTENTS OF
                                ;$REG10 INTO DR #,CYL,SUR,SEC BITS
        RTS    PC           ;RETURN TO THE ERROR MESSAGE

```

6294
 6295
 6296
 6297
 6298
 6299
 6300
 6301
 6302 021316 005777 160012
 6303 021322 001442
 6304
 6305 021324 017737 160004 001162
 6306 021332 017737 160002 001164
 6307 021340 000207
 6308
 6309
 6310
 6311
 6312
 6313
 6314
 6315
 6316 021342 005777 157762
 6317 021346 001430
 6318
 6319 021350 004737 021002
 6320
 6321 021354 000207
 6322
 6323
 6324
 6325
 6326
 6327
 6328
 6329 021356 005777 157746
 6330 021362 001422
 6331 021364 013737 001330 001162
 6332 021372 017737 157732 001164
 6333 021400 000207
 6334
 6335
 6336
 6337
 6338
 6339
 6340 021402 022777 000200 157722
 6341 021410 001407
 6342 021412 013737 001332 001162
 6343 021420 017737 157706 001164
 6344 021426 000207
 6345
 6346 021430 062716 000002
 6347 021434 000207
 6348
 6349

```

.SBTTL CHKWC: CHECK IF RKWC OVERFLOWED

;CHKWC
;THIS ROUTINE CHECKS IF RKWC OVERFLOWED TO 0. IF IT DID A RETURN IS MADE
;TO SKIP THE ERROR MESSAGE. IF NOT, THE CONTENTS OF RKWC AND RKDA ARE SAVED
;AND A RETURN IS MADE TO THE ERROR MESSAGE FOLLOWING THE 'JSR' CALL.
CHKWC: TST @RKWC ;DID WORD COUNT OVERFLOW TO 0?
        BEQ CRETRN ;IF YES, BRANCH
        ;IF NOT, ERROR
        MOV @RKWC,$REG0 ;GET RKWC
        MOV @RKDA,$REG1 ;GET RKDA
        RTS PC ;RETURN TO THE ERROR MESSAGE

.SBTTL CHKER: CHECK RKER CONTENTS

;CHKER
;THIS ROUTINE CHECKS IF ANY BIT IN RKER SET. IF NOT RETURN IS MADE TO SKIP
;THE ERROR MESSAGE. IF ANY BIT IS SET THE CONTENTS OF RKCS, RKER, RKDS ARE
;SAVED AND A RETURN IS MADE TO THE ERROR MESSAGE.
CHKER: TST @RKER ;DID ANY BIT IN RKER SET?
        BEQ CRETRN ;NO, BRANCH
        ;YES, ERROR
        JSR PC,GT3RG ;GO, GET RKCS, ER, DS
        RTS PC ;RETURN TO THE ERROR MESSAGE

;CHKECLR
;THIS ROUTINE CHECKS THAT RKER IS CLEAR. IF NOT, THE CONTENTS OF RKER
;ARE SAVED AND A RETURN IS MADE TO THE ERROR MESSAGE FOLLOWING THE "JSR"
;CALL. IF RKER IS CLEAR THE ERROR MESSAGE IS SKIPPED ON RETURN.
CHKECLR: TST @RKER ;ANY BIT IN RKER SET?
        BEQ CRETRN ;NO
        MOV RKER,$REG0 ;GET ADRES OF RKER
        MOV @RKER,$REG1 ;GET CONTENTS OF RKER
        RTS PC ;RETURN TO THE ERROR MESSAGE

;CHKCCLR
;THIS ROUTINE CHECKS THAT RKCS IS CLEAR. IF NOT, THE CONTENTS OF RKCS ARE
;SAVED AND A RETURN IS MADE TO THE ERROR MESSAGE. IF RKCS IS CLEAR THE
;ERROR MESSAGE IS SKIPPED ON RETURN.
CHKCCLR: CMP #200,@RKCS ;IS RKCS CLEAR?
        BEQ CRETRN ;YES
        MOV RKCS,$REG0 ;SAVE ADRES OF RKCS
        MOV @RKCS,$REG1 ;SAVE THE CONTENT OF RKCS
        RTS PC ;RETURN TO THE ERROR MESSAGE

CRETRN: ADD #2,(SP) ;SKIP ERROR MESSAGE ON
        RTS PC ;RETURN
  
```

```

6350          .SBTTL TSTRWS: WAIT FOR R/W/S RDY ROUTINE
6351
6352          ;TSTRWS
6353          ;THIS ROUTINE WAITS FOR R/W/S RDY TO SET. WHEN IT SETS, THE RETURN PC
6354          ;IS INCREMENTED SO THAT ON RETURN (TO THE MAIN PROGRAM) THE ERROR
6355          ;MESSAGE FOLLOWING THE 'JSR' CALL IS SKIPPED. IF R/W/S RDY DOES NOT SET
6356          ;THEN A RETURN IS MADE TO THE ERROR MESSAGE (FOLLOWING THE 'JSR' CALL).
6357          ;WAITING TIME IS APPROX. 1040 MS FOR 11/20, APPROX. 208 MS FOR 11/45
6358          ;CALL: JSR TSTRWS
6359
6360 021436 013777 001350 157674 TSTRWS: MOV DRIVAD,@RKDA ;ADRES THE DRIVE
6361 021444 005037 001366          CLR TIMER ;INITIALIZE COUNT
6362 021450 C32777 000100 157650 1$: BIT #100,@RKDS ;DID R/W/S RDY SET?
6363 021456 001007          BNE 2$ ;YES, BRANCH
6364 021460 005237 001366          INC TIMER ;WAIT FOR R/W/S RDY
6365 021464 001371          BNE 1$ ;ERROR IF IT'S NOT SET BY NOW
6366 021466 017737 157634 001162 MOV @RKDS,$REGO ;GET RKDS
6367 021474 000207          RTS PC ;EXIT (TO ERROR FOOLOWING 'JSR TSTRWS')
6368
6369 021476 062716 000002          2$: ADD #2,(SP) ;ADJUST RETURN ADRES TO SKIP OVER
6370          ;ERROR (FOLLOWING 'JSR TSTRWS')
6371 021502 000207          RTS PC ;EXIT
6372
6373
6374
6375
6376
6377
6378
6379

```

.SBTTL DRESET: DRIVE RESET ROUTINE

```

6380          ;DRESET
6381          ;THIS ROUTINE DOES A DRIVE RESET ON THE DRIVE WHOOSE ADDRESS IS IN
6382          ;RKDA. MULTIPLE RETURN ADDRESSES FOR THIS ROUTINE ARE PROVIDED.
6383          ;IF THERE IS NO ERROR (R/W/S RDY SETS WITHIN CERTAIN TIME) , THEN BEFORE
6384          ;EXITNG FROM THIS ROUTINE THE RETURN ADDRESS IS INCREMENTED BY 2, TO SKIP
6385          ;THE ERROR MESSAGE ON RETURN. IF THERE IS AN ERROR, THE 3 REGISTERS (CS,ER,DS)
6386          ;ARE STORED AND THEN A NORMAL EXIT IS MADE FROM THIS ROUTINE TO THE
6387          ;ERROR MESSAGE FOLLOWING THE CALL FOR THIS ROUTINE.
6388          ;CALL: JSR PC,DRESET
6389
6390 021504 005037 001364          DRESET: CLR COUNT1 ;INITIALIZE THE COUNT
6391 021510 013777 001350 157622 MOV DRIVAD,@RKDA ;ADRES THE DRIVE
6392 021516 012777 000015 157606 MOV #15,@RKCS ;DRIVE RESET, GO
6393 021524 104414          CNT.RDY ;THIS IS A CALL FOR 'CN.RDY'
6394          ;ROUTINE WHICH WAITS FOR CNT
6395          ;RDY TO SET. IF CNTRL RDY DOES
6396          ;NOT SET WITHIN 883 MS/ 11-20
6397          ;(176 MS FOR 11-45 WITH BIPOLAR)
6398          ;AN ERROR IS REPORTED
6399 021526 032777 000100 157572 1$: BIT #100,@RKDS ;DID R/W/S RDY SET?
6400 021534 001013          BNE 2$
6401 021536 012746 177770          MOV #-10,-(SP) ;PUSH COUNT ON SP
6402 021542 005216          INC (SP) ;COUNT IT DOWN
6403 021544 001376          BNE .-2
6404 021546 005726          TST (SP)+ ;POP UP $P
6405 021550 005237 001364          INC COUNT1 ;IF NOT WAIT

```

6406 021554 001364
 6407 021556 004737 020774
 6408 021562 000402
 6409 021564 062716 000002
 6410 021570 000207

BNE 1\$;WAITED LONG?
 JSR PC,GT4RG
 BR 2\$+4
 2\$: ADD #2,@R6
 RTS PC

6411
 6412
 6413
 6414

.SBTTL TSTSIN: CHECK 'SIN' ROUTINE

6415
 6416
 6417

:TSTSIN
 :THIS ROUTINE CHECKS IF 'SIN' IS SET, IF IT IS SET A
 :DRIVE RESET IS DONE TO CLEAR 'SIN' AND INITIALIZE POSITIONER.
 :CALL: TST.SIN
 :IF ON DOING DRIVE RESET R/W/S RDY DOES NOT SET A MESSAGE
 : ERROR PC=XXXXXX IS GIVEN.
 :XXXXXX=PC IN THE MAIN PROGRAM WHERE 'TST.SIN' CALL IS LOCATED.

6418
 6419
 6420
 6421

6422
 6423
 6424

TSTSIN: MOV DRIVAD,@RKDA ;ADRES THE DRIVE
 BIT #1000,@RKDS ;IS SIN SET?
 BEQ 1\$
 JSR PC,DRESET ;GO DO DRIVE RESET, SIN SET
 BR 2\$;REPORT ERROR
 1\$: RTI
 2\$: BIT #SW13,@SWR ;INHIBIT TYPEOUT?
 BNE 1\$;IF YES, SKIP TYPEOUT
 TYPE ,65\$;;TYPE ASCIZ STRING
 BR 64\$;;GET OVER THE ASCIZ
 ;;65\$: .ASCIZ /ERROR PC= /
 64\$: MOV (SP),-(SP)
 ADD #-2,(SP) ;GET THE PC WHERE 'TST.SIN' IS LOCATED
 TYPOC ;GO TYPE OUT PC
 BR 1\$

6425 021572 013777 001350 157540
 6426 021600 032777 001000 157520
 6427 021606 001403
 6428 021610 004737 021504
 6429 021614 000401

6430 021616 000002
 6431 021620 032777 020000 157312
 6432 021626 001373
 6433 021630 104401 021636
 6434 021634 000406

6435
 6436 021652
 6437 021652 011646
 6438 021654 062716 177776
 6439 021660 104402
 6440 021662 000755

6441
 6442
 6443
 6444

.SBTTL DELAY: TIME DELAY ROUTINE

6445
 6446
 6447
 6448

:DELAY
 :THIS ROUTINE PROVIDES A VARIABLE TIME DELAY. THE CALL FOR THIS
 :ROUTINE IS AN ENCODED 'TRAP' INSTRUCTION.
 :CALL: DELAY ,N N IS ANY OCTAL NO. FROM 1 TO 177777
 :THE DELAY PROVIDED IS 7.5N US (CONVERT N TO DECIMAL) FOR 11/20
 :1.5N US FOR 11/45
 :IF THE USER WANTS TO CHANGE THE DELAY TIME (EXMP: SHORTER DELAY TO
 :GET A TIGHTER SCOPE LOOP) THE VARIABLE 'N' FOLLOWING 'DELAY' SHOULD
 :BE CHANGED TO SUIT THE INDIVIDUAL NEED.

6449
 6450
 6451
 6452
 6453
 6454
 6455

6456 021664 017637 000000 001366
 6457 021672 062716 000002
 6458
 6459 021676 005337 001366
 6460 021702 001375
 6461

DELA.Y: MOV @(SP),TIMER ;GET 'AMOUNT' (N) FOR WHICH
 ADD #2,(SP) ;DELAY IS TO BE PROVIDED
 ;ADJUST STACK POINTER TO SKIP OVER 'N'
 1\$: DEC TIMER ;COUNT DOWN TO 0
 BNE 1\$

```
6462 021704 000002          RTI          ;RETURN TO MAIN PROGRAM
6463
6464
6465
6466
6467          .SBTTL  WAT.INT:          WAIT FOR INTERRUPT ROUTINE
6468
6469          ;WAT.INT
6470          ;THIS ROUTINE PROVIDES A VARIABLE TIME WAIT LOOP DURING WHICH AN INTERRUPT
6471          ;FROM RK11 CAN OCCUR. THE CALL IS AN ENCODED 'TRAP' INSTRUCTION.
6472
6473          ;CALL:          WAT.INT ,N          N IS ANY OCTAL NO. FROM 1 TO 177777
6474
6475          ;WAIT LOOP TIME= APPROX. 7.5N US (CONVERT N TO DECIMAL) FOR 11/20
6476          ;APPROX. 1.5N US FOR 11/45
6477          ;UPON ENTERING THE ROUTINE THE CPU PRIORITY IS DROPPED SO THAT
6478          ;RK11 CAN INTERRUPT. NOTE THAT WHEN RK11 INTERRUPTS THIS ROUTINE
6479          ;IS EXITED WITHOUT POPPING THE STACK, THIS POPPING IS DONE AFTER GETTING
6480          ;TO RK11 INTERRUPT HANDLER.
6481          ;IF FOR ANY REASON THE WAIT LOOP TIME HAS TO BE CHANGED IT CAN BE DONE
6482          ;BY SIMPLY CHANGING THE VARIABLE 'N' FOLLOWING THE 'WAT.INT'.
6483
6484 021706 017637 000000 001366 WATINT: MOV      @ (SP),TIMER          ;GET 'AMOUNT' (N) FOR WHICH
6485 021714 062716 000002          ADD      #2,(SP)          ;WAITING IS TO BE DONE
6486          ;ADJUST STACK POINTER FOR CORRECT RETURN
6487 021720 013746 001400          MOV      RKPRI,-(SP)          ;DROP CPU PRIORITY SO THAT RK11 CAN
6488 021724 012746 021732          MOV      #18,-(SP)          ; INTERRUPT
6489 021730 000002          RTI
6490 021732 005337 001366 1$:      DEC      TIMER          ;WAIT FOR RK11 TO INTERRUPT
6491 021736 001375          BNE      1$
6492          ;IF INTERRUPT HAS NOT OCCURED BY NOW
6493          ;RETURN AND REPORT ERROR
6494 021740 000002          RTI          ;EXIT
6495
6496
6497
6498          ;WATIME
6499
6500 021742 005000          WATIME: CLR      R0
6501 021744 005001          CLR      R1
6502 021746 005200 1$:      INC      R0
6503 021750 001376          BNE      1$
6504 021752 105201          INCB    R1
6505 021754 001374          BNE      1$
6506 021756 000207          RTS      PC
6507
6508
6509          .SBTTL  CHKCRDY:          CHECK CONTROL READY
6510
6511          ;;CH.CRDY
6512          ;THIS ROUTINE WAITS FOR THE CONTROL READY TO SET. IF THE CONTROL READY BIT
6513          ;DOES NOT SET WITHIN A CERTAIN TIME, THEN THE CONTENTS OF RKCS, RKER, RKDS
6514          ;AND RKDA ARE SAVED AND AN EXIT MADE TO THE ERROR MESSAGE FOLLOWING THE
6515          ;'JSR' CALL FOR THIS ROUTINE.
6516          ;IF CONTROL READY SETS THEN THE RETURN ADDRESS IS ADJUSTED TO SKIP THE
6517          ;ERROR MESSAGE ON RETURN.
```

```
6518 ;CALL: CHKCRDY
6519 ;      ERROR                ;RETURN HERE IF ERROR
6520 ;      ---                  ;RETURN HERE IF NO ERROR
6521
6522 021760 005037 001366 CH.CRDY: CLR TIMER
6523 021764 105777 157342 1$: TSTB @RKCS ;CNTRL RDY SET?
6524 021770 100406 BMI 2$ ;YES
6525 021772 005237 001366 INC TIMER
6526 021776 001372 BNE 1$ ;NO, WAIT
6527 022000 004737 020774 JSR PC,GT4RG ;SAVE RKCS, ER, DS, DA
6528 022004 000002 RTI
6529
6530 022006 062716 000002 2$: ADD #2,(SP) ;ADJUST RETURN ADDRESS TO
6531 022012 000002 RTI ;SKIP ERROR MESSAGE ON RETURN
6532
6533
6534 .SBTTL CON.RESET: CONTROL REST ROUTINE
6535
6536 ;CON.RESET
6537 ;THIS ROUTINE ISSUES A CONTROL RESET AND WAITS FOR
6538 ;THE 'CNTRL RDY' FLAG TO SET. WHEN THE FLAG SETS
6539 ;AN EXIT IS MADE OUT OF THE ROUTINE. IF 'CNTRL-RDY'
6540 ;DOES NOT SET WITHIN A CERTAIN TIME AN ERROR MESSAGE
6541 ; CNT RDY DIDN'T SET
6542 ; PC=XXXXXX RKCS=YYYYYY
6543 ;IS GIVEN. NOTE THAT XXXXXX IS THE PC WHERE 'CNT.RESET' OR 'CNT.RDY'
6544 ;IS CALLED.
6545
6546 ;CALL: CNT.RESET
6547
6548
6549
6550
6551 .SBTTL CNT.RDY: WAIT FOR CONTROL READY ROUTINE
6552
6553 ;CN.RDY
6554 ;THIS ROUTINE WAITS FOR THE CONTROL READY BIT TO SET AND WHEN IT
6555 ;SETS EXITS OUT. IF WITHIN A CERTAIN TIME CNTRL RDY DOES
6556 ;NOT SET AN ERROR IS REPORTED. WAITING TIME IS 883 MS FOR 11/20
6557 ;175 MS FOR 11/45 WITH BIPOLAR MEMORY.
6558 ;CALL: CNT.RDY
6559 022014 012777 000001 ,57310 CN.RST: MOV #1,@RKCS ;ISSUE A CONTROL RESET
6560 022022 012737 177500 001170 MOV #-300,$REG3 ;SET UP COUNT
6561 022030 000402 BR CN.RDY+4 ;SKIP OVER CN.RDY
6562 022032 005037 001170 CN.RDY: CLR $REG3
6563 022036 105777 157270 1$: TSTB @RKCS ;DID CNTRL-RDY SET?
6564 022042 100435 BMI 3$ ;YES, EXIT
6565 022044 005237 001170 INC $REG3 ;WAITED LONG?
6566 022050 001372 BNE 1$ ;IF NOT, GO BAK & WAIT
6567 022052 032777 020000 157060 2$: BIT #SW13,@SWR ;INHIBIT TYPEOUT?
6568 022060 001026 BNE 3$ ;IF YES, SKIP TYPEOUT
6569 022062 104401 TYPE
6570 022064 001245 MSG3
6571 022066 104401 022074 TYPE ,65$ ;;TYPE ASCIZ STRING
6572 022072 000403 BR 64$ ;;GET OVER THE ASCIZ
6573 ;;65$: .ASCIZ <15><12>/PC=/
```



```

6574 022102
6575 022102 011646
6576 022104 162716 000002
6577 022110 104402
6578
6579 022112 104401 022120
6580 022116 000404
6581
6582 022130
6583 022130 017746 157176
6584 022134 104402
6585
6586 022136 000002
6587
6588
6589
6590
6591
6592
6593
6594
6595
6596
6597
6598
6599
6600
6601
6602
6603
6604
6605
6606
6607
6608
6609 022140
6610 022140 104407
6611 022142 032777 040000 156770
6612 022150 001111
6613
6614 022152 000416
6615
6616 022154 013746 000004
6617 022160 012737 022200 000004
6618 022166 005737 177060
6619 022172 012637 000004
6620 022176 000463
6621 022200 022626
6622 022202 012637 000004
6623 022206 000423
6624 022210
6625 022210 032777 000400 156722
6626 022216 001404
6627 022220 127737 156714 001102
6628 022226 001462
6629 022230 105737 001103

```

```

64$: MOV (SP),-(SP)
SUB #2,(SP)
TYPOC ;GO TYPE PC IN THE MAIN PROGRAM,
; WHERE ERROR OCCURRED
TYPE ,67$ ;:TYPE ASCIZ STRING
BR 66$ ;:GET OVER THE ASCIZ
;:67$: .ASCIZ / RKCS=/
66$: MOV @RKCS,-(SP) ;GET RKCS
TYPOC ;GO TYPE IT
3$: RTI ;RETURN FROM THIS
;ROUTINE TO THE MAIN
;PROGRAM

;THIS PART OF THE PROGRAM CONTAINS THE COMMON ROUTINES CALLED
;FROM THE SYSMAC.SML PACKAGE
;
.SBTTL SCOPE HANDLER ROUTINE
;:*****
;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
;*SW14=1 LOOP ON TEST
;*SW11=1 INHIBIT ITERATIONS
;*SW09=1 LOOP ON ERROR
;*SW08=1 LOOP ON TEST IN SWR<7:0>
;*CALL
;* SCOPE ;:SCOPE=IOT

$SCOPE:
1$: CKSWR ;:TEST FOR CHANGE IN SOFT-SWR
BIT #BIT14,@SWR ;:LOOP ON PRESENT TEST?
BNE $OVER ;:YES IF SW14=1
;####START OF CODE FOR THE XOR TESTER####
$XTSTR: BR 6$ ;:IF RUNNING ON THE 'XOR' TESTER CHANGE
;:THIS INSTRUCTION TO A 'NOP' (NOP=240)
MOV @#ERRVEC,-(SP) ;:SAVE THE CONTENTS OF THE ERROR VECTOR
MOV #5$,@#ERRVEC ;:SET FOR TIMEOUT
TST @#177060 ;:TIME OUT ON XOR?
MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
BR $$VLAD ;:GO TO THE NEXT TEST
5$: CMP (SP)+,(SP)+ ;:CLEAR THE STACK AFTER A TIME OUT
MOV (SP)+,@#ERRVEC ;:RESTORE THE ERROR VECTOR
BR 7$ ;:LOOP ON THE PRESENT TEST
6$;####END OF CODE FOR THE XOR TESTER####
BIT #BIT08,@SWR ;:LOOP ON SPEC. TEST?
BEQ 2$ ;:BR IF NO
CMPB @SWR,$STNM ;:ON THE RIGHT TEST? SWR<7:0>
BEQ $OVER ;:BR IF YES
2$: TSTB $ERFLG ;:HAS AN ERROR OCCURRED?

```

```

6630 022234 001421          BEQ      3$          ;;BR IF NO
6631 022236 123737 001115 001103  CMPB    $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
6632 022244 101015          BHI     3$          ;;BR IF NO
6633 022246 032777 001000 156664  BIT     #BIT09,@SWR   ;;LOOP ON ERROR?
6634 022254 001404          BEQ     4$          ;;BR IF NO
6635 022256 013737 001110 001106 7$:    MOV     $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
6636 022264 000443          BR      $OVER
6637 022266 105037 001103 4$:    CLRB   $ERFLG       ;;ZERO THE ERROR FLAG
6638 022272 005037 001206          CLR     $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
6639 022276 000415          BR      1$          ;;ESCAPE TO THE NEXT TEST
6640 022300 032777 004000 156632 3$:    BIT     #BIT11,@SWR  ;;INHIBIT ITERATIONS?
6641 022306 001011          BNE    1$          ;;BR IF YES
6642 022310 005737 001100          TST    $PASS       ;;IF FIRST PASS OF PROGRAM
6643 022314 001406          BEQ    1$          ;;      INHIBIT ITERATIONS
6644 022316 005237 001104          INC    $ICNT       ;;INCREMENT ITERATION COUNT
6645 022322 023737 001206 001104  CMP     $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
6646 022330 002021          BGE    $OVER       ;;BR IF MORE ITERATION REQUIRED
6647 022332 012737 000001 001104 1$:    MOV     #1,$ICNT   ;;REINITIALIZE THE ITERATION COUNTER
6648 022340 013737 022410 001206  MOV     $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
6649 022346 105237 001102          $SVLAD: INCB   $TSTNM   ;;COUNT TEST NUMBERS
6650 022352 011637 001106          MOV     (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
6651 022356 011637 001110          MOV     (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
6652 022362 005037 001210          CLR     $ESCAPE    ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
6653 022366 112737 000001 001115  MOVB   #1,$ERMAX   ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
6654 022374 013777 001102 156540 $OVER: MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
6655 022402 013716 001106          MOV     $LPADR,(SP) ;;FUDGE RETURN ADDRESS
6656 022406 000002          RTI
6657 022410 000050          $MXCNT: 50        ;;FIXES PS
6658
6659
6660
6661
6662
6663
6664
6665
6666
6667
6668
6669
6670
6671
6672 022412 104407          $ERROR: CKSWR      ;;CHECK FOR SOFTWARE SWITCH REGISTER REQUEST
6673 022414 105237 001103 7$:    INCB   $ERFLG     ;;SET THE ERROR FLAG
6674 022420 001775          BEQ     7$          ;;DON'T LET THE FLAG GO TO ZERO
6675 022422 013777 001102 156512  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
6676 022430 005237 001112 1$:    INC     $ERTTL     ;;COUNT THE NUMBER OF ERRORS
6677
6678 022434 032777 000100 156476  BIT     #BIT6,@SWR   ;;DESELECT DRIVE SW SET?
6679 022442 001404          BEQ     6$          ;;NO
6680 022444 023727 001112 000005  CMP     $ERTTL,#5   ;;MORE THAN 5 ERRORS ON THIS DRIVE?
6681 022452 101053          BHI     8$          ;;YES, DESELCT THE DRIVE
6682
6683 022454 011637 001116 6$:    MOV     (SP),$ERRPC ;;GET ADDRESS OF ERROR INSTRUCTION
6684 022460 162737 000002 001116  SUB     #2,$ERRPC
6685 022466 117737 156424 001114  MOVB   @ $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
    
```

.SBTTL ERROR HANDLER ROUTINE

```

;*SW15=1      HALT ON ERROR
;*SW13=1      INHIBIT ERROR TYPEOUTS
;*SW10=1      TESTING ON SIMULATOR
;*SW09=1      LOOP ON ERROR
;*SW12=1      CYCLE ON ERROR TO PREVIOUS 'SCOPE'
;*SW06=1      DROP DRIVE AFTER MAXIMUM (ALLOWABLE) ERRORS ON THE DRIVE
;*GO TO $ERRTYP ON ERROR
    
```

```

6686 022474 032777 020000 156436      BIT      #SW13,@SWR      ;SKIP TYPEOUT IF SET
6687 022502 001004                    BNE      2$          ;SKIP TYPEOUTS
6688 022504 004737 022734                    JSR      PC,@$ERRRYP ;GO TO USER ERROR ROUTINE
6689 022510 104401 001213                    TYPE     ,SCLRF
6690 022514 023737 000042 000046 2$:    CMP      @#42,@#46    ;ARE WE IN ACT11 AUTO MODE?
6691 022522 001403                    BEQ      .+10        ;YES, HALT ON ERROR
6692 022524 005777 156410                    TST      @SWR        ;HALT ON ERROR?
6693 022530 100002                    BPL      3$          ;SKIP IF CONTINUE
6694 022532 000000                    HALT     ;HALT ON ERROR!
6695 022534 104407                    CKSWR
6696 022536 032777 010000 156374 3$:    BIT      #SW12,@SWR  ;CHECK FOR SOFTWARE SWITCH REGIATER REQUEST
6697 022544 001402                    BEQ      .+6         ;SW 12 SET?
6698 022546 013716 001106                    MOV      $LPADR,(SP) ;NO, BRANCH
6699 022552 032777 001000 156360                    BIT      #SW09,@SWR ;ADJUST RETURN ADRES FOR SW12
6700 022560 001402                    BEQ      4$          ;LOOP ON ERROR SWITCH SET?
6701 022562 013716 001110                    MOV      $LPERR,(SP) ;BR IF NO
6702 022566 005737 001210 4$:    TST      $ESCAPE    ;FUDGE RETURN FOR LOOPING
6703 022572 001402                    BEQ      5$          ;CHECK FOR AN ESCAPE ADDRESS
6704 022574 013716 001210                    MOV      $ESCAPE,(SP) ;BR IF NONE
6705 022600 000002 5$:    RTI
6706
6707 022602 005737 001434 8$:    TST      T56FLG    ;IF EROR WAS IN LAST TEST (POLL)
6708
6709 022606 001407                    BEQ      10$         ;DROP ALL THE DRIVES
6710 022610 104401 001303                    TYPE     ,MSG5
6711 022614 005037 001412                    CLR      DRIVS
6712 022620 022626                    CMP      (SP)+,(SP)+
6713 022622 000137 020652                    JMP      $EOP
6714 022626 013746 001354 10$:    MOV      DRVPTR,-(SP) ;DROP THE DRIVE FROM THE
6715 022632 162716 000002                    SUB      #2,(SP)    ;SELECTION LIST
6716 022636 013746 001350                    MOV      DRIVAD,-(SP) ;DRIVE ADDR TO STACK
6717 022642 004737 021200                    JSR      PC,$HFTRT  ;RIGHT JUSTIFY
6718 022646 042716 000001                    BIC      #1,(R6)    ;MAKE EVEN
6719 022652 062716 001414                    ADD      #DRIV0,(SP) ;POINTS TO TABLE FOR EVEN DRIVE
6720 022656 042776 100000 000000                    BIC      #BIT15,@(R6) ;TEST REMAINING DRIVE AS RK05E
6721 022664 062716 000002                    ADD      #2,(R6)    ;POINT TO ODD
6722 022670 042736 100000                    BIC      #BIT15,@(SP)+ ;TEST AS RK-05E
6723 022674 012736 010000                    MOV      #BIT12,@(SP)+ ;INDICATE THIS DRIVE DROPPED
6724 022700 104401 001272                    TYPE     ,MSG4
6725 022704 013746 001350                    MOV      DRIVAD,-(R6) ;PUSH DRIVE # ON STACK
6726 022710 004737 021200                    JSR      PC,$HFTRT  ;SHIFT IT BEFORE TYPING
6727 022714 104407                    TYPOC
6728 022716 104401 001315                    TYPE     ,MSG6
6729 022722 005337 001412                    DEC      DRIVS      ;DECREMENT # OF DRIVES PRESNT
6730 022726 022626 9$:    CMP      (SP)+,(SP)+ ;RESTORE STACK
6731 022730 000137 020026                    JMP      BTEOP      ;GO BACK TO THE END OF PROGRM
6732
6733
6734
6735
6736
6737
6738
6739
6740
6741 022734
    
```

.SBTTL ERROR MESSAGE TYPEOUT ROUTINE

```

;*****
;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
    
```

\$ERRRYP:

6742	022734	104401	001213		TYPE	, \$CRLF	:: "CARRIAGE RETURN" & "LINE FEED"
6743	022740	010046			MOV	RO, -(SP)	:: SAVE RO
6744	022742	005000			CLR	RO	:: PICKUP THE ITEM INDEX
6745	022744	153700	001114		BISB	@#\$ITEMB, RO	
6746	022750	001004			BNE	1\$:: IF ITEM NUMBER IS ZERO, JUST
6747							:: TYPE THE PC OF THE ERROR
6748	022752	013746	001116		MOV	\$ERRPC, -(SP)	:: SAVE \$ERRPC FOR TYPEOUT
6749							:: ERROR ADDRESS
6750	022756	104402			TYPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS)
6751	022760	000426			BR	6\$:: GET OUT
6752	022762	005300		1\$:	DEC	RO	:: ADJUST THE INDEX SO THAT IT WILL
6753	022764	006300			ASL	RO	:: WORK FOR THE ERROR TABLE
6754	022766	006300			ASL	RO	
6755	022770	006300			ASL	RO	
6756	022772	062700	001442		ADD	#\$ERRTB, RO	:: FORM TABLE POINTER
6757	022776	012037	023006		MOV	(RO)+, 2\$:: PICKUP "ERROR MESSAGE" POINTER
6758	023002	001404			BEQ	3\$:: SKIP TYPEOUT IF NO POINTER
6759	023004	104401			TYPE		:: TYPE THE "ERROR MESSAGE"
6760	023006	000000		2\$:	.WORD	0	:: "ERROR MESSAGE" POINTER GOES HERE
6761	023010	104401	001213		TYPE	, \$CRLF	:: "CARRIAGE RETURN" & "LINE FEED"
6762	023014	012037	023024	3\$:	MOV	(RO)+, 4\$:: PICKUP "DATA HEADER" POINTER
6763	023020	001404			BEQ	5\$:: SKIP TYPEOUT IF 0
6764	023022	104401			TYPE		:: TYPE THE "DATA HEADER"
6765	023024	000000		4\$:	.WORD	0	:: "DATA HEADER" POINTER GOES HERE
6766	023026	104401	001213		TYPE	, \$CRLF	:: "CARRIAGE RETURN" & "LINE FEED"
6767	023032	011000		5\$:	MOV	(RO), RO	:: PICKUP "DATA TABLE" POINTER
6768	023034	001004			BNE	7\$:: GO TYPE THE DATA
6769	023036	012600		6\$:	MOV	(SP)+, RO	:: RESTORE RO
6770	023040	104401	001213		TYPE	, \$CRLF	:: "CARRIAGE RETURN" & "LINE FEED"
6771	023044	000207			RTS	PC	:: RETURN
6772	023046			7\$:			
6773	023046	013046			MOV	@(RO)+, -(SP)	:: SAVE @(RO)+ FOR TYPEOUT
6774	023050	104402			TYPOC		:: GO TYPE--OCTAL ASCII(ALL DIGITS)
6775	023052	005710			TST	(RO)	:: IS THERE ANOTHER NUMBER?
6776	023054	001770			BEQ	6\$:: BR IF NO
6777	023056	104401	023064		TYPE	, 8\$:: TYPE TWO(2) SPACES
6778	023062	000771			BR	7\$:: LOOP
6779	023064	020040	000	8\$:	.ASCIZ	/ /	:: TWO(2) SPACES
6780		023070			.EVEN		

.SBTTL TYPE ROUTINE

```

*****
*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
*
*CALL:
*1) USING A TRAP INSTRUCTION
*   TYPE      ,MESADR      ;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
*OR
*   TYPE
*   MESADR
*

```

```

6798
6799 023070 105737 001157 $TYPE: TSTB $TPFLG ;; IS THERE A TERMINAL?
6800 023074 100002 BPL 1$ ;; BR IF YES
6801 023076 000000 HALT ;; HALT HERE IF NO TERMINAL
6802 023100 000407 BR 3$ ;; LEAVE
6803 023102 010046 1$: MOV R0,-(SP) ;; SAVE R0
6804 023104 017600 000002 MOV @2(SP),R0 ;; GET ADDRESS OF ASCIZ STRING
6805 023110 112046 2$: MOVB (R0)+,-(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
6806 023112 001005 BNE 4$ ;; BR IF IT ISN'T THE TERMINATOR
6807 023114 005726 TST (SP)+ ;; IF TERMINATOR POP IT OFF THE STACK
6808 023116 012600 60$: MOV (SP)+,R0 ;; RESTORE R0
6809 023120 062716 3$: ADD #2,(SP) ;; ADJUST RETURN PC
6810 023124 000002 RTI ;; RETURN
6811 023126 122716 000011 4$: CMPB #HT,(SP) ;; BRANCH IF <HT>
6812 023132 001430 BEQ 8$
6813 023134 122716 000200 CMPB #CRLF,(SP) ;; BRANCH IF NOT <CRLF>
6814 023140 001006 BNE 5$
6815 023142 005726 TST (SP)+ ;; POP <CR><LF> EQUIV
6816 023144 104401 TYPE ;; TYPE A CR AND LF
6817 023146 001213 $CRLF
6818 023150 105037 023304 CLRB $CHARCNT ;; CLEAR CHARACTER COUNT
6819 023154 000755 BR 2$ ;; GET NEXT CHARACTER
6820 023156 004737 023240 5$: JSR PC,$TYPEC ;; GO TYPE THIS CHARACTER
6821 023162 123726 001156 6$: CMPB $FILLC,(SP)+ ;; IS IT TIME FOR FILLER CHARS.?
6822 023166 001350 BNE 2$ ;; IF NO GO GET NEXT CHAR.
6823 023170 013746 001154 MOV $NULL,-(SP) ;; GET # OF FILLER CHARS. NEEDED
6824 ;; AND THE NULL CHAR.
6825 023174 105366 000001 7$: DECB 1(SP) ;; DOES A NULL NEED TO BE TYPED?
6826 023200 002770 BLT 6$ ;; BR IF NO--GO POP THE NULL OFF OF STACK
6827 023202 004737 023240 JSR PC,$TYPEC ;; GO TYPE A NULL
6828 023206 105337 023304 DECB $CHARCNT ;; DO NOT COUNT AS A COUNT
6829 023212 000770 BR 7$ ;; LOOP
6830
6831 ;HORIZONTAL TAB PROCESSOR
6832
6833 023214 112716 000040 8$: MOVB #' ,(SP) ;; REPLACE TAB WITH SPACE
6834 023220 004737 023240 9$: JSR PC,$TYPEC ;; TYPE A SPACE
6835 023224 132737 000007 023304 BITB #7,$CHARCNT ;; BRANCH IF NOT AT
6836 023232 001372 BNE 9$ ;; TAB STOP
6837 023234 005726 TST (SP)+ ;; POP SPACE OFF STACK
6838 023236 000724 BR 2$ ;; GET NEXT CHARACTER
6839 023240 105777 155704 $TYPEC: TSTB @STPS ;; WAIT UNTIL PRINTER IS READY
6840 023244 100375 BPL $TYPEC
6841 023246 116677 000002 155676 MOVB 2(SP),@STPB ;; LOAD CHAR TO BE TYPED INTO DATA REG.
6842 023254 122766 000015 000002 CMPB #CR,2(SP) ;; IS CHARACTER A CARRIAGE RETURN?
6843 023262 001003 BNE 1$ ;; BRANCH IF NO
6844 023264 105037 023304 CLRB $CHARCNT ;; YES--CLEAR CHARACTER COUNT
6845 023270 000406 BR $TYPEX ;; EXIT
6846 023272 122766 000012 000002 1$: CMPB #LF,2(SP) ;; IS CHARACTER A LINE FEED?
6847 023300 001402 BEQ $TYPEX ;; BRANCH IF YES
6848 023302 105227 INCB (PC)+ ;; COUNT THE CHARACTER
6849 023304 000000 $CHARCNT: .WORD 0 ;; CHARACTER COUNT STORAGE
6850 023306 000207 $TYPEX: RTS PC
6851
6852
6853
    
```

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

6854
6855
6856
6857
6858
6859
6860
6861
6862
6863
6864
6865 023310
6866 023310 010046
6867 023312 010146
6868 023314 010246
6869 023316 010346
6870 023320 010546
6871 023322 012746 020200
6872 023326 016605 000020
6873 023332 100004
6874 023334 005405
6875 023336 112766 000055 000001
6876 023344 005000
6877 023346 012703 023524
6878 023352 112723 000040
6879 023356 005002
6880 023360 016001 023514
6881 023364 160105
6882 023366 002402
6883 023370 005202
6884 023372 000774
6885 023374 060105
6886 023376 005702
6887 023400 001002
6888 023402 105716
6889 023404 100407
6890 023406 106316
6891 023410 103003
6892 023412 116663 000001 177777
6893 023420 052702 000060
6894 023424 052702 000040
6895 023430 110223
6896 023432 005720
6897 023434 020027 000010
6898 023440 002746
6899 023442 003002
6900 023444 010502
6901 023446 000764
6902 023450 105726
6903 023452 100003
6904 023454 116663 177777 177776
6905 023462 105013
6906 023464 012605
6907 023466 012603
6908 023470 012602
6909 023472 012601

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

$TYPDS:
MOV      R0,-(SP)      ;;PUSH R0 ON STACK
MOV      R1,-(SP)      ;;PUSH R1 ON STACK
MOV      R2,-(SP)      ;;PUSH R2 ON STACK
MOV      R3,-(SP)      ;;PUSH R3 ON STACK
MOV      R5,-(SP)      ;;PUSH R5 ON STACK
MOV      #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
MOV      20(SP),R5      ;;GET THE INPUT NUMBER
BPL      1$            ;;BR IF INPUT IS POS.
NEG      R5            ;;MAKE THE BINARY NUMBER POS.
MOVB     #'-,1(SP)      ;;MAKE THE ASCII NUMBER NEG.
1$:      CLR      R0      ;;ZERO THE CONSTANTS INDEX
MOV      #SDBLK,R3      ;;SETUP THE OUTPUT POINTER
MOVB     #' ,(R3)+      ;;SET THE FIRST CHARACTER TO A BLANK
2$:      CLR      R2      ;;CLEAR THE BCD NUMBER
MOV      $DTBL(R0),R1    ;;GET THE CONSTANT
3$:      SUB      R1,R5      ;;FORM THIS BCD DIGIT
BLT      4$            ;;BR IF DONE
INC      R2            ;;INCREASE THE BCD DIGIT BY 1
BR       3$
4$:      ADD      R1,R5      ;;ADD BACK THE CONSTANT
TST      R2            ;;CHECK IF BCD DIGIT=0
BNE      5$            ;;FALL THROUGH IF 0
TSTB     (SP)          ;;STILL DOING LEADING 0'S?
BMI      7$            ;;BR IF YES
5$:      ASLB     (SP)          ;;MSD?
BCC      6$            ;;BR IF NO
MOVB     1(SP),-1(R3)    ;;YES--SET THE SIGN
6$:      BIS      #'0,R2      ;;MAKE THE BCD DIGIT ASCII
7$:      BIS      #' ,R2      ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB     R2,(R3)+      ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST      (R0)+          ;;JUST INCREMENTING
CMP      R0,#10        ;;CHECK THE TABLE INDEX
BLT      2$            ;;GO DO THE NEXT DIGIT
BGT      8$            ;;GO TO EXIT
MOV      R5,R2          ;;GET THE LSD
BR       6$            ;;GO CHANGE TO ASCII
8$:      TSTB     (SP)+      ;;WAS THE LSD THE FIRST NON-ZERO?
BPL      9$            ;;BR IF NO
MOVB     -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
9$:      CLRB     (R3)          ;;SET THE TERMINATOR
MOV      (SP)+,R5      ;;POP STACK INTO R5
MOV      (SP)+,R3      ;;POP STACK INTO R3
MOV      (SP)+,R2      ;;POP STACK INTO R2
MOV      (SP)+,R1      ;;POP STACK INTO R1

```

6910	023474	012600		
6911	023476	104401	023524	
6912	023502	016666	000002	000004
6913	023510	012616		
6914	023512	000002		
6915	023514	023420		
6916	023516	001750		
6917	023520	000144		
6918	023522	000012		
6919	023524	000004		
6920				
6921				
6922				
6923				
6924				
6925				
6926				
6927				
6928				
6929				
6930				
6931				
6932				
6933				
6934				
6935				
6936				
6937				
6938				
6939				
6940				
6941				
6942				
6943				
6944				
6945				
6946	023534	017646	000000	
6947	023540	116637	000001	023757
6948	023546	112637	023761	
6949	023552	062716	000002	
6950	023556	000406		
6951	023560	112737	000001	023757
6952	023566	112737	000006	023761
6953	023574	112737	000005	023756
6954	023602	010346		
6955	023604	010446		
6956	023606	010546		
6957	023610	113704	023761	
6958	023614	005404		
6959	023616	062704	000006	
6960	023622	110437	023760	
6961	023626	113704	023757	
6962	023632	016605	000012	
6963	023636	005003		
6964	023640	006105		
6965	023642	000404		

```

MOV (SP)+,R0      ;;POP STACK INTO R0
TYPE $DBLK        ;;NOW TYPE THE NUMBER
MOV 2(SP),4(SP)  ;;ADJUST THE STACK
MOV (SP)+,(SP)
RTI              ;;RETURN TO USER

$DTBL: 10000.
        1000.
        100.
        10.

$DBLK: .BLKW 4

.SBTTL BINARY TO OCTAL (ASCII) AND TYPE

*****
*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
*OCTAL (ASCII) NUMBER AND TYPE IT.
*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOS    ;;CALL FOR TYPEOUT
*      .BYTE   N              ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
*      .BYTE   M              ;;M=1 OR 0
*                               ;;1=TYPE LEADING ZEROS
*                               ;;0=SUPPRESS LEADING ZEROS

*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
*$TYPOS OR $TYPOC
*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPON    ;;CALL FOR TYPEOUT

*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
*CALL:
*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
*      TYPOC    ;;CALL FOR TYPEOUT

$TYPOS: MOV      @ (SP),-(SP)    ;;PICKUP THE MODE
        MOV      1(SP),%OFILL   ;;LOAD ZERO FILL SWITCH
        MOV      (SP)+,%SOMODE+1 ;;NUMBER OF DIGITS TO TYPE
        ADD      #2,(SP)        ;;ADJUST RETURN ADDRESS
        BR       $TYPON

$TYPOC: MOV      #1,%OFILL      ;;SET THE ZERO FILL SWITCH
        MOV      #6,%SOMODE+1   ;;SET FOR SIX(6) DIGITS
$TYPON: MOV      #5,%OCNT       ;;SET THE ITERATION COUNT
        MOV      R3,-(SP)       ;;SAVE R3
        MOV      R4,-(SP)       ;;SAVE R4
        MOV      R5,-(SP)       ;;SAVE R5
        MOV      %SOMODE+1,R4   ;;GET THE NUMBER OF DIGITS TO TYPE
        NEG      R4
        ADD      #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
        MOV      R4,%SOMODE     ;;SAVE IT FOR USE
        MOV      %OFILL,R4      ;;GET THE ZERO FILL SWITCH
        MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
        CLR      R3            ;;CLEAR THE OUTPUT WORD
        ROL     R5             ;;ROTATE MSB INTO 'C'
        BR      3$            ;;GO DO MSB
  
```

```
6966 023644 006105 2$: ROL R5 ;;FORM THIS DIGIT
6967 023646 006105 ROL R5
6968 023650 006105 ROL R5
6969 023652 010503 MOV R5,R3
6970 023654 006103 3$: ROL R3 ;;GET LSB OF THIS DIGIT
6971 023656 105337 023760 DECB $OMODE ;;TYPE THIS DIGIT?
6972 023662 100016 BPL 7$ ;;BR IF NO
6973 023664 042703 177770 BIC #177770,R3 ;;GET RID OF JUNK
6974 023670 001002 BNE 4$ ;;TEST FOR 0
6975 023672 005704 TST R4 ;;SUPPRESS THIS 0?
6976 023674 001403 BEQ 5$ ;;BR IF YES
6977 023676 005204 4$: INC R4 ;;DON'T SUPPRESS ANYMORE 0'S
6978 023700 052703 000060 BIS #'0,R3 ;;MAKE THIS DIGIT ASCII
6979 023704 052703 000040 5$: BIS #' ,R3 ;;MAKE ASCII IF NOT ALREADY
6980 023710 110337 023754 MOVB R3,8$ ;;SAVE FOR TYPING
6981 023714 104401 023754 TYPE ,8$ ;;GO TYPE THIS DIGIT
6982 023720 105337 023756 7$: DECB $OCNT ;;COUNT BY 1
6983 023724 003347 BGT 2$ ;;BR IF MORE TO DO
6984 023726 002402 BLT 6$ ;;BR IF DONE
6985 023730 005204 INC R4 ;;INSURE LAST DIGIT ISN'T A BLANK
6986 023732 000744 BR 2$ ;;GO DO THE LAST DIGIT
6987 023734 012605 6$: MOV (SP)+,R5 ;;RESTORE R5
6988 023736 012604 MOV (SP)+,R4 ;;RESTORE R4
6989 023740 012603 MOV (SP)+,R3 ;;RESTORE R3
6990 023742 016666 000002 000004 MOV 2(SP),4(SP) ;;SET THE STACK FOR RETURNING
6991 023750 012616 MOV (SP)+,(SP)
6992 023752 000002 RTI ;;RETURN
6993 023754 000 8$: .BYTE 0 ;;STORAGE FOR ASCII DIGIT
6994 023755 000 .BYTE 0 ;;TERMINATOR FOR TYPE ROUTINE
6995 023756 000 $OCNT: .BYTE 0 ;;OCTAL DIGIT COUNTER
6996 023757 000 $OFILL: .BYTE 0 ;;ZERO FILL SWITCH
6997 023760 000000 $OMODE: .WORD 0 ;;NUMBER OF DIGITS TO TYPE
6998
6999 .SBTTL TTY INPUT ROUTINE
7000
7001 ;;*****
7002 .ENABL LSB
7003
7004 ;;*****
7005 ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
7006 ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
7007 ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
7008 ;*WHEN OPERATING IN TTY FLAG MODE.
7009 023762 022737 000176 001140 $CKSWR: CMP #SWREG,SWR ;;IS THE SOFT-SWR SELECTED?
7010 023770 001074 BNE 15$ ;;BRANCH IF NO
7011 023772 105777 155146 TSTB @TKS ;;CHAR THERE?
7012 023776 100071 BPL 15$ ;;IF NO, DON'T WAIT AROUND
7013 024000 117746 155142 MOVB @TKB,-(SP) ;;SAVE THE CHAR
7014 024004 042716 177600 BIC #C177,(SP) ;;STRIP-OFF THE ASCII
7015 024010 022726 000007 CMP #7,(SP)+ ;;IS IT A CONTROL G?
7016 024014 001062 BNE 15$ ;;NO, RETURN TO USER
7017 024016 123727 001134 000001 CMPB $AUTOB,#1 ;;ARE WE RUNNING IN AUTO-MODE?
7018 024024 001456 BEQ 15$ ;;BRANCH IF YES
7019
7020 024026 104401 024647 $GTSWR: TYPE ,SCNTLG ;;ECHO THE CONTROL-G ( G)
7021 024032 104401 024654 TYPE ,MSWR ;;TYPE CURRENT CONTENTS
```



```

7022 024036 013746 000176      MOV      SWREG,-(SP)      ;;SAVE SWREG FOR TYPEOUT
7023 024042 104402              TYP0C      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7024 024044 104401 024665      TYPE      ,SMNEW      ;;PROMPT FOR NEW SWR
7025 024050 005046              19$: CLR      -(SP)      ;;CLEAR COUNTER
7026 024052 005046              CLR      -(SP)      ;;THE NEW SWR
7027 024054 105777 155064      7$: TSTB   @STKS      ;;CHAR THERE?
7028 024060 100375              BPL      7$          ;;IF NOT TRY AGAIN
7029
7030 024062 117746 155060      MOVB   @STKB,-(SP)      ;;PICK UP CHAR
7031 024066 042716 177600      BIC    # C177,(SP)     ;;MAKE IT 7-BIT ASCII
7032
7033
7034
7035 024072 021627 000025      9$:  CMP    (SP),#25    ;;IS IT A CONTROL-U?
7036 024076 001005              BNE     10$          ;;BRANCH IF NOT
7037 024100 104401 024642      TYPE   ,SCNTLU      ;;YES, ECHO CONTROL-U ( U)
7038 024104 062706 000006      20$: ADD   #6,SP      ;;IGNORE PREVIOUS INPUT
7039 024110 000757              BR     19$          ;;LET'S TRY IT AGAIN
7040
7041
7042 024112 021627 000015      10$: CMP   (SP),#15    ;;IS IT A <CR>?
7043 024116 001022              BNE     16$          ;;BRANCH IF NO
7044 024120 005766 000004      TST    4(SP)        ;;YES, IS IT THE FIRST CHAR?
7045 024124 001403              BEQ     11$          ;;BRANCH IF YES
7046 024126 016677 000002 155004  MOV    2(SP),@SWR     ;;SAVE NEW SWR
7047 024134 062706 000006      11$: ADD   #6,SP      ;;CLEAR UP STACK
7048 024140 104401 001213      14$: * TYPE   ,$CRLF   ;;ECHO <CR> AND <LF>
7049 024144 123727 001135 000001  CMPB   $INTAG,#1     ;;RE-ENABLE TTY KBD INTERRUPTS?
7050 024152 001003              BNE     15$          ;;BRANCH IF NOT
7051 024154 012777 000100 154762  MOV    #100,@STKS    ;;RE-ENABLE TTY KBD INTERRUPTS
7052 024162 000002              15$: RTI                    ;;RETURN
7053 024164 004737 023240      16$: JSR    PC,$TYPEC    ;;ECHO CHAR
7054 024170 021627 000060      CMP    (SP),#60     ;;CHAR < 0?
7055 024174 002420              BLT     18$          ;;BRANCH IF YES
7056 024176 021627 000067      CMP    (SP),#67     ;;CHAR > 7?
7057 024202 003015              BGT     18$          ;;BRANCH IF YES
7058 024204 042726 000060      BIC    #60,(SP)+    ;;STRIP-OFF ASCII
7059 024210 005766 000002      TST    2(SP)        ;;IS THIS THE FIRST CHAR
7060 024214 001403              BEQ     17$          ;;BRANCH IF YES
7061 024216 006316              ASL    (SP)         ;;NO, SHIFT PRESENT
7062 024220 006316              ASL    (SP)         ;; CHAR OVER TO MAKE
7063 024222 006316              ASL    (SP)         ;; ROOM FOR NEW ONE.
7064 024224 005266 000002      17$: INC   2(SP)        ;;KEEP COUNT OF CHAR
7065 024230 056616 177776      BIS    -2(SP),(SP)  ;;SET IN NEW CHAR
7066 024234 000707              BR     7$          ;;GET THE NEXT ONE
7067 024236 104401 001212      18$: TYPE   ,SQUES    ;;TYPE ?<CR><LF>
7068 024242 000720              BR     20$          ;;SIMULATE CONTROL-U
7069 .DSABL  LSB
7070
7071
7072 *****
7073 ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
7074 ;*CALL:
7075 ;*      RDCHR          ;;INPUT A SINGLE CHARACTER FROM THE TTY
7076 ;*      RETURN HERE   ;;CHARACTER IS ON THE STACK
7077 ;*                    ;;WITH PARITY BIT STRIPPED OFF

```

```

7078 ;
7079 ;
7080 024244 011646 $RDCHR: MOV (SP),-(SP) ;;PUSH DOWN THE PC
7081 024246 016666 000004 000002 MOV 4(SP),2(SP) ;;SAVE THE PS
7082 024254 105777 154664 1$: TSTB @STKS ;;WAIT FOR
7083 024260 100375 BPL 1$ ;;A CHARACTER
7084 024262 117766 154660 000004 MOVB @STKB,4(SP) ;;READ THE TTY
7085 024270 042766 177600 000004 BIC #(<177>,4(SP) ;;GET RID OF JUNK IF ANY
7086 024276 026627 000004 000023 CMP 4(SP),#23 ;;IS IT A CONTROL-S?
7087 024304 001013 BNE 3$ ;;BRANCH IF NO
7088 024306 105777 154632 2$: TSTB @STKS ;;WAIT FOR A CHARACTER
7089 024312 100375 BPL 2$ ;;LOOP UNTIL ITS THERE
7090 024314 117746 154626 MOVB @STKB,-(SP) ;;GET CHARACTER
7091 024320 042716 177600 BIC #(<177>,(SP) ;;MAKE IT 7-BIT ASCII
7092 024324 022627 000021 CMP (SP)+,#21 ;;IS IT A CONTROL-Q?
7093 024330 001366 BNE 2$ ;;IF NOT DISCARD IT
7094 024332 000750 BR 1$ ;;YES, RESUME
7095 024334 026627 000004 000140 3$: CMP 4(SP),#140 ;;IS IT UPPER CASE?
7096 024342 002407 BLT 4$ ;;BRANCH IF YES
7097 024344 026627 000004 000175 CMP 4(SP),#175 ;;IS IT A SPECIAL CHAR?
7098 024352 003003 BGT 4$ ;;BRANCH IF YES
7099 024354 042766 000040 000004 BIC #40,4(SP) ;;MAKE IT UPPER CASE
7100 024362 000002 4$: RTI ;;GO BACK TO USER
7101 ;*****
7102 ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
7103 ;*CALL:
7104 ;* RDLIN ;;INPUT A STRING FROM THE TTY
7105 ;* RETURN HERE ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
7106 ;* ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
7107
7108 024364 010346 $RDLIN: MOV R3,-(SP) ;;SAVE R3
7109 024366 005046 CLR -(SP) ;;CLEAR THE RUBOUT KEY
7110 024370 012703 024620 1$: MOV #STTYIN,R3 ;;GET ADDRESS
7111 024374 022703 024642 2$: CMP #STTYIN+22,R3 ;;BUFFER FULL?
7112 024400 101456 BLOS 4$ ;;BR IF YES
7113 024402 104410 RDCHR ;;GO READ ONE CHARACTER FROM THE TTY
7114 024404 112613 MOVB (SP)+,(R3) ;;GET CHARACTER
7115 024406 122713 000177 10$: CMPB #177,(R3) ;;IS IT A RUBOUT
7116 024412 001022 BNE 5$ ;;BR IF NO
7117 024414 005716 TST (SP) ;;IS THIS THE FIRST RUBOUT?
7118 024416 001007 BNE 6$ ;;BR IF NO
7119 024420 112737 000134 024616 MOVB #' ,9$ ;;TYPE A BACK SLASH
7120 024426 104401 024616 TYPE ,9$
7121 024432 012716 177777 MOV #-1,(SP) ;;SET THE RUBOUT KEY
7122 024436 005303 6$: DEC R3 ;;BACKUP BY ONE
7123 024440 020327 024620 CMP R3,#STTYIN ;;STACK EMPTY?
7124 024444 103434 BLO 4$ ;;BR IF YES
7125 024446 111337 024616 MOVB (R3),9$ ;;SETUP TO TYPEOUT THE DELETED CHAR.
7126 024452 104401 024616 TYPE ,9$ ;;GO TYPE
7127 024456 000746 BR 2$ ;;GO READ ANOTHER CHAR.
7128 024460 005716 5$: TST (SP) ;;RUBOUT KEY SET?
7129 024462 001406 BEQ 7$ ;;BR IF NO
7130 024464 112737 000134 024616 MOVB #' ,9$ ;;TYPE A BACK SLASH
7131 024472 104401 024616 TYPE ,9$
7132 024476 005016 CLR (SP) ;;CLEAR THE RUBOUT KEY
7133 024500 122713 000025 7$: CMPB #25,(R3) ;;IS CHARACTER A CTRL U?

```

```

7134 024504 001003          BNE      8$          ;;BR IF NO
7135 024506 104401 024642   TYPE     ,%CNTLU    ;;TYPE A CONTROL "U"
7136 024512 000726          BR       1$          ;;GO START OVER
7137 024514 122713 000022   8$:     CMPB     #22,(R3) ;;IS CHARACTER A "R"?
7138 024520 001011          BNE      3$          ;;BRANCH IF NO
7139 024522 105013          CLR      (R3)        ;;CLEAR THE CHARACTER
7140 024524 104401 001213   TYPE     ,%CRLF     ;;TYPE A "CR" & "LF"
7141 024530 104401 024620   TYPE     ,%TTYIN    ;;TYPE THE INPUT STRING
7142 024534 000717          BR       2$          ;;GO PICKUP ANOTHER CHACTER
7143 024536 104401 001212   4$:     TYPE     ,%QUES ;;TYPE A '?'
7144 024542 000712          BR       1$          ;;CLEAR THE BUFFER AND LOOP
7145 024544 111337 024616   3$:     MOV      (R3),9$ ;;ECHO THE CHARACTER
7146 024550 104401 024616   TYPE     ,9$
7147 024554 122723 000015   CMPB     #15,(R3)+  ;;CHECK FOR RETURN
7148 024560 001305          BNE      2$          ;;LOOP IF NOT RETURN
7149 024562 105063 177777   CLR      -1(R3)     ;;CLEAR RETURN (THE 15)
7150 024566 104401 001214   TYPE     ,%LF       ;;TYPE A LINE FEED
7151 024572 005726          TST     (SP)+        ;;CLEAN RUBOUT KEY FROM THE STACK
7152 024574 012603          MOV     (SP)+,R3    ;;RESTORE R3
7153 024576 011646          MOV     (SP),-(SP)  ;;ADJUST THE STACK AND PUT ADDRESS OF THE
7154 024600 016666 000004 000002   MOV     4(SP),2(SP) ;; FIRST ASCII CHARACTER ON IT
7155 024606 012766 024620 000004   MOV     #%TTYIN,4(SP)
7156 024614 000002          RTI              ;;RETURN
7157 024616 000          9$:     .BYTE    0          ;;STORAGE FOR ASCII CHAR. TO TYPE
7158 024617 000          .BYTE    0          ;;TERMINATOR
7159 024620 000022          $TTYIN: .BLKB    22      ;;RESERVE 22 BYTES FOR TTY INPUT
7160 024642 052536 005015 000   $CNTLU: .ASCIZ  / U/<15><12> ;;CONTROL "U"
7161 024647 136 006507 000012   $CNTLG: .ASCIZ  / G/<15><12> ;;CONTROL "G"
7162 024654 005015 053523 020122   $MSWR:  .ASCIZ  <15><12>/SWR = /
7163 024662 020075 000
7164 024665 040 047040 053505   $MNEW:  .ASCIZ  / NEW = /
7165 024672 036440 000040
7166                                     ;;CONTROL U, RUBOUT CAPABILITY
7167   .SBTTL  TRAP DECODER
7168
7169   ;;*****
7170   ;;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
7171   ;;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
7172   ;;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
7173   ;;*GO TO THAT ROUTINE.
7174
7175   $TRAP: MOV     R0,-(SP)      ;;SAVE R0
7176   MOV     2(SP),R0          ;;GET TRAP ADDRESS
7177   TST     -(R0)             ;;BACKUP BY 2
7178   MOV     (R0),R0           ;;GET RIGHT BYTE OF TRAP
7179   ASL     R0                ;;POSITION FOR INDEXING
7180   MOV     $TRPAD(R0),R0     ;;INDEX TO TABLE
7181   RTS     R0                ;;GO TO ROUTINE
7182
7183
7184   ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
7185
7186   $TRAP2: MOV     (SP),-(SP)  ;;MOVE THE PC DOWN
7187   MOV     4(SP),2(SP)       ;;MOVE THE PSW DOWN
7188   RTI
7189

```

7190
7191
7192
7193
7194
7195
7196
7197 024732 024720
7198 024734 023070
7199 024736 023560
7200 024740 023534
7201 024742 023574
7202 024744 023310
7203
7204 024746 024032
7205
7206 024750 023762
7207 024752 024244
7208 024754 024364
7209
7210 024756 021760
7211
7212 024760 022014
7213
7214 024762 022032
7215
7216 024764 021056
7217
7218 024766 021066
7219
7220 024770 021664
7221
7222 024772 021706
7223
7224 024774 021572
7225
7226
7227
7228
7229
7230
7231 024776 012737 025142 000024
7232 025004 012737 000340 000026
7233 025012 010046
7234 025014 010146
7235 025016 010246
7236 025020 010346
7237 025022 010446
7238 025024 010546
7239 025026 017746 154106
7240 025032 010637 025146
7241 025036 012737 025050 000024
7242 025044 000000
7243 025046 000776
7244
7245

.SBTTL TRAP TABLE

;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
;*BY THE "TRAP" INSTRUCTION.

:	ROUTINE		
:	-----		
\$TRPAD:	.WORD	\$TRAP2	
	\$TYPE	::CALL=TYPE	TRAP+1(104401) TTY TYPEOUT ROUTINE
	\$TYPOC	::CALL=TYPOC	TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
	\$TYPOS	::CALL=TYPOS	TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
	\$TYPON	::CALL=TYPON	TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
	\$TYPDS	::CALL=TYPDS	TRAP+5(104405) TYPE DECIMAL NUMBER (WITH SIGN)
	\$GTSWR	::CALL=GTSWR	TRAP+6(104406) GET SOFT-SWR SETTING
	\$CKSWR	::CALL=CKSWR	TRAP+7(104407) TEST FOR CHANGE IN SOFT-SWR
	\$RDCHR	::CALL=RDCHR	TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
	\$RDLIN	::CALL=RDLIN	TRAP+11(104411) TTY TYPEIN STRING ROUTINE
	CH.CRDY	::CALL=CHKCRDY	TRAP+12(104412) CHECK CONTROL READY
	CN.RST	::CALL=CNT.RESET	TRAP+13(104413) CONTROL RESET ROUTINE
	CN.RDY	::CALL=CNT.RDY	TRAP+14(104414) WAIT FOR CNTRL RDY TO SET
	BDA0	::CALL=BRKDA0	TRAP+15(104415) BREAK RKDA INTO DR #,CYL,SUR,SEC BITS
	BDA4	::CALL=BRKDA4	TRAP+16(104416) BREAK RKDA INTO DR #,CYL,SUR,SEC BITS
	DELA.Y	::CALL=DELAY	TRAP+17(104417) TIME DELAY ROUTINE
	WATINT	::CALL=WAT.INT	TRAP+20(104420) WAIT FOR RK11 INTERRUPT ROUTINE
	TSTSIN	::CALL=TST.SIN	TRAP+21(104421) TEST SIN ROUTINE

.SBTTL POWER DOWN AND UP ROUTINES

```

;*****
;POWER DOWN ROUTINE
$PWRDN: MOV    #SILLUP,@#PWRVEC ;;SET FOR FAST UP
        MOV    #340,@#PWRVEC+2 ;;PRIO:7
        MOV    R0,-(SP)      ;;PUSH R0 ON STACK
        MOV    R1,-(SP)      ;;PUSH R1 ON STACK
        MOV    R2,-(SP)      ;;PUSH R2 ON STACK
        MOV    R3,-(SP)      ;;PUSH R3 ON STACK
        MOV    R4,-(SP)      ;;PUSH R4 ON STACK
        MOV    R5,-(SP)      ;;PUSH R5 ON STACK
        MOV    @SWR,-(SP)     ;;PUSH @SWR ON STACK
        MOV    SP,$SAVR6     ;;SAVE SP
        MOV    #SPWRUP,@#PWRVEC ;;SET UP VECTOR
        HALT
        BR     .-2          ;;HANG UP
;*****

```

```
7246      :POWER UP ROUTINE
7247 025050 012737 025142 000024 $PWRUP: MOV    #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
7248 025056 013706 025146          MOV    $$SAVR6,SP      ;;GET SP
7249 025062 005037 025146          CLR    $$SAVR6        ;;WAIT LOOP FOR THE TTY
7250 025066 005237 025146 1$: INC    $$SAVR6        ;;WAIT FOR THE INC
7251 025072 001375          BNE    1$            ;;OF WORD
7252 025074 012677 154040          MOV    (SP)+,@SWR     ;;POP STACK INTO @SWR
7253 025100 012605          MOV    (SP)+,R5      ;;POP STACK INTO R5
7254 025102 012604          MOV    (SP)+,R4      ;;POP STACK INTO R4
7255 025104 012603          MOV    (SP)+,R3      ;;POP STACK INTO R3
7256 025106 012602          MOV    (SP)+,R2      ;;POP STACK INTO R2
7257 025110 012601          MOV    (SP)+,R1      ;;POP STACK INTO R1
7258 025112 012600          MOV    (SP)+,R0      ;;POP STACK INTO R0
7259 025114 012737 024776 000024 MOV    #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
7260 025122 012737 000340 000026 MOV    #340,@#PWRVEC+2 ;;PRIO:7
7261 025130 104401          TYPE                                ;REPORT THE POWER FAILURE
7262 025132 025150 $PWRMG: .WORD $POWER      ;;POWER FAIL MESSAGE POINTER
7263 025134 012716          MOV    (PC)+,(SP)    ;;RESTART AT PFSTRT
7264 025136 004702 $PWRAD: .WORD PFSTRT     ;;RESTART ADDRESS
7265 025140 000002          RTI
7266 025142 000000 $ILLUP: HALT            ;;THE POWER UP SEQUENCE WAS STARTED
7267 025144 000776          BR    .-2           ;; BEFORE THE POWER DOWN WAS COMPLETE
7268 025146 000000          $$SAVR6: 0         ;;PUT THE SP HERE
7269 025150 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER'
7270 025156 000122          .EVEN
7271
7272
7273 025160 004737 021504 FCHECK: JSR    PC,DRESET      ;RESETB DRIVE
7274 025164 104026          ERROR    26
7275 025166 104413          CNT.RESET
7276 025170 013737 001350 025302 MOV    DRIVAD,DRHOLD     ;SAVE DRIVE ADRR
7277 025176 032737 020000 001350 BIT    #20000,DRIVAD    ;SEE IF ODD
7278 025204 001404          BEQ    1$
7279 025206 042737 020000 001350 BIC    #20000,DRIVAD    ;MAKE EVEN
7280 025214 000403          BR    2$
7281 025216 052737 020000 001350 1$: BIS    #20000,DRIVAD    ;MAKE ODD
7282 025224 013777 001350 154106 2$: MOV    DRIVAD,@RKDA    ;DRIVE ADDR
7283 025232 012777 000011 154072 MOV    #11,@RKCS       ;DRIVE SEEK
7284 025240 104414          CNT.RDY
7285 025242 013777 025302 154070 MOV    DRHOLD,@RKDA    ;OTHER DRIVE
7286 025250 104414          CNT.RDY
7287 025252 032777 000100 154046 BIT    #100,@RKDS      ;HEAEDS IN MOTIONN?
7288 025260 001001          BNE    3$            ;NO SO RK-05J
7289 025262 005725          TST    (R5)+         ;YES RK-05F
7290 025264 013737 025302 001350 3$: MOV    DRHOLD,DRIVAD    ;RESTORE ADDR
7291 025272 004737 021504          JSR    PC,DRESET     ;WAIT FOR RESET
7292 025276 104026          ERROR    26
7293 025300 000205          RTS    R5
7294 025302 000000          DRHOLD: 0
7295 025304 005037 001350 SIZEF: CLR    DRIVAD     ;START AT DRO
7296 025310 012700 001414          MOV    #DRIVO,R0     ;TABLE OF AVAIL DRIVES
7297 025314 005710          4$: TST    (R0)       ;THIS DRIVE HERE?
7298 025316 001413          BEQ    2$            ;NO
7299 025320 005760 000002          TST    2(R0)        ;COMPLEMENT HERE?
7300 025324 001410          BEQ    2$            ;NO
7301 025326 004537 025160          JSR    R5,FCHECK     ;SEE IF F MODEL
```


7358	025623	123	041505	041455	EM36:	.ASCIZ /SEC-CNTR DIDN'T INCRMNT/
7359	025630	052116	020122	044504		
7360	025636	047104	052047	044440		
7361	025644	041516	046522	052116		
7362	025652	000				
7363						
7364	025653	123	041505	041455	EM37:	.ASCIZ /SEC-COUNTR INCRMENTED WRONG/
7365	025660	052517	052116	020122		
7366	025666	047111	051103	042515		
7367	025674	052116	042105	053440		
7368	025702	047522	043516	000		
7369						
7370	025707	104	042111	023516	EM40:	.ASCIZ /DIDN'T GET SC=SA FOR THIS SECTR/
7371	025714	020124	042507	020124		
7372	025722	041523	051475	020101		
7373	025730	047506	020122	044124		
7374	025736	051511	051440	041505		
7375	025744	051124	000			
7376						
7377	025747	105	047522	026522	EM41:	.ASCIZ "EROR-R/W/S RDY SHOULD BE SET"
7378	025754	027522	027527	020123		
7379	025762	042122	020131	044123		
7380	025770	052517	042114	041040		
7381	025776	020105	042523	000124		
7382						
7383	026004	047125	054105	042520	EM43:	.ASCIZ /UNEXPECTED RK11 INTERRUPT/
7384	026012	052103	042105	051040		
7385	026020	030513	020061	047111		
7386	026026	042524	051122	050125		
7387	026034	000124				
7388						
7389	026036	047103	051124	020114	EM44:	.ASCIZ /CNTRL RDY DIDN'T SET AFTER SEEK OR DR RESET/
7390	026044	042122	020131	044504		
7391	026052	047104	052047	051440		
7392	026060	052105	040440	052106		
7393	026066	051105	051440	042505		
7394	026074	020113	051117	042040		
7395	026102	020122	042522	042523		
7396	026110	000124				
7397						
7398	026112	051105	020122	051117	EM45:	.ASCIZ /FRR OR HE BIT SET ON SEEK OR DR RESET/
7399	026120	044040	020105	044502		
7400	026126	020124	042523	020124		
7401	026134	047117	051440	042505		
7402	026142	020113	051117	042040		
7403	026150	020122	042522	042523		
7404	026156	000124				
7405						
7406	026160	045522	051105	041040	EM46:	.ASCIZ /RKER BIT, ON SEEK OR DR RESET/
7407	026166	052111	020054	047117		
7408	026174	051440	042505	020113		
7409	026202	051117	042040	020122		
7410	026210	042522	042523	000124		
7411						
7412	026216	045522	051503	041440	EM47:	.ASCIZ /RKCS CHNGD AFTR FUNCTION WAS DONE/
7413	026224	047110	042107	040440		

7414	026232	052106	020122	052506		
7415	026240	041516	044524	047117		
7416	026246	053440	051501	042040		
7417	026254	047117	000105			
7418						
7419	026260	027522	027527	020123	EM50:	.ASCIZ 'R/W/S RDY DIDN'T CLEAR''
7420	026266	042122	020131	044504		
7421	026274	047104	052047	041440		
7422	026302	042514	051101	000		
7423						
7424	026307	122	053457	051457	EM51:	.ASCIZ 'R/W/S RDY DIDN'T SET AFTR SEEK OR DR RESET''
7425	026314	051040	054504	042040		
7426	026322	042111	023516	020124		
7427	026330	042523	020124	043101		
7428	026336	051124	051440	042505		
7429	026344	020113	051117	042040		
7430	026352	020122	042522	042523		
7431	026360	000124				
7432						
7433	026362	045522	040504	041440	EM52:	.ASCIZ /RKDA CHNGD AFTR SEEK/
7434	026370	047110	042107	040440		
7435	026376	052106	020122	042523		
7436	026404	045505	000			
7437						
7438	026407	103	052116	046122	EM53:	.ASCIZ /CNTRL RDY DIDN'T CLR AS GO WAS SET/
7439	026414	051040	054504	042040		
7440	026422	042111	023516	02124		
7441	026430	046103	020122	051501		
7442	026436	043440	020117	040527		
7443	026444	020123	042523	000124		
7444						
7445	026452	047103	051124	020114	EM54:	.ASCIZ ''CNTRL RDY DIDN'T SET ON WRT/FMT STARTING FROM <DSK-ADRES>''
7446	026460	042122	020131	044504		
7447	026466	047104	052047	051440		
7448	026474	052105	047440	020116		
7449	026502	051127	027524	046506		
7450	026510	020124	052123	051101		
7451	026516	044524	043516	043040		
7452	026524	047522	020115	042074		
7453	026532	045523	040455	051104		
7454	026540	051505	000076			
7455						
7456	026544	042510	047440	020122	EM55:	.ASCIZ ''HE OR ERR ON WRT/FMT STARTING FROM <DSK-ADRES>''
7457	026552	051105	020122	047117		
7458	026560	053440	052122	043057		
7459	026566	052115	051440	040524		
7460	026574	052122	047111	020107		
7461	026602	051106	046517	036040		
7462	026610	051504	026513	042101		
7463	026616	042522	037123	000		
7464						
7465	026623	122	042113	020101	EM56:	.ASCIZ /RKDA INCRMNTD WRONG ON WRT-FMT/
7466	026630	047111	051103	047115		
7467	026636	042124	053440	047522		
7468	026644	043516	047440	020116		
7469	026652	051127	026524	046506		

7470	026660	000124			
7471					
7472	026662	045522	041527	042040	EM57: .ASCIZ /RKWC DIDN'T OVRFLO ON WRT FMT/
7473	026670	042111	023516	020124	
7474	026676	053117	043122	047514	
7475	026704	047440	020116	051127	
7476	026712	020124	046506	000124	
7477					
7478	026720	045522	040502	044440	EM60: .ASCIZ /RKBA INCRMNTD WRONG ON WRT FMT/
7479	026726	041516	046522	052116	
7480	026734	020104	051127	047117	
7481	026742	020107	047117	053440	
7482	026750	052122	043040	052115	
7483	026756	000			
7484					
7485	026757	122	042513	020122	EM61: .ASCIZ /RKER SET,ON WRT OR RD OR FMT/
7486	026764	042523	026124	047117	
7487	026772	053440	052122	047440	
7488	027000	020122	042122	047440	
7489	027006	020122	046506	000124	
7490					
7491	027014	045522	041104	042440	EM62: .ASCIZ /RKDB EROR/
7492	027022	047522	000122		
7493					

7494	027026	045522	040504	044440	EM63:	.ASCIZ /RKDA INCRMNTD WRONG ON RD OR RD FMT/
7495	027034	041516	046522	052116		
7496	027042	020104	051127	047117		
7497	027050	020107	047117	051040		
7498	027056	020104	051117	051040		
7499	027064	020104	046506	000124		
7500						
7501	027072	045522	041527	042040	EM64:	.ASCIZ /RKWC DIDN'T OVRFLO ON RD OR RD FMT/
7502	027100	042111	023516	020124		
7503	027106	053117	043122	047514		
7504	027114	047440	020116	042122		
7505	027122	047440	020122	042122		
7506	027130	043040	052115	000		
7507						
7508	027135	122	041113	020101	EM65:	.ASCIZ /RKBA INCRMNTD WRONG ON RD OR RD FMT/
7509	027142	047111	051103	047115		
7510	027150	042124	053440	047522		
7511	027156	043516	047440	020116		
7512	027164	042122	047440	020122		
7513	027172	042122	043040	052115		
7514	027200	000				
7515						
7516	027201	111	041516	051117	EM66:	.ASCIZ /INCORRECT HEADER FROM 'SECTOR'/
7517	027206	042522	052103	044040		
7518	027214	040505	042504	020122		
7519	027222	051106	046517	023440		
7520	027230	042523	052103	051117		
7521	027236	000047				
7522						
7523	027240	040504	040524	042440	EM67:	.ASCIZ /DATA ERROR/
7524	027246	051122	051117	000		
7525						
7526	027253	103	052116	046122	EM70:	.ASCIZ ''CNTRL RDY DIDN'T SET ON RD/FMT STARTING FROM <DSK-ADRES>''
7527	027260	051040	054504	042040		
7528	027266	042111	023516	020124		
7529	027274	042523	020124	047117		
7530	027302	051040	027504	046506		
7531	027310	020124	052123	051101		
7532	027316	044524	043516	043040		
7533	027324	047522	020115	042074		
7534	027332	045523	040455	051104		
7535	027340	051505	000076			
7536						
7537	027344	042510	047440	020122	EM71:	.ASCIZ ''HE OR ERR ON RD/FMT STARTING FROM <DSK-ADRES>''
7538	027352	051105	020122	047117		
7539	027360	051040	027504	046506		
7540	027366	020124	052123	051101		
7541	027374	044524	043516	043040		
7542	027402	047522	020115	042074		
7543	027410	045523	040455	051104		
7544	027416	051505	000076			
7545						
7546	027422	051127	047117	020107	EM72:	.ASCIZ /WRONG DRIVE ID IN RKDS AFTER SEEK/
7547	027430	051104	053111	020105		
7548	027436	042111	044440	020116		
7549	027444	045522	051504	040440		

7550	027452	052106	051105	051440	
7551	027460	042505	000113		
7552					
7553	027464	051110	053504	042522	EM73: .ASCIZ /HRDWRE POLL-DRV ID BITS(13-15) SHLD BE CLR/
7554	027472	050040	046117	026514	
7555	027500	051104	020126	042111	
7556	027506	041040	052111	024123	
7557	027514	031461	030455	024465	
7558	027522	051440	046110	041104	
7559	027530	020105	046103	000122	
7560					
7561	027536	051110	053504	042522	EM74: .ASCIZ /HRDWRE POLL-INTRUPTING DRIV # NOT PRSNT/
7562	027544	050040	046117	026514	
7563	027552	047111	051124	050125	
7564	027560	044524	043516	042040	
7565	027566	044522	020126	020043	
7566	027574	047516	020124	051120	
7567	027602	047123	000124		
7568					
7569	027606	051104	053111	021440	EM75: .ASCIZ /DRIV # DIDN'T INTRUPT AFTER HRDWRE POLL/
7570	027614	042040	042111	023516	
7571	027622	020124	047111	051124	
7572	027630	050125	020124	043101	
7573	027636	042524	020122	051110	
7574	027644	053504	042522	050040	
7575	027652	046117	000114		
7576					
7577	027656	041523	020120	044504	EM76: .ASCIZ /SCP DIDN'T SET AFTER SEEK WAS DONE/
7578	027664	047104	052047	051440	
7579	027672	052105	040440	052106	
7580	027700	051105	051440	042505	
7581	027706	020113	040527	020123	
7582	027714	047504	042516	000	
7583					
7584	027721	122	042113	020101	EM77: .ASCIZ /RKDA CHANGD AFTER DRIV RESET/
7585	027726	044103	047101	042107	
7586	027734	040440	052106	051105	
7587	027742	042040	044522	020126	
7588	027750	042522	042523	000124	
7589					
7590	027756	040504	040524	042440	EM100: .ASCIZ /DATA EROR AT WORD#/
7591	027764	047522	020122	052101	
7592	027772	053440	051117	021504	
7593	030000	000			
7594					
7595	030001	103	052116	046122	EM101: .ASCIZ /CNTRL RDY DIDN'T SET AFTER RD CHK/
7596	030006	051040	054504	042040	
7597	030014	042111	023516	020124	
7598	030022	042523	020124	043101	
7599	030030	042524	020122	042122	
7600	030036	041440	045510	000	
7601					
7602	030043	105	051122	047440	EM102: .ASCIZ /ERR OR HE ON RD CHK/
7603	030050	020122	042510	047440	
7604	030056	020116	042122	041440	
7605	030064	045510	000		

7606					
7607	030067	103	042523	047440	EM103: .ASCIZ /CSE ON RD CHK/
7608	030074	020116	042122	041440	
7609	030102	045510	000		
7610					
7611	030105	122	053513	020103	EM104: .ASCIZ /RKWC DIDN'T OVERFLO ON RD CHK OR WRT CHK/
7612	030112	044504	047104	052047	
7613	030120	047440	042526	043122	
7614	030126	047514	047440	020116	
7615	030134	042122	041440	045510	
7616	030142	047440	020122	051127	
7617	030150	020124	044103	000113	
7618					
7619	030156	045522	040504	044440	EM105: .ASCIZ /RKDA INCRMNTD WRONG ON RD CHK/
7620	030164	041516	046522	052116	
7621	030172	020104	051127	047117	
7622	030200	020107	047117	051040	
7623	030206	020104	044103	000113	
7624					
7625	030214	045522	040502	041440	EM106: .ASCIZ /RKBA CHANGD AFTER RD CHK/
7626	030222	040510	043516	020104	
7627	030230	043101	042524	020122	
7628	030236	042122	041440	045510	
7629	030244	000			
7630					
7631	030245	115	046505	051117	EM107: .ASCIZ /MEMORY WORD CHANGED AFTER RD CHK/
7632	030252	020131	047527	042122	
7633	030260	041440	040510	043516	
7634	030266	042105	040440	052106	
7635	030274	051105	051040	020104	
7636	030302	044103	000113		
7637					
7638	030306	047103	051124	020114	EM110: .ASCIZ /CNTRL RDY DIDN'T SET AFTER WRT CHK/
7639	030314	042122	020131	044504	
7640	030322	047104	052047	051440	
7641	030330	052105	040440	052106	
7642	030336	051105	053440	052122	
7643	030344	041440	045510	000	
7644					
7645	030351	110	020105	051117	EM111: .ASCIZ /HE OR ERR ON WRT CHK/
7646	030356	042440	051122	047440	
7647	030364	020116	051127	020124	
7648	030372	044103	000113		
7649					
7650	030376	051127	052111	020105	EM112: .ASCIZ /WRITE CHECK EROR/
7651	030404	044103	041505	020113	
7652	030412	051105	051117	000	
7653					
7654	030417	122	042113	020101	EM113: .ASCIZ /RKDA INCRMNTD WRONG ON WRT CHK/
7655	030424	047111	051103	047115	
7656	030432	042124	053440	047522	
7657	030440	043516	047440	020116	
7658	030446	051127	020124	044103	
7659	030454	000113			
7660					
7661	030456	045522	040502	044440	EM114: .ASCIZ /RKBA INCRMNTD WRONG ON WRT CHK/

7662	030464	041516	046522	052116	
7663	030472	020104	051127	047117	
7664	030500	020107	047117	053440	
7665	030506	052122	041440	045510	
7666	030514	000			
7667					
7668	030515	122	041113	020101	EM115: .ASCIZ /RKBA INCRMNTD, WITH IBA SET/
7669	030522	047111	051103	047115	
7670	030530	042124	020054	044527	
7671	030536	044124	044440	040502	
7672	030544	051440	052105	000	
7673					
7674	030551	127	047522	043516	EM116: .ASCIZ /WRONG MEMORY LOCATION CHANGED WITH IBA SET/
7675	030556	046440	046505	051117	
7676	030564	020131	047514	040503	
7677	030572	044524	047117	041440	
7678	030600	040510	043516	042105	
7679	030606	053440	052111	020110	
7680	030614	041111	020101	042523	
7681	030622	000124			
7682					
7683	030624	045522	030461	042040	EM117: .ASCIZ /RK11 DIDN'T INTRUPT WHEN IDE WAS SET/
7684	030632	042111	023516	020124	
7685	030640	047111	051124	050125	
7686	030646	020124	044127	047105	
7687	030654	044440	042504	053440	
7688	030662	051501	051440	052105	
7689	030670	000			
7690					
7691	030671	122	030513	020061	EM120: .ASCIZ /RK11 DIDN'T INTRUPT AFTER SK WAS INITIATED/
7692	030676	044504	047104	052047	
7693	030704	044440	052116	052522	
7694	030712	052120	040440	052106	
7695	030720	051105	051440	020113	
7696	030726	040527	020123	047111	
7697	030734	052111	040511	042524	
7698	030742	000104			
7699					
7700	030744	041523	020120	042523	EM121: .ASCIZ /SCP SET BEFORE SEEK COMPLETED/
7701	030752	020124	042502	047506	
7702	030760	042522	051440	042505	
7703	030766	020113	047503	050115	
7704	030774	042514	042524	000104	
7705					
7706	031002	045522	030461	042040	EM122: .ASCIZ /RK11 DIDN'T INTRUPT AFTER SK COMPLETED/
7707	031010	042111	023516	020124	
7708	031016	047111	051124	050125	
7709	031024	020124	043101	042524	
7710	031032	020122	045523	041440	
7711	031040	046517	046120	052105	
7712	031046	042105	000		
7713					
7714	031051	103	052116	046122	EM123: .ASCIZ /CNTRL RESET DIDN'T CLEAR 'SCP'/
7715	031056	051040	051505	052105	
7716	031064	042040	042111	023516	
7717	031072	020124	046103	040505	

7718	031100	020122	051447	050103	
7719	031106	000047			
7720					
7721	031110	045522	030461	042040	EM124: .ASCIZ /RK11 DIDN'T INTRUPT AFTER RD DONE/
7722	031116	042111	023516	020124	
7723	031124	047111	051124	050125	
7724	031132	020124	043101	042524	
7725	031140	020122	042122	042040	
7726	031146	047117	000105		
7727					
7728	031152	047103	051124	020114	EM125: .ASCIZ /CNTRL RESET DIDN'T CLR REGISTR/
7729	031160	042522	042523	020124	
7730	031166	044504	047104	052047	
7731	031174	041440	051114	051040	
7732	031202	043505	051511	051124	
7733	031210	000			
7734					
7735	031211	122	030513	020061	EM126: .ASCIZ /RK11 DIDN'T INTRUPT AT CPU LEVEL/
7736	031216	044504	047104	052047	
7737	031224	044440	052116	052522	
7738	031232	052120	040440	020124	
7739	031240	050103	020125	042514	
7740	031246	042526	000114		
7741					
7742	031252	045522	030461	044440	EM127: .ASCIZ /RK11 INTRUPTED AT WRONG CPU LEVEL/
7743	031260	052116	052522	052120	
7744	031266	042105	040440	020124	
7745	031274	051127	047117	020107	
7746	031302	050103	020125	042514	
7747	031310	042526	000114		
7748					
7749	031314	042447	051122	041040	EM130: .ASCIZ /'ERR BIT' DIDN'T SET IN RKER/
7750	031322	052111	020047	044504	
7751	031330	047104	052047	051440	
7752	031336	052105	044440	020116	
7753	031344	045522	051105	000	
7754					
7755	031351	110	020105	051117	EM131: .ASCIZ /HE OR ERR DIDN'T SET/
7756	031356	042440	051122	042040	
7757	031364	042111	023516	020124	
7758	031372	042523	000124		
7759					
7760	031376	045522	051105	042440	EM132: .ASCIZ /RKER EROR/
7761	031404	047522	000122		
7762					
7763	031410	054116	020103	044502	EM133: .ASCIZ /NXC BIT DIDN'T SET/
7764	031416	020124	044504	047104	
7765	031424	052047	051440	052105	
7766	031432	000			
7767					
7768	031433	122	030513	020061	EM134: .ASCIZ /RK11 DIDN'T INTRUPT ON SOFT EROR/
7769	031440	044504	047104	052047	
7770	031446	044440	052116	052522	
7771	031454	052120	047440	020116	
7772	031462	047523	052106	042440	
7773	031470	047522	000122		

```

7774
7775 031474 042515 020130 044502 EM135: .ASCIZ /MEX BITS INCRMNTD WRONG-RKCS/
7776 031502 051524 044440 041516
7777 031510 046522 052116 020104
7778 031516 051127 047117 026507
7779 031524 045522 051503 000
7780
7781 031531 127 051520 047040 EM137: .ASCIZ /WPS NOT CLEAR/
7782 031536 052117 041440 042514
7783 031544 051101 000
7784
7785 031547 104 052101 020101 EM140: .ASCIZ /DATA EROR ON TRANSFER FROM DISK TO TTY/
7786 031554 051105 051117 047440
7787 031562 020116 051124 047101
7788 031570 043123 051105 043040
7789 031576 047522 020115 044504
7790 031604 045523 052040 020117
7791 031612 052124 000131
7792
7793 031616 042047 044522 020126 EM141: .ASCIZ /'DRIV #' PRESENT, BUT NOT INDICATED/
7794 031624 023443 050040 042522
7795 031632 042523 052116 020054
7796 031640 052502 020124 047516
7797 031646 020124 047111 044504
7798 031654 040503 042524 000104
7799 031662 047040 020117 052502 EM142: .ASCIZ / NO BUSY ON OTHER HALF OF RK-05F/
7800 031670 054523 047440 020116
7801 031676 052117 042510 020122
7802 031704 040510 043114 047440
7803 031712 020106 045522 030055
7804 031720 043065 000
7805
7806
7807
7808
7809
7810 031724 .EVEN
7811
7812 .SBTTL ERROR DATA POINTERS
7813
7814 031724 001116 001162 000000 DT1: .WORD $ERRPC,$REG0,0
7815
7816 031732 001116 001162 001164 DT2: .WORD $ERRPC,$REG0,$REG1,0
7817 031740 000000
7818
7819 031742 001116 001162 001164 DT20: .WORD $ERRPC,$REG0,$REG1,$REG2,$REG3,0
7820 031750 001166 001170 000000
7821
7822 031756 001116 000000 DT21: .WORD $ERRPC,0
7823
7824 031762 001116 001162 001164 DT26: .WORD $ERRPC,$REG0,$REG1,$REG2,0
7825 031770 001166 000000
7826
7827 031774 001116 001162 001164 DT54: .WORD $ERRPC,$REG0,$REG1,$REG2,$REG3,$REG4,$REG5,$REG6,$REG7,0
7828 032002 001166 001170 001172
7829 032010 001174 001176 001200
  
```

					.SBTTL ERROR HEADERS				
7830	032016	000000							
7831									
7832									
7833									
7834									
7835									
7836									
7837	032020	020040	041520	020040	DH2:	.ASCIZ	/ PC	REGADD	RECVD/
7838	032026	051040	043505	042101					
7839	032034	020104	020040	051040					
7840	032042	041505	042126	000					
7841									
7842	032047	040	050040	020103	DH4:	.ASCIZ	/ PC	EXPCT	RECVD/
7843	032054	020040	042440	050130					
7844	032062	052103	020040	051040					
7845	032070	041505	042126	000					
7846									
7847	032075	040	050040	020103	DH5:	.ASCIZ	/ PC	RECVD/	
7848	032102	020040	051040	041505					
7849	032110	042126	000						
7850									
7851	032113	040	050040	020103	DH14:	.ASCIZ	/ PC	RKCS	RKER RKWC/
7852	032120	020040	051040	041513					
7853	032126	020123	020040	051040					
7854	032134	042513	020122	020040					
7855	032142	051040	053513	000103					
7856									
7857	032150	020040	041520	000	DH21:	.ASCIZ	/ PC/		
7858									
7859	032155	040	050040	020103	DH30:	.ASCIZ	/ PC	RKCS	RKER RKDS/
7860	032162	020040	020040	045522					
7861	032170	051303	020040	020040					
7862	032176	045522	051105	020040					
7863	032204	020040	045522	051504					
7864	032212	000							
7865									
7866	032213	040	050040	020103	DH34:	.ASCIZ	/ PC	RKDS/	
7867	032220	020040	020040	045522					
7868	032226	051504	000						
7869									
7870	032231	040	050040	020103	DH35:	.ASCIZ	/ PC	SEC-CNTR/	
7871	032236	020040	042523	026503					
7872	032244	047103	051124	000					
7873									
7874	032251	040	050040	020103	DH36:	.ASCIZ	/ PC	PRSNT	NXT-CNT/
7875	032256	020040	020040	051120					
7876	032264	047123	020124	047040					
7877	032272	052130	041455	052116					
7878	032300	000							
7879									
7880	032301	040	050040	020103	DH40:	.ASCIZ	/ PC	SECTOR	RKDS/
7881	032306	020040	051440	041505					
7882	032314	047524	020122	020040					
7883	032322	045522	051504	000					
7884									
7885	032327	040	050040	020103	DH44:	.ASCIZ	/ PC	RKCS	RKER RKDS RKDA/

MD-11-CZRKKF, RK11 BASIC LOGIC TEST 2
CZRKKF.P11 21-FEB-78 08:51

D 12
MACY11 30A(1052) 21-FEB-78 08:58 PAGE 147
ERROR HEADERS

SEQ 0146

7998
7999
8000 033342 000400
8001
8002
8003
8004 000001

;DATA BUFFER
OUTBUF: .BLKW 256. ;THIS 256 WORD BUFFER IS FOR
;DATA TRANSFERS FROM AND
;TO THE DISK.
.END

INDX2	001360	1134#	2465*	2469*	3652*	3664*	5781*	5872*	5954	5975*				
IOTVEC=	000020	1001#	1837*	1838*										
LF =	000012	907#	6846	6852										
MSG1	001216	1084#	2045											
MSG2	001236	1088#	2080											
MSG3	001245	1091#	6570											
MSG4	001272	1096#	2288	6724										
MSG5	001303	1099#	6710											
MSG6	001315	1102#	6728											
NUDRV	005040	2250#	5824											
ODDEVN	001406	1153#	5861*	5910	5913	6055*	6056							
OUTBUF	033342	2788	2813	2855	2857	2909	2951	2953	3011	3053	3055	3141*	3146	3237
		3272	3274	3334	3409	3521*	3525	3651	3781	3964	3968	3972	4014	4147
		4160	4247	4253	4277	4324*	4327	4365	4367	4372	4378	4380	4411	4418
		4462	4464	4503	4789	4829	4831	4943	5319	5392	5402*	5406	5461*	5467
		5525*	5532	5677	5893	5898	5982	6013	8000#					
PFSTRT	004702	2174#	7264											
PHYDRV	001436	1180#	5850*	5854*	5981	6012								
PIRQ =	177772	913#												
PIRQVE=	000240	1007#												
PRO =	000000	930#												
PR1 =	000040	931#												
PR2 =	000100	932#												
PR3 =	000140	933#												
PR4 =	000200	934#												
PR5 =	000240	935#												
PR6 =	000300	936#												
PR7 =	000340	937#												
PS =	177776	910#	911	1894										
PSW =	177776	911#												
PWRVEC=	000024	1002#	1843*	1844*	7231*	7232*	7241*	7247*	7259*	7260*				
RDCHR =	104410	7113	7207#											
RDLIN =	104411	1968	7208#											
RESVEC=	000010	997#												
RKBA	001336	1116#	2815	2908	3010	3146*	3238*	3272	3275	3358*	3525*	3653*	3805*	3964*
		4020*	4086*	4160*	4195	4198	4253*	4323	4418*	4462	4465	4510*	4534	4537
		4784	4943*	5085*	5319*	5406*	5467*	5532*	5600*	5678	5751			
RKCS	001332	1114#	2327*	2608	2668	2741*	2814	2907	3008	3120	3152	3233	3240*	3242
		3331	3529*	3531	3660*	3662	3810*	3812	3900	4002	4132	4172	4256*	4258
		4320	4409	4501	4612	4672	4782	4861	4946*	4949	4967	5019*	5035	5083
		5088	5151	5220	5315	5379	5460	5537*	5566*	5579	5666	5758*	5866	6133
		6265	6340	6342	6343	6392*	6523	6559*	6563	6583	7283*			
RKDA	001340	1117#	1818*	2042	2051*	2052	2201*	2321*	2367*	2404*	2416*	2456*	2544*	2567*
		2610*	2669*	2738*	2739*	2818*	2913*	3015*	3122	3235	3357*	3527*	3658*	3694
		3699	3808*	3922*	3957	3960	3963*	4018*	4078	4081	4087*	4161*	4254*	4322
		4419*	4456	4459	4511*	4673*	4680*	4749*	4783	4862*	4944*	5017*	5086*	5154*
		5241*	5252	5316*	5317*	5407*	5468*	5486	5491	5533*	5551	5556	5599*	5683*
		5697*	5757*	5928*	6130	6284	6290	6306	6360*	6391*	6425*	7282*	7285*	
RKDB	001342	1118#	3084	3087	3099	3101	3102	3181	3186	3462	3464	3465	3854	3857
RKDS	001326	1112#	1820	2041	2063	2202	2320	2345	2368	2377	2380	2405	2418	2422
		2457	2505	2542	2569	2619	2630	2679	2685	2740	3545	3679	3827	3901
		4051	4273	4540	4725	4750	4752	5157	5965	5983	6131	6362	6366	6399
		6426	7287											
RKER	001330	1113#	2699	3902	4344	4346	4438	4961	5016	5098	5168	5249	5328	5339
		5414	5476	5480	5623	5629	6132	6316	6329	6331	6332			
RKPRI	001400	1144#	1897*	4865	5945	6487								

TSTEND	020626	6028	6055#												
TSTRWS	021436	2750	5128	5207	5236	5256	5585	5752	6360#						
TSTSIN	021572	6425#	7224												
TST.Si=	104421	2666	2732	2786	2905	3006	3138	3231	3355	3518	3648	3802	3916	4130	
		4244	4318	4407	4499	4606	4779	4857	4937	5014	5080	5149	5239	5390	
		5458	5523	5744	7224#										
TST1	004706	2110	2192#												
TST10	005742	2481	2503	2508	2515	2530#									
TST11	006030	2554	2565#												
TST12	006110	2579	2597#												
TST13	006252	2642	2655#												
TST14	006416	2705	2721#												
TST15	006510	2775#													
TST16	006746	2866	2893#												
TST17	007174	2970	2995#												
TST2	005040	2259#													
TST20	007504	3100	3119#												
TST21	007702	3219#													
TST22	010154	3300	3328#												
TST23	010524	3463	3497#												
TST24	011004	3614#													
TST25	011414	3733	3749	3777#											
TST26	011722	3867	3894#												
TST27	012242	3998#													
TST3	005236	2309#													
TST30	012530	4001	4100	4119#											
TST31	012764	4215	4233#												
TST32	013170	4274	4294	4307#											
TST33	013436	4376	4396#												
TST34	013720	4470	4488#												
TST35	014176	4541	4559	4584	4595#										
TST36	014330	4660#													
TST37	014654	4758	4768#												
TST4	005350	2341	2356#												
TST40	015070	4830	4846#												
TST41	015274	4901	4926#												
TST42	015456	5003#													
TST43	015564	5069#													
TST44	015740	5138#													
TST45	016076	5163	5219#												
TST46	016304	5291	5304#												
TST47	016456	5378#													
TST5	005436	2381	2393#												
TST50	016634	5447#													
TST51	016762	5489	5512#												
TST52	017134	5578#													
TST53	017330	5647#													
TST54	017622	5659	5717	5742#											
TST55	017760	5797#													
TST56	020004	5814#													
TST57	020054	5845#													
TST6	005464	2406	2415#												
TST7	005522	2419	2445#												
TYERM	021026	2055	2086	6148#											
TYPDS =	104405	6085	7202#												
TYPE =	104401	1872	1924	1936	1945	1949	1964	1988	2044	2066	2079	2120	2140	2142	

COMMEN	1#	1008#														
ENDCOM	1#	1008#														
ERROR	902#	2212	2223	2325	2336	2346	2372	2378	2383	2408	2423	2501	2506	2512	2519	
	2558	2577	2581	2614	2627	2633	2638	2645	2674	2684	2688	2697	2702	2708	2735	
	2747	2751	2824	2836	2843	2849	2853	2859	2863	2869	2919	2932	2938	2944	2949	
	2955	2959	2967	2973	3021	3034	3041	3047	3051	3057	3061	3069	3088	3095	3103	
	3162	3168	3177	3187	3251	3258	3267	3276	3280	3289	3297	3303	3372	3379	3392	
	3397	3421	3450	3466	3540	3549	3554	3561	3672	3684	3689	3702	3717	3728	3821	
	3831	3835	3845	3858	3927	3937	3942	3946	3954	3961	3973	4037	4069	4075	4082	
	4104	4168	4181	4189	4199	4204	4208	4212	4218	4267	4289	4333	4337	4342	4347	
	4354	4362	4369	4381	4426	4430	4436	4441	4452	4460	4466	4473	4525	4529	4538	
	4552	4568	4622	4640	4675	4690	4701	4710	4723	4736	4753	4760	4800	4810	4817	
	4821	4833	4886	4912	4957	4965	4970	4985	4989	4992	5031	5038	5053	5057	5094	
	5102	5107	5122	5126	5130	5172	5187	5201	5205	5209	5238	5245	5253	5258	5261	
	5273	5288	5294	5333	5340	5347	5362	5366	5412	5418	5423	5428	5432	5472	5481	
	5492	5542	5557	5586	5608	5618	5630	5694	5710	5724	5753	5761	5773	5802	5819	
	5926	5962	5969	6003	6021	7274	7292									
ESCAPE	1#	1008#														
GETPRI	1#	1008#														
GETSWR	1#	852#	1008#	1873#												
MESSAGE	2177#	2180	2253#	2255	2410#	2412	2427#	2430	2524#	2527	2588#	2590	2648#	2650	2711#	
	2713	2760#	2762	2872#	2874	2978#	2981	3105#	3107	3200#	3202	3308#	3311	3470#	3472	
	3594#	3597	3753#	3756	3885#	3888	3989#	3992	4110#	4113	4221#	4224	4300#	4303	4387#	
	4389	4475#	4477	4587#	4589	4651#	4653	4762#	4764	4836#	4838	4918#	4920	4995#	4997	
	5060#	5062	5132#	5134	5212#	5214	5297#	5299	5369#	5371	5435#	5437	5502#	5505	5568#	
	5571	5637#	5639	5731#	5734	5791#	5793	5806#	5808	5828#	5831					
MORETA	1029#	1084														
MULT	1#	1008#														
NEWTST	1#	1008#	2178	2253	2306	2353	2390	2410	2428	2525	2562	2588	2648	2711	2760	
	2872	2979	3105	3200	3309	3470	3595	3754	3886	3990	4111	4222	4301	4387	4475	
	4587	4651	4762	4836	4918	4995	5060	5132	5212	5297	5369	5435	5503	5569	5637	
	5732	5791	5806	5829												
POP	1#	1008#	6906	7252	7253											
PRIOR7	1198#	4608	4645	4743	4824	4897	5561	5662	5899							
PUSH	1#	1008#	6865	7233	7239											
REPORT	1#	1008#														
SCOPE	903#	2192	2259	2309	2356	2393	2415	2445	2530	2565	2597	2655	2721	2775	2893	
	2995	3119	3219	3328	3497	3614	3777	3894	3998	4119	4233	4307	4396	4488	4595	
	4660	4768	4846	4926	5003	5069	5138	5219	5304	5378	5447	5512	5578	5647	5742	
	5797	5814	5845	6072												
SETPRI	1#	1008#														
SETTRA	7190#	7199	7200	7201	7202	7204	7206	7207	7208	7210	7212	7214	7216	7218	7220	
	7222	7224														
SETUP	1#	1008#	1828													
SKIP	1#	1008#	2341	2381	2406	2419	2481	2503	2508	2515	2554	2579	2642	2705	2866	
	2970	3100	3300	3463	3733	3749	3867	4001	4100	4215	4274	4294	4376	4470	4541	
	4559	4584	4758	4830	4901	5163	5291	5489	5659	5717						
SLASH	1#	1008#														
SPACE	1008#															
STARS	1#	1008#	1021	1031	1083	2178	2191	2253	2258	2306	2308	2353	2355	2390	2392	
	2410	2414	2428	2444	2525	2529	2562	2564	2588	2596	2648	2654	2711	2720	2760	
	2774	2872	2892	2979	2994	3105	3118	3200	3218	3309	3327	3470	3496	3595	3613	
	3754	3776	3886	3893	3990	3997	4111	4118	4222	4232	4301	4306	4387	4395	4475	
	4487	4587	4594	4651	4659	4762	4767	4836	4845	4918	4925	4995	5002	5060	5068	
	5132	5137	5212	5218	5297	5303	5369	5377	5435	5446	5503	5511	5569	5577	5637	
	5646	5732	5741	5791	5796	5806	5813	5829	5844	6064	6597	6736	6784	6855	6923	

MD-11-CZRKKF, RK11 BASIC LOGIC TEST 2
CZRKKF.P11 21-FEB-78 08:51

E 13
MACY11 30A(1052) 21-FEB-78 08:58 PAGE 163
CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0160

.\$TYPD	1#	852#	6853
.\$TYPE	1#	852#	6782
.\$TYPO	1#	852#	6921
.\$40CA	1#		
.1170	1#		

. ABS. 034342 000

ERRORS DETECTED: 0

CZRKKF,CZRKKF.LST/CRF/SOL=CZRKKF.SML,CZRKKF.P11
RUN-TIME: 22 31 1 SECONDS
RUN-TIME RATIO: 392/55=7.0
CORE USED: 34K (67 PAGES)