IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZRKI-D-D
PRODUCT NAME:	RK11 UTILITY PACKAGE
DATE CREATED:	DECEMBER, 1976
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	BOB COLLINS
REVISED BY:	JIM KAPADIA TOM SAWYER CHUCK HESS

TABLE OF CONTENTS

1. ABSTRACT

1.1 THIS PACKAGE CONTAINS 4 INDIVIDUAL UTILITY PROGRAMS FOR THE RKXX PLUS A MINI-MONITOR WHICH ALLOWS TEST SELECTION AND PARAMETER INPUT VIA THE CONSOLE DEVICE. ALL UTILITY PACKAGES ARE EXPLAINED IN DETAIL IN PARAGRAPH 9.

2. REQUIREMENTS

2.1 EQUIPMENT
PDP-11 PROCESSOR
8K MEMORY
RK11 OR RKV11 CONTROLLER
1-8 RK05 OR RK05F DISK DRIVES (DRIVE TYPES MAY BE MIXED)

2.2 STORAGE
THIS PROGRAM REQUIRES 8K

2.3 PRELIMINARY PROGRAMS
THIS IS NOT A DIAGNOSTIC, PACKAGE IT IS ASSUMED THAT ALL EQUIPMENT IS FUNCTIONAL

3. LOADING PROCEDURE

3.1 METHOD
PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED
A. ABSOLUTE LOADER MUST BE IN MEMORY.
B. PLACE BINARY TAPE IN READER.
C. LOAD ADDRESS *7500 (*DETERMINED BY LOCATION OF LOADER).
D. PRESS "START" PROGRAM WILL LOAD.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS
NONE

4.2 STARTING ADDRESS
200 MINI MONITOR

4.3 PROGRAM AND/OR OPERATOR ACTION
LOAD PROGRAM INTO MEMORY
SET SWITCH REGISTER TO STARTING ADDRESS (200)
LOAD ADDRESS
PRESS START

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:

CTRL E

'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.

PROGRAM WILL TYPE MINI MONITOR ROUTINE

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS
SEE SEC. 9.0 FOR SWITCHES APPLICABLE TO INDIVIDUAL ROUTINES.

5.2 SUBROUTINE ABSTRACTS
NOT APPLICABLE

5.3 PROGRAM AND/OR OPERATOR ACTOR
SEE INDIVIDUAL PACKAGE DESCRIPTION (PARAGRAPH 9)

6. ERRORS

6.1 ERROR HALTS AND DESCRIPTION
IF HALTED A MAJOR PROBLEM EXIST CHECK CODE AT HALT PC TO DETERMINE WHAT OCCURRED.

6.2 ERROR RECOVERY
EXPLAINED IN DETAIL IN INDIVIDUAL PACKAGE DESCRIPTION (PARAGRAPH 9)

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS
IT IS NOT RECOMMENDED THAT YOU START AT AN ADDRESS OTHER THAN 200, (REASON EXPLAINED IN PARAGRAPH 9.1) UNLESS DIRECTED TO BY THE PROGRAM.

7.2 OPERATIONAL RESTRICTIONS
EXPLAINED IN DETAIL IN INDIVIDUAL PACKAGE DESCRIPTIONS (PARAGRAPH. 9)

8. EXECUTION TIME

VARIES WITH SELECTED ROUTINE, NUMBER OF DRIVES, ETC.

9. PROGRAM DESCRIPTION

THE RK11 UTILITY PACKAGE IS DIVIDED INTO EIGHT SECTIONS
WHICH ALLOW COMPATABILITY TESTING, OSCILLATING SEEKS FOR SERVO
ADJUSTMENT AND SEEK LOGIC WAVEFORM ANALYSIS, PACK FORMATTING
AND SURFACE VERIFICATION, AND FRONT PANEL TESTING (INDICATOR LAMPS,
SWITCHES, INTERLOCKS, ETC) AND VERIFICATION.
THE PACKAGE IS DIVIDED INTO FIVE SECTIONS

SECTION	NAME
0	INDEX
1	COMPATIBILITY TEST
2	OSCILLATING SEEK PACKAGE
3	FORMATTER SURFACE VERIFIER
4	FRONT PANEL TEST
5	RK05 CONTROL PANEL TEST #2
6	HEAD ALIGNMENT ROUTINE
7	POWER FAILURE (DURING WRITE) TEST

NOTE: NORMAL LINKAGE TO ANY OF THESE PACKAGES IS THRU
SECTION 0 (SEE PARAGRAPH 9.1)

9.1 SECTION 0 INDEX

PURPOSE: TO ALLOW THE USER TO SELECT AND RUN TESTS VIA THE CONSOLE DEVICE IN AN EFFORT TO FREE HIM FROM REMEMBERING VARIOUS SWITCH SETTINGS.

DESCRIPTION: LOAD START ADDRESS 200, A TABLE IS PRODUCED WHICH TELLS THE USER THE NAME AND TYPE OF THE TEST. (TYPE IS AN OCTAL CODE BY WHICH THE USER SELECTS THE TEST). AFTER THE TABLE IS TYPED, THE QUESTION "TYPE =" IS ASKED, THE USER THEN TYPES THE NUMERAL 0-④ ? ?
TO SELECT A TEST.

USE: THIS IS EXAMPLE OF THE ACTUAL OUTPUT:

RK11 UTILITY PACKAGE

NAME	TYPE
INDEX	0
COMPATIBILITY PACKAGE	1
OSCILLATING SEEK PACKAGE	2
FORMATTER-SURFACE VERIFIER	3
RK05 CONTROL PANEL TEST	4
RK05 CONTROL PANEL TEST #2	5
HEAD ALIGNMENT ROUTINE	6
POWER FAILURE (WRITE) TEST	7

TYPE=X

WERE "X" IS THE RESPONSE (0-7) BY THE USER

ERROR INFO: ANY ILLEGAL INPUT IS HANDLED, A QUESTION MARK IS TYPED AND THE QUESTION "TYPE =" IS RE-ASKED.

9.2 SECTION 1 COMPATIBILITY PACKAGE *

PURPOSE: TO CONFIRM THE FACT THAT A GROUP OF DRIVES (A MAXIMUM OF EIGHT) ARE TRULY COMPATIBLE.
THIS PACKAGE DOES NOT APPLY TO RK-05F DRIVES.

DESCRIPTION: THIS PACKAGE ALLOWS A USER TO AUTOMATICALLY TEST

* ONLY ONE PACK IS USED - IT IS MOVED FROM DRIVE TO DRIVE

COMPATIBILITY OF UP TO EIGHT (8) DRIVES SIMPLY
BY STATING THE DRIVE NUMBERS TO BE TESTED. THE
TEST DOES THE REST, INSTRUCTING THE USER WHERE
TO PLACE THE PACK. THE LIMITATIONS OF TESTING ARE
IF THERE ARE (2) TWO PROCESSORS, FROM ONE (1) TO
SEVEN (7) DRIVES MAY BE ON SYSTEM ONE, AND
ONLY ONE (1) DRIVE (ANY DRIVE NUMER) MAY
BE ON SYSTEM TWO.
COMPATIBILITY-A DEFINITION, COMPATIBILITY INFERS
MORE THAN THE FACT THAT INFORMATION WHICH
WAS WRITTEN ON ONE DRIVE CAN BE
READ ON ANOTHER.
FOR DRIVES TO BE CONSIDERED TRULY COMPATIBLE
ANY DRIVE SHOULD BE ABLE TO READ WHAT WAS
WRITTEN BY ANY OTHER DRIVE AND ALSO MUST BE ABLE
TO OVERWRITE A PORTION OF INFORMATION WRITTEN
BY ANOTHER DRIVE, WITH NEW INFORMATION, AND
READ IT BACK. THIS IS A VERY BROAD DEFINITION
BUT IS THE BASIC PREMISE OF TRUE COMPATIBILITY.

USE: THE BELOW IS AN EXAMPLE OF ACTUAL OUTPUT, THE USER
WANTS TO RUN SINGLE PROCESSOR MODE AND TEST
COMPATIBILITY ON THREE (3) DRIVES WHOSE UNIT NUMBERS
ARE Ø,1,3.....

****

EXAMPLE 1

```
          NAME                      TYPE
   INDEX                             Ø
   COMPATIBILITY PACKAGE             1
   OSCILLATING SEEK PACKAGE          2
   FORMATTER-SURFACE VERIFIER        3
   RKØ5 CONTROL PANEL TEST           4
   RKØ5 CONTROL PANEL TEST #2        5
   HEAD ALIGNMENT ROUTINE            6
   POWER FAILURE (WRITE) TEST        7

   TYPE=1
   DRIVE NUMBERS ON SYSTEM 1=Ø,1,3.

   IS THERE A SECOND SYSTEM?N
   MOUNT PACK ON DRIVE #Ø
   MAKE PACK WRITE ENABLE
   PRESS CONTINUE WHEN DRIVE RDY
   MOUNT PACK ON DRIVE #1
   MAKE PACK WRITE ENABLE
   PRESS CONTINUE WHEN DRIVE RDY
   MOUNT PACK ON DRIVE #3
   MAKE PACK WRITE ENABLE
   PRESS CONTINUE WHEN DRIVE RDY
   MOUNT PACK ON DRIVE #Ø
   MAKE PACK WRITE ENABLE
   PRESS CONTINUE WHEN DRIVE RDY
   MOUNT PACK ON DRIVE #1
   MAKE PACK WRITE ENABLE
   PRESS CONTINUE WHEN DRIVE RDY
   MOUNT PACK ON DRIVE #3
   MAKE PACK WRITE ENABLE
   PRESS CONTINUE WHEN DRIVE RDY
   DONE!
```

```
        RK11 UTILITY PACKAGE

        NAME                        TYPE
INDEX                                0
COMPATIBILITY PACKAGE                1
                          **********

        ...THE USER SELECTED TYPE ONE (1) AND RECEIVED
THE MESSAGE RKXX COMPATIBILITY PACKAGE AND WAS THEN
ASKED FOR SYSTEM 1 DRIVES HE TYPES EACH SELECTED
DRIVE NUMBER SEPARATED BY COMMAS HE TERMINATES THE
STRING WITH A PERIOD THEN A CARRIAGE RETURN
HE IS ASKED IF THERE IS A SECOND SYSTEM,
HE TYPES N FOR NO.
HE NOW RECEIVES A STRING OF MOVE DIRECTIVES TELLING
HIM EXACTLY WHERE TO MOVE THE TEST PACK AND WHAT TO
DO.  FINALLY THE USER RECEIVES THE MESSAGE "DONE!"
INDICATING A SUCCESSFUL PASS.
AT THIS POINT ANY DRIVE WHICH HAS NOT BEEN
DECLARED DOWN AND DID NOT RECEIVE AN ERROR*
MESSAGE IS COMPATIBLE WITH ANY OTHER SELECTED
DRIVE MEETING THE SAME CONDITIONS.
FINALLY THE INDEX ROUTINE IS AUTOMATICALLY
RE-ENTERED AND USER IS READY TO MAKE ANOTHER
SELECTION.
*SEE ERROR INFO TO DETERMINE THE TYPE OF ERROR
WHICH CONSTITUTES INCOMPATABILITY.
                          **********
```

EXAMPLE 2

```
        THE USER NOW DESIRES TO TEST COMPATIBILITY ON
        TWO SYSTEMS HE HAS UNITS 0,1 ON SYSTEM ONE
        AND UNIT 0 ON SYSTEM 2, IT GOES LIKE THIS....
        RK11 UTILITY PACKAGE

        NAME                        TYPE
INDEX                                0
COMPATIBILITY PACKAGE                1
OSCILLATING SEEK PACKAGE             2
FORMATTER-SURFACE VERIFIER           3
RK05 CONTROL PANEL TEST              4
RK05 CONTROL PANEL TEST #2           5
HEAD ALIGNMENT ROUTINE               6
POWER FAILURE (WRITE) TEST           7


TYPE=1
DRIVE NUMBERS ON SYSTEM 1=1,0

IS THERE A SECOND SYSTEM?Y
DRIVE # =0
MOUNT PACK ON DRIVE #1
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
MOUNT PACK ON DRIVE #0
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
LOAD AND START ADDRESS 210 ON SYSTEM #2
AND TYPE THE BELOW WHEN ASKED ON SYSTEM #2
```

```
AND TYPE THE BELOW WHEN ASKED FOR IT.
WORD            1=000002
WORD            2=000200

                        **********
        ....THE ONLY DIFFERENCE BETWEEN THIS AND SINGLE
SYSTEM IS THE NEW DIRECTIVE TO LOAD START 210
ETC.  THE USER NOW LOADS AND STARTS SYSTEM TWO
AND THE BELOW IS TYPED...
                        **********

COMPATIBILITY-SYSTEM#2
WORD            1=000002

WORD            2=000200

MOUNT PACK ON DRIVE #0
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
DONE SYSTEM 2 RESTART SYSTEM 1, TYPE WORD 000077

                        **********
        .....THE USER RESPONSE TO THE QUESTION WORD 1 =
BY TYPING WORD 1 FROM PROCESSOR ONE AND
WORD 2 =, BY TYPING WORD TWO FROM PROCESSOR 1
HE RECEIVES THE MOUNT COMMAND MOVES THE TEST PACK
TO SYSTEM TWO, DRIVE NUMBER (0), AND PRESSES
CONTINUE.  NOW THE MESSAGE TO RETURN TO SYSTEM
ONE*

*SYSTEM ONE HAS BEEN IN A HALT STATE AND
SHOULD BE LEFT THAT WAY UNTIL THE RETURN FROM
SYSTEM TWO SO THAT TABLES, ETC. BUILT FOR THE
TEST WILL NOT BE DISTURBED.
                        **********

WORD=000077

MOUNT PACK ON DRIVE #1
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
MOUNT PACK ON DRIVE #0
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
DONE!
        RK11 UTILITY PACKAGE

        NAME                    TYPE
INDEX                            0
COMPATIBILITY PACKAGE            1
OSCILLATING SEEK PACKAGE         2
FORMATTER-SURFACE VERIFIER       3
RK05 CONTROL PANEL TEST          4
RK05 CONTROL PANEL TEST #2       5
HEAD ALIGNMENT ROUTINE           6
POWER FAILURE (WRITE) TEST       7

TYPE=

                        **********
```

THE USER NOW PRESSES CONTINUE ON PROCESSOR ONE
AND IN RESPONSE TO THE QUESTION, WORD =, TYPES
THE WORD GIVEN TO HIM FROM PROCESSOR TWO
THEN EVERYTHING BECOMES THE SAME AS A SINGLE
SYSTEM. THE USER MEARLY FOLLOWS DIRECTIONS.
ERROR INFO: SEE PARAGRAPH 9.6 SPECIAL SECTION

9.3 SECTION 2 OSCILLATING SEEK PACKAGE

PURPOSE: TO ALLOW THE USER TO MAKE SERVO ADJUSTMENTS
AND/OR SEEK LOGIC CHECKOUT BY PERFORMING
SEEKS BETWEEN USER SPECIFIED ADDRESS
DESCRIPTION: SELECT TYPE 2, THE USER THEN INSERTS
THE DRIVES TO BE TESTED IN SW0 TO SW7
OF THE SWITCH REGISTER. A SWITCH IS SET
FOR EACH DRIVE (E.G. SW2 TO TEST DRIVE 2.
THE USER THEN INSERTS THE
ADDRESS TO SEEK IN THE SWR. IF BOTH ADDRESS
ARE LEGAL, 50 CYCLES (100 SEEKS) WILL BE
MADE BETWEEN THE SPECIFIED ADDRESS THEN
THE PROGRAM WILL LOOK AT THE SWR FOR
POSSIBLE CHANGES THIS SHOULD ALLOW FOR GOOD
STABLE TRACES ON AN OSCILLISCOPE.
IT SHOULD BE NOTED THAT THE OSCILLATING
SEEKS BETWEEN THE SPECIFIED CYLINDERS
ARE DONE ON ALL AVAILABLE DRIVES.
THE ONLY WAY TO EXIT IS HALT!, LOAD ADDRESS 200,
HIT START.
USE: SELECT TYPE 2, RESPOND TO QUESTION WITH UNIT NUMBER...
TYPE=2
OSCILLATING SEEK PACKAGE
SET SW0 TO SW7 TO SELECT THE DRIVES TO TEST
AND CONTINUE. IF ALL SWITCHES ARE RESET,
ALL AVAILABLE DRIVES WILL BE TESTED.
TOGGLE THE "FIRST CYLINDER ADDRESS" (OUTER LIMIT)
INTO THE LOW BYTE (BIT0-7) OF THE SWITCH REGISTER AND THE "LAST
CYLINDER ADDRESS" (INNER LIMIT) INTO THE HIGH
BYTE (BIT8-15), THEN PRESS CONTINUE
...FOLLOW INSTRUCTIONS TYPED
ERROR INFO: IF AN ILLEGAL ADDRESS IS SELECTED A MESSAGE IS TYPED
AND USER MEARLY SELECTS LEGAL ADDRESS AND DEPRESSES
CONTINUE
EXAMPLE TYPEOUT
INVALID ADDRESS IN SWITCH REGISTER TRY AGAIN
INVALID ADDRESS IN SWITCH REGISTER TRY AGAIN
INVALID ADDRESS IN SWITCH REGISTER TRY AGAIN

NOTE: BOTH DRIVES OF AN RK-05F SHOULD NOT BE SELECTED
FOR TESTING AT THE SAME TIME.

9.4 SECTION 3 FORMATTER-SURFACE VERIFIER

PURPOSE: TO FORMAT VIRGIN PACKS OR REFORMAT AN OLDER
PACK AND VERIFY ITS SURFACE
DESCRIPTION: SELECT TYPE 3, RESPOND TO THE QUESTION
BY SETTING SWITCHES CORRESPONDING TO
DRIVE NUMBERS TO BE FORMATTED. THUS IF
DRIVES 0,1,2 ARE TO BE FORMATTED SET
SWITCHES 0,1,2. THE DRIVES ARE FORMATTED
ONE AFTER ANOTHER AT COMPLETION PACK GOOD

```
                MESSAGE IS TYPED AND PACK IS FORMATTED.
USE:            SELECT TYPE 3, RESPOND TO QUESTION WITH
                SETTING OF SWITCH REGISTER.
                          **********

             RK11 UTILITY PACKAGE

             NAME                      TYPE
   INDEX                               Ø
   COMPATIBILITY PACKAGE               1

   OSCILLATING SEEK PACKAGE            2
   FORMATTER-SURFACE VERIFIER          3
   RKØ5 CONTROL PANEL TEST             4
   RKØ5 CONTROL PANEL TEST #2          5
   HEAD ALIGNMENT ROUTINE              6
   POWER FAILURE (WRITE) TEST          7

   TYPE=3
   FORMATTER-SURFACE VERIFIER, SET SW REG WITH DRIVE #'S

   PACK GOOD.
           RK11 UTILITY PACKAGE

           NAME                      TYPE
   INDEX                             Ø
   COMPATIBILITY PACKAGE             1
                          **********
   AFTER THE PACK IS FORMATTED A GOOD MESSAGE IS
   GIVEN AND A CHECK IS MADE TO SEE IF THERE ARE
   ANY MORE PACKS TO BE FORMATTED.  IF THERE ARE
   NONE CONTROL IS TRANSFERRED TO THE MINI-MONITOR
ERROR INFO: DRIVE PROBLEM, IF THE MESSAGE....
            SYSTEM ERROR
            ....IS TYPED IT INDICATES A FAULTY DRIVE OR
            CONTROLLER, RUN DIAGNOSTICS, THE PROCESSOR WILL HALT
            PRESS CONTINUE TO RETURN TO MINI MONITOR.
            BAD SPOT, OR SURFACE PROBLEM, ETC.

            PACK FAILED AT (IN OCTAL) CYLINDER SECTOR SURFACE

9.5 SECTION 4    RKØ5 CONTROL PANEL TEST

PURPOSE:         TO INSURE ALL SWITCHES INDICATOR LAMPS, AND INTERLOCKS
                 ARE FUNCTIONAL IN THE RKØ5
DESCRIPTION:     SELECT TYPE 4, RESPOND TO QUESTION WITH UNIT NUMBER, FOLLOW
                 DIRECTIONS GIVEN.  AT COMPLETION MESSAGE "DONE!" IS GIVEN
USE:             SELECT TYPE 4, RESPOND TO QUESTION WITH THE UNIT NUMBER....
                          **********
           NAME                      TYPE
   INDEX                             Ø
   COMPATABILITY PACKAGE             1
   OSCILLATING SEEK PACKAGE          2
   FORMATTER-SURFACE VERIFIER        3
   RKØ5 CONTROL PANEL TEST           4
   RKØ5 CONTROL PANEL TEST #2        5
   HEAD ALIGNMENT ROUTINE            6
   POWER FAILURE (WRITE) TEST        7

   TYPE=4
```

```
RK05 CONTROL PANEL TEST, WHICH DRIVE?0
MOUNT PACK ON DRIVE#0
PLACE DRIVE IN RUN ;SHOULD SEE THE RUN,
POWER, AND ON CYLINDER LAMPS LIGHT.
MAKE DRIVE WRITE ENABLE PRESS CONTINUE

WRITE PROTECT THE DRIVE THEN PRESS CONTINUE

CLEAR WRITE PROTECT THEN PRESS CONTINUE

CAUTION! TRY TO OPEN THE DOOR, DO NOT FORCE:
DOOR SHOULD NOT OPEN!
PRESS CONTINUE WHEN FINISHED

PUT DRIVE IN LOAD, WAIT FOR LOAD LIGHT
PRESS CONTINUE WHEN FINISHED

OPEN THE DOOR, PUT DRIVE IN RUN
CAUTION! IF RUN LIGHT ON ERROR! DEPRESS
LOAD IMMEDIATELY, CONTINUE WHEN FINISHED

REMOVE THE PACK, CLOSE THE DOOR
PUT DRIVE IN RUN, DRIVE SHOULD NOT
RUN...INTERLOCKS HAVE BEEN CHECKED
DONE!

        RK11 UTILITY PACKAGE

          NAME                    TYPE
INDEX                              0
COMPATABILITY PACKAGE              1
OSCILLATING SEEK PACKAGE           2
FORMATTER-SURFACE VERIFIER         3
RK05 CONTROL PANEL TEST            4
RK05 CONTROL PANEL TEST #2         5
HEAD ALIGNMENT ROUTINE             6
POWER FAILURE (WRITE) TEST         7

TYPE=
```

****

9.6 SECTION 5 RK05 CONTROL PANEL TEST #2

PURPOSE: TO GIVE A CONTINUOUS MONITORING AND CHECKING CAPABILITY FOR THE FOLLOWING CONDITIONS ON THE VARIOUS DRIVES:
OFF LINE (RDY CLR)/ON LINE (RDY SET)
WRITE PROTECTED/WRITE ENABLED
POWER LOW/POWER UP
SEEK INCOMPLETE/SEEK OK

DESCRIPTION: SELECT TYPE 5, PUT ALL THE DRIVES THAT ARE TO BE MONITORED AND CHECKED ON 'RUN'. NOTE THAT THIS IS IMPORTANT BECAUSE THE PROGRAM HAS TO KNOW WHICH DRIVES ARE TO BE CHECKED.

USE: AFTER HAVING SELECTED TYPE 5 AND PUTTING THE DRIVES THAT ARE TO BE MONITORED ON 'RUN', THE PROGRAM PRINTS OUT ALL THE DRIVES THAT ARE 'ON LINE'.

DRIVE 0 ON LINE
DRIVE 1 ON LINE
DRIVE 2 ON LINE

THE PROGRAM, THEN STARTS SCANNING ALL DRIVES, ONE AFTER THE OTHER. CHECKS IF THE DRIVE IS ON LINE OR OFF LINE (DRY SET OR CLEAR). THEN IT CHECKS IF THE DRIVE IS WRITE ENABLED OR WRITE PROTECTED. THEN A SEEK (TO CYLINDER 1) IS DONE AND 'DPL' BIT IS CHECKED TO SEE IF DRIVE POWER IS LOW OR OK. IF THE DRIVE IS POWERED, IT IS CHECKED IF THE SEEK IS DONE OR SEEK INCOMPLETE OCCURS. WHEN EVER ANY CHANGE IN THE STATUS IS FOUND, IT IS REPORTED. IF THE DRIVEIS PUT ON 'LOAD' AND BACK TO 'RUN', THE PROGRAM CHECKS IF THE DRIVE COMES ON LINE IN THE WRITE ENABLED MODE. IF NOT, AN ERROR MESSAGE (ERROR, NOT WRITE ENABLED) IS REPORTED. THEN THE DRIVE IS WRITE PROTECTED.
EX: IN A SYSTEM UNDER TEST, IF A DRIVE IS PUT ON 'LOAD' BY THE USER IT GETS REPORTED, IF THE USER SET 'WRITE PROT' IT GETS REPORTED. THE MESSAGES APPEAR AS FOLLOWING:

DRIVE 0 OFF LINE
DRIVE 1 WRITE PROTECTED
DRIVE 2 SIN
DRIVE 1 WRITE ENABLED
DRIVE 0 POWER LO
DRIVE 2 SEEK OK
DRIVE 0 POWER OK

NOTE THAT ONLY CHANGES IN STATUS ARE REPORTED. THESE CHANGES HAVE TO BE AFFECTED BY THE USER, IF ANY CHANGE IN STATUS IS NOT DETECTED AND REPORTED BY THE PROGRAM IT MIGHT IMPLY AN ERROR CONDITION.

9.7 SECTION 6 HEAD ALIGNMENT ROUTINE

PURPOSE: TO PROVIDE A FACILITY FOR HEAD ALIGNMENT, WITH DYNAMIC SELECTION OF THE UPPER OR LOWER HEAD.

DESCRIPTION: WHEN THE ROUTINE IS SELECTED THE FOLLOWING MESSAGE APPEARS:
SET SW0=0 FOR SURFACE 0, SW0=1 FOR SURFACE 1,
SET SW1=1 TO TEST CYL 64, SET SW1=0 TO TEST CYLINDER 105.
SW2-15=0
PUT ANY SW FROM 2-15 HI TO SELECT NEW DRIVE

THEN THE FOLLOWING QUESTION IS ASKED:
DRIVE? THE USER
SHOULD TYPE IN THE DRIVE NUMBER THAT HE WANTS TO SELECT. THE DRIVE NUMBER IS SUFFIXED WITH AN 'F' TO TEST RK-05F TYPE DRIVES.

TYPE=6
DRIVE=0<CR>

THE UPPER OR THE LOWER HEAD CAN BE
SELECTED BY SWITCH 0. IF SURFACE 0 IS
TO BE SELECTED, PUT SW 0 TO 0. IF
SURFACE 1 IS TO BE SELECTED PUT SW 0 ON 1.
THE HEADS MAY BE POSITONED AT CYLINDER 64
OR CYLINDER 105. SET SW1=0 FOR CYLINDER 105,
SW1=1 FOR CYLINDER 64.
THE PROGRAM POSITIONS THE HEADS ON THE SELECTED
CYLINDER AND CONTINUOUSLY READS FROM
THE SURFACE SELECTED, IF THE USER WISHES
TO SELECT THE OTHER HEAD OR CYLINDER IT CAN BE
DYNAMICALLY DONE BY FLIPPING SW 0 OR SW 1.
IF SOME OTHER DRIVE IS TO BE SELECTED,
ANY SWITCH BETWEEN SW 2 AND SW 15 SHOULD
BE PUT UP. THE QUESTION - DRIVE? IS
ASKED AGAIN. THIS IS A CONTINUOUS ROUTINE,
HENCE TO EXIT A HALT HAS TO BE DONE.

NOTE ALIGNMENT IS DONE WITH AN RK-05J CARTRIDGE
SO IF AN F TYPE DRIVE IS SELECTED, CYLINDER
64 OF THE RK-05J IS CYLINDER 130 OF THE F DRIVE
(EVEN DRIVE). CYLINDER 105 BECOMES CYLINDER 5
OF THE ODD DRIVE ON THE RK-05F.

9.8 SECTION 7 (DISK) POWER FAILURE (DURING WRITE) TEST

PURPOSE: THIS TEST CHECKS THAT DATA WRITTEN ON THE DISK
IS NOT DESTROYED WHEN THE DISK SENSES A LOSS OF
POWER (POWER FAILS) WHILE DOING A WRITE.

DESCRIPTION: UPON SELECTING THIS TEST, THE PROGRAM FINDS OUT
THE FIRST AVAILABLE DRIVE AND INDICATES IT TO
THE USER BY TYPING A MESSAGE:

DRIVE X X=DRIVE NUMBER 0,1,..7

THEN IT PROCEEDS TO TO WRITE UNIQUE PATTERNS
ON CYLINDERS 0 TO 15 (DECIMAL) OF THAT DRIVE.
THE HEADS ARE THEN POSITIONED ON CYLINDER 10
AND THE USER IS ASKED TO DROP POWER ON THAT
DRIVE:

DROP POWER

MEANWHILE WRITE IS BEING DONE ON CYLINDER 10.
ON GETTING THE ABOVE MESSAGE THE USER SHOULD
DROP THE POWER ON THAT DRIVE. ON SENSING A LOSS
OF POWER, THE PROGRAM WILL ASK THE USER TO PUT
THE POWER ON AGAIN:

POWER ON

ON RECEIVING THE ABOVE MESSAGE THE USER SHOULD
PUT THE POWER ON. ON DETECTING POWER UP THE
PROGRAM PROCEEDS TO CHECK THAT THE DATA WRITTEN
ON CYLINDERS 0 TO 15 WAS INTACT. IF A WRITE
CHECKS ERROR OCCURS (POSSIBLY MEANING THAT
SOME OF THE DATA WAS DESTROYED DURING THE
LOSS OF POWER) IT IS REPORTED AS FOLLOWING:

ERROR, ON POWER-UP, RKDA=XXXX

XXXX IS THE CONTENTS OF RKDA AT THE TIME OF
ERROR.

THE PROGRAM DOES THE ABOVE POWER FAIL TEST

ON ALL DRIVES THAT ARE PRESENT, ONE AFTER THE
OTHER IN A ROUND BOBBIN FASHION. EXIT IS THROUGH
HALT.

9.9 SECTION SPECIAL

```
              FOR THE BELOW EXAMPLES THE FOLLOWING FORMAT
              WILL BE USED.
              THE ACTUAL TYPEOUT          ;COMMENTS ON WHAT
              AND RESPONSE          ;OCCURRED OR WHAT TO DO
            *NOTES IF NECESSARY FOR CLARITY
```

ERROR EXAMPLE 1

```
   FORMATTER-SURFACE VERIFIER       3
   RK05 CONTROL PANEL TEST          4

   TYPE=1                           ;TYPE 1 SELECTION
   DRIVE NUMBERS ON SYTEM 1=0.      ;DRIVE #0 SELECTED

   IS THERE A SECOND SYSTEM?N       ;NO SECOND SYSTEM
   MOUNT PACK ON DRIVE #0           ;CONTINUE PRESSED BUT
   MAKE PACK WRITE ENABLE           ;WRITE PROTECT ON
   PRESS CONTINUE WHEN DRIVE RDY    ;CLEAR WRITE PROTECT SWITCH
   DRIVE WRITE PROTECTED.           ;NOW RUNS TO FINISH
   DRIVE WRITE PROTECTED            ;THIS DOES NOT EFFECT
   DONE!                            ;OUTCOME OF TEST
           RK11 UTILITY PACKAGE

           NAME             TYPE
   INDEX                     0
   COMPATIBILITY PACKAGE     1
   OSCILLATING SEEK PACKAGE  2
```

ERROR EXAMPLE 2

```
           RK11 UTILITY PACKAGE

           NAME                      TYPE
   INDEX                              0
   COMPATIBILITY PACKAGE              1
   OSCILLATING SEEK PACKAGE           2
   FORMATTER-SURFACE VERIFIER         3
   RK05 CONTROL PANEL TEST            4
   RK05 CONTROL PANEL TEST #2         5
   HEAD ALIGNMENT ROUTINE             6
   POWER FAILURE (WRITE) TEST         7

   TYPE=1
   DRIVE NUMBERS ON SYTEM 1=0.

   IS THERE A SECOND SYSTEM?N
   MOUNT PACK ON DRIVE #0
   MAKE PACK WRITE ENABLE
   PRESS CONTINUE WHEN DRIVE
   DRIVE NOT READY
   DRIVE NOT READY                  ;CONTINUE PRESSED BUT
   DRIVE NOT READY                  ;DRIVE NOT READY. IF UP
   DRIVE NOT READY                  ;TO SPEED ETC. AND MESSAGE
   DRIVE NOT READY                  ;OCCURRING - STATIC SHOULD BE
```

```
   DRIVE NOT READY                  ;RUN IF NOT LOADED OR NOT
   DRIVE NOT READY                  ;READY MAKING DRIVE READY
   DRIVE NOT READY                  ;WILL STOP THE MESSAGE
   DRIVE NOT READY                  ;IT DOES NOT EFFECT THE
   DRIVE NOT READY
   DRIVE NOT READY                  ;THE OUTCOME OF COMPATABILITY
   DRIVE NOT READY
   DRIVE NOT READY
   DRIVE NOT READY
   DONE!
           RK11 UTILITY PACKAGE

           NAME                     TYPE
   INDEX                             0
   COMPATIBILITY PACKAGE             1
   OSCILLATING SEEK PACKAGE          2
```

ERROR EXAMPLE 3

```
           RK11 UTILITY PACKAGE

           NAME                     TYPE
   INDEX                             0
   COMPATIBILITY PACKAGE             1
   OSCILLATING SEEK PACKAGE          2
   FORMATTER-SURFACE VERIFIER        3
   RK05 CONTROL PANEL TEST           4
   RK05 CONTROL PANEL TEST #2        5
   HEAD ALIGNMENT ROUTINE            6
   POWER FAILURE (WRITE) TEST        7

   TYPE=1                           ;DRIVE RESET TIMED OUT
   DRIVE NUMBERS ON SYTEM 1=0,1,4,7.

   IS THERE A SECOND SYSTEM?Y
   DRIVE # =2
   MOUNT PACK ON DRIVE #0
   MAKE PACK WRITE ENABLE           ;THIS MESSAGE IF CONTINUOUS
   PRESS CONTINUE WHEN DRIVE RDY    ;INDICATED A DRIVE PROBLEM
   MOUNT PACK ON DRIVE #1           ;THERE IS NO RECOVERY
   MAKE PACK WRITE ENABLE           ;AND IF CONTINUOUS, A
   PRESS CONTINUE WHEN DRIVE RDY    ;LOAD START ADDRESS 200
   DRIVE RESET TIMED OUT            ;IS NECESSARY, DIAGNOSTIC
   DRIVE RESET TIMED OUT            ;SHOULD BE RUN AGAINST THE
   DRIVE RESET TIMED OUT            ;FAILING DRIVE.
   DRIVE RESET TIMED OUT
   DRIVE RESET TIMED OUT

   *NOTE A SLOW DRIVE OR FAST PROCESSOR AND MEMORY
   MAY CAUSE THE MESSAGE TO APPEAR A FEW TIMES AND
   THEN CONTINUE THIS IS OK AND WILL NOT EFFECT THE
   OUTCOME OF THE TEST.
```

ERROR EXAMPLE 4

```
   OSCILLATING SEEK PACKAGE         2
   FORMATTER-SURFACE VERIFIER       3
   RK05 CONTROL PANEL TEST          4

   TYPE=1
```

```
   DRIVE NUMBERS ON SYTEM 1=0.

   IS THERE A SECOND SYSTEM?N       ;SAME AS ABOVE BUT FUNCTION
   MOUNT PACK ON DRIVE #0           ;WAS A CONTROL RESET
   MAKE PACK WRITE ENABLE
   PRESS CONTINUE WHEN DRIVE RDY    ;ALL COMMENTS ARE THE SAME
   CONTROL RESET TIMED OUT          ;AS EXAMPLE 3
   CONTROL RESET TIMED OUT
   CONTROL RESET TIMED OUT
   CONTROL RESET TIMED OUT

   *A SINGULAR OCCURANCE AS ABOVE IS NOT A PROBLEM
   AND WILL NOT EFFECT COMPATABILITY

ERROR EXAMPLE 5

          THE BELOW ERRORS DO, ALWAYS, EFFECT COMPATABILITY.
          IN THE FIRST TYPE THE DRIVE IS DOWN
          INDICATING THAT (5) FIVE HARD OR SOFT
          ERRORS OCCURRED.  THE TEST WILL CONTINUE
          AGAINST THE OTHER DRIVES BUT THERE IS A
          PROBLEM IN THIS DRIVE AND IT SHOULD BE
          CONSIDERED NON EXISTENT AS FAR AS COMPATABILITY
          GOES.  THAT IS TO SAY IT IS NOT TESTED, THEREFORE
          NOT NECESSARILY COMPATABLE OR INCOMPATABLE.

          RK11 UTILITY PACKAGE

          NAME                     TYPE
   INDEX                            0
   COMPATIBILITY PACKAGE            1
   OSCILLATING SEEK PACKAGE         2
   FORMATTER-SURFACE VERIFIER       3
   RK05 CONTROL PANEL TEST          4
   RK05 CONTROL PANEL TEST #2       5
   HEAD ALIGNMENT ROUTINE           6
   POWER FAILURE (WRITE) TEST       7

   TYPE=1
   DRIVE NUMBERS ON SYTEM 1=0.

   IS THERE A SECOND SYSTEM?N
   MOUNT PACK ON DRIVE #0
   MAKE PACK WRITE ENABLE
   PRESS CONTINUE WHEN DRIVE RDY
   5 ERRORS OCCURRED DRIVE DECLARED DOWN!! NOT TESTED !
   DONE!
          RK11 UTILITY PACKAGE

          NAME                     TYPE
   INDEX                            0
   COMPATIBILITY PACKAGE            1

   *IN THE ABOVE CASE THE MESSAGE "3 SEEK INCOMPLETE
   ERRORS OCCURRED DRIVE DECLARED DOWN!! NOT TESTED!"
   MAY OCCUR IT IS THE SAME ERROR AS DESCRIBED ABOVE
   EXCEPT THAT IT IS CAUSED BY 3 SEEK ERRORS OCCURRING
   ON ONE DRIVE.

ERROR EXAMPLE 6
```

```
        RK11 UTILITY PACKAGE

          NAME                    TYPE
INDEX                              0
COMPATABILITY PACKAGE              1
OSCILLATING SEEK PACKAGE           2
FORMATTER-SURFACE VERIFIER         3
RK05 CONTROL PANEL TEST            4
RK05 CONTROL PANEL TEST ROUTINE    5
HEAD ALIGNMENT ROUTINE             6
POWER FAILURE (WRITE) TEST         7

TYPE=1
DRIVE NUMBERS ON SYSTEM 1=0.

IS THERE A SECOND SYSTEM?Y
DRIVE # =1
MOUNT PACK ON DRIVE #0
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
LOAD AND START ADDRESS 210 ON SYSTEM #2
AND TYPE THE BELOW WHEN ASKED FOR IT.
WORD            1=101000
WORD=000177

MOUNT PACK ON DRIVE #0
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=002764  EXPCTD=077400  RECVD=177000
  ADDR=002764  EXPCTD=077400  RECVD=077600
  ADDR=002764  EXPCTD=077400  RECVD=037600
  ADDR=002764  EXPCTD=077400  RECVD=037600
  ADDR=002764  EXPCTD=077400  RECVD=037600
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=007624  EXPCTD=077400  RECVD=177000
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=007633  EXPCTD=077400  RECVD=177000
  ADDR=007633  EXPCTD=077400  RECVD=177000
DONE!
        RK11 UTILITY PACKAGE

          NAME                    TYPE
INDEX                              0
COMPATABILITY PACKAGE              1
OSCILLATING SEEK PACKAGE           2
FORMATTER-SURFACE VERIFIER         3
RK05 CONTROL PANEL TEST            4
RK05 CONTROL PANEL TEST #2         5
HEAD ALIGNMENT ROUTINE             6
POWER FAILURE (WRITE) TEST         7

TYPE=
THE ABOVE ERROR MESSAGE SHOWS A COMPATABILITY
PROBLEM.  ALL ERRORS OCCURRED ON HEAD ONE OF
DRIVE 0 TRYING TO READ INFORMATION WRITTEN BY
DRIVE 1.
```

ERROR EXAMPLE 7

```
MOUNT PACK ON DRIVE #0
MAKKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=000367   EXPCTD=077400   RECVD=077600
  ADDR=000367   EXPCTD=077400   RECVD=037600
  ADDR=000367   EXPCTD=077400   RECVD=037600
  ADDR=000367   EXPCTD=077400   RECVD=037600
  ADDR=000367   EXPCTD=077400   RECVD=037600
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=002564   EXPCTD=077400   RECVD=077600
  ADDR=002564   EXPCTD=077400   RECVD=037600
  ADDR=002564   EXPCTD=077400   RECVD=037600
  ADDR=002564   EXPCTD=077400   RECVD=037600
  ADDR=002564   EXPCTD=077400   RECVD=037600
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=002764   EXPCTD=077400   RECVD=077600
  ADDR=002764   EXPCTD=077400   RECVD=037600
  ADDR=002764   EXPCTD=077400   RECVD=037600
  ADDR=002764   EXPCTD=077400   RECVD=037600
  ADDR=002764   EXPCTD=077400   RECVD=037600
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=002767   EXPCTD=077400   RECVD=177000
5 ERRORS OCCURRED DRIVE DECLARED DOWN!! NOT TESTED!
DONE!
```

IN THE ABOVE EXAMPLE THE PROBLEM IS EXTREME. THE DRIVE WAS DECLARED DOWN DO TO CHECKSUM ERRORS. (TO SEE HOW THIS WAS DETERMINED SEE PARAGRAPH 9.7). NOTICE ALSO THE PROBLEM DID NOT START APPEARING UNTIL CYLINDER 7, AND WAS NOT FATAL UNTIL CYLINDER 57, AGAIN HEAD #1 WAS A COMMON FACTOR.

****

9.10 COMPATIBILITY ERROR RECOVERY

ALTHOUGH A UTILITY PACKAGE IS NOT A TRUE DIAGNOSTIC IT IS OF BENEFIT TO THE USER TO AT TIMES, BE ABLE TO MODIFY THE PROGRAM TO RECIEVE MORE INFORMATION OR CONTROL PARAMETERS

1. THERE ARE TWO STRATEGICALY PLACED NO-OPS, WHICH IF CHANGED TO HALTS, MAY BE OF HELP TO THE USER. ONE IS IN THE 'EXECUTE' ROUTINE WHICH ALLOWS THE USER TO EXAMINE THE DISK ADDRESS, BUS ADDRESS, WORD COUNT AND CONTROL REGISTERS IN TEMPORARY LOCATIONS JUST PRIOR TO LOADING AND EXECUTION. THE SECOND IS IN THE 'ERRCHK' ROUTINE WHICH ALLOW THE USER TO EXAMINE THE RKER REGISTER BEFORE THE PROGRAM CORRECTS ANY ERRORS WHICH WHICH MAY HAVE OCCURRED.

2. IF PLAGED BY CHECKSUM ERRORS AND THE USER WISHES MORE ERROR MAPING THEN HE MAY MODIFY THE MASK WORD AT LOCATION 'ERRCHK+2' TO ONLY RECOGNIZE HARD ERRORS.

3. TO INCREASE OR DECREASE THE NUMBER OF RETRYS ALLOWED

BEFORE A DRIVE IS DECLARED DOWN, GO TO THE 'MOUNT' ROUTINE, MODIFY THE SETUP OF LOCATIONS 'ECNT' AND 'CNTSIN' AND YOU HAVE IT!

4. IF THE USER DECIDES, SAY BECAUSE OF A LARGE NUMBER OF FAILURES, TO ALTER THE NUMBER OF PRINTOUTS PER SECTOR ON FAILURES (THE TYPE IN ERROR EXAMPLE 6 AND 7) HE MAY MODIFY THE SETUP OF 'CHKCNT' IN THE 'RDCHK' ROUTINE.

A FINAL LOOK; THE FOLLOWING SECTION SHOWS ALL PACKAGES CALLED IN SEQUENCE, NONE WITH ERRORS.

```
        RK11 UTILITY PACKAGE


          NAME                    TYPE
INDEX                              0
COMPATABILITY PACKAGE              1
OSCILLATING SEEK PACKAGE           2
FORMATTER-SURFACE VERIFIER         3
RK05 CONTROL PANEL TEST            4
RK05 CONTROL PANEL TEST #2         5
HEAD ALIGNMENT ROUTINE             6
POWER FAILURE (WRITE) TEST         7

TYPE=0

        RK11 UTILITY PACKAGE

          NAME                    TYPE
INDEX                              0
COMPATABILITY PACKAGE              1
OSCILLATING SEEK PACKAGE           2
FORMATTER-SURFACE VERIFIER         3
RK05 CONTROL PANEL TEST            4
RK05 CONTROL PANEL TEST #2         5
HEAD ALIGNMENT ROUTINE             6
POWER FAILURE (WRITE) TEST         7

TYPE=1
DRIVE NUMBERS ON SYSTEM 1=0,1,3.

IS THERE A SECOND SYSTEM?N
MOUNT PACK ON DRIVE #0
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
MOUNT PACK ON DRIVE #1
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
MOUNT PACK ON DRIVE #3
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
MOUNT PACK ON DRIVE #0
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
MOUNT PACK ON DRIVE #1
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
MOUNT PACK ON DRIVE #3
```

```
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
DONE!

        RK11 UTILITY PACKAGE

          NAME                    TYPE
INDEX                              0
COMPATABILITY PACKAGE              1
OSCILLATING SEEK PACKAGE           2
FORMATTER-SURFACE VERIFIER         3
RK05 CONTROL PANEL TEST            4
RK05 CONTROL PANEL TEST #2         5
HEAD ALIGNMENT ROUTINE             6
POWER FAILURE (WRITE) TEST         7

TYPE=2
OSCILLATING SEEK PACKAGE, WHICH DRIVE?0
TOGGLE THE "FIRST CYLINDER ADDRESS" (OUTER LIMIT)
INTO THE LOW BYTE (BIT0-7) OF THE SWITCH REGISTER AND THE "LAST
CYLINDER ADDRESS" (INNER LIMIT) INTO THE HIGH
BYTE (BIT8-15), THEN PRESS CONTINUE.

        RK11 UTILITY PACKAGE

          NAME                    TYPE
INDEX                              0
COMPATABILITY PACKAGE              1
OSCILLATING SEEK PACKAGE           2
FORMATTER-SURFACE VERIFIER         3
RK05 CONTROL PANEL TEST            4
RK05 CONTROL PANEL TEST #2         5
HEAD ALIGNMENT ROUTINE             6
POWER FAILURE (WRITE) TEST         7

TYPE=3
FORMATTER-SURFACE VERIFIER, WHICH DRIVE?0

PACK GOOD.

        RK11 UTILITY PACKAGE

          NAME                    TYPE
INDEX                              0
COMPATABILITY PACKAGE              1
OSCILLATING SEEK PACKAGE           2
FORMATTER-SURFACE VERIFIER         3
RK05 CONTROL PANEL TEST            4
RK05 CONTROL PANEL TEST #2         5
HEAD ALIGNMENT ROUTINE             6
POWER FAILURE (WRITE) TEST         7

TYPE=4
RK05 CONTROL PANEL TEST, WHICH DRIVE?0
MOUNT PACK ON DRIVE #0
PLACE DRIVE IN RUN ;SHOULD SEE THE RUN,
POWER, AND ON CYLINDER LAMPS LIGHT.
MAKE DRIVE WRITE ENABLE PRESS CONTINUE

WRITE PROTECT THE DRIVE THEN PRESS CONTINUE
```

```
CLEAR WRITE PROTECT THEN PRESS CONTINUE

CAUTION! TRY TO OPEN THE DOOR, DO NOT FORCE:
DOOR SHOULD NOT OPEN!
PRESS CONTINUE WHEN FINISHED

PUT DRIVE IN LOAD, WAIT FOR LOAD LIGHT
PRESS CONTINUE WHEN FINISHED

OPEN THE DOOR, PUT DRIVE IN RUN
CAUTION! IF RUN LIGHT ON ERROR! DEPRESS
LOAD IMMEDIATELY, CONTINUE WHEN FINISHED

REMOVE THE PACK, CLOSE THE DOOR
PUT DRIVE IN RUN, DRIVE SHOULD NOT
RUN...INTERLOCKS HAVE BEEN CHECKED
DONE!
```

RK11 UTILITY PACKAGE

```
                                .TITLE  MAINDEC-11-DZRKI-D
                                ;*COPYRIGHT (C) 1974,1976
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY BOB COLLINS
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
                                ;*
000001                          $TN=1
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT


                                ;*REVISED BY JIM KAPADIA
                                ;*REVISED BY TOM SAWYER FEB 27, 1976
                                ;*REVISED BY CHUCK HESS AUGUST, 1976

                                .SBTTL  BASIC DEFINITIONS

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

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

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

                                ;*PRIORITY LEVEL DEFINITIONS
000000                          PR0=    0               ;;PRIORITY LEVEL 0
000040                          PR1=    40              ;;PRIORITY LEVEL 1
000100                          PR2=    100             ;;PRIORITY LEVEL 2
000140                          PR3=    140             ;;PRIORITY LEVEL 3
000200                          PR4=    200             ;;PRIORITY LEVEL 4
000240                          PR5=    240             ;;PRIORITY LEVEL 5
```



```
000300                          PR6=    300             ;;PRIORITY LEVEL 6
000340                          PR7=    340             ;;PRIORITY LEVEL 7

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
002000                          SW10=   2000
001000                          SW09=   1000
000400                          SW08=   400
000200                          SW07=   200
000100                          SW06=   100
000040                          SW05=   40
000020                          SW04=   20
000010                          SW03=   10
000004                          SW02=   4
000002                          SW01=   2
000001                          SW00=   1
                                .EQUIV  SW09,SW9
                                .EQUIV  SW08,SW8
                                .EQUIV  SW07,SW7
                                .EQUIV  SW06,SW6
                                .EQUIV  SW05,SW5
                                .EQUIV  SW04,SW4
                                .EQUIV  SW03,SW3
                                .EQUIV  SW02,SW2
                                .EQUIV  SW01,SW1
                                .EQUIV  SW00,SW0

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

```
                                .EQUIV  BIT01,BIT1
                                .EQUIV  BIT00,BIT0

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

            000000                      .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
            000174                      .=174
000174  000000                 DISPREG: .WORD  0               ;;SOFTWARE DISPLAY REGISTER
000176  000000                 SWREG:   .WORD  0               ;;SOFTWARE SWITCH REGISTER
                                .SBTTL  STARTING ADDRESS(ES)
000200  000137  001434                  JMP     @#STARTR        ;;JUMP TO STARTING ADDRESS OF PROGRAM
            000210                      .=210
000210  112737  000377  001312          MOVB    #377,@#MODE
000216  000137  001440                  JMP     @#START
                                .SBTTL  ACT11 HOOKS

                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
                                        $SVPC=.                 ;SAVE PC
            000046                      .=46
000046  001400                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
            000052                      .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
            000222                      .=@SVPC                 ;; RESTORE PC
```



```
                                .SBTTL  COMMON TAGS

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

            001100                      .=1100
001100  001100                 $CMTAG:                         ;;START OF COMMON TAGS
001100  000000                 $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102     000                  $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103     000                  $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000                 $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                 $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000                 $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                 $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114     000                  $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115     001                  $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                 $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                 $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000                 $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000                 $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000                 $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                         .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                         .WORD   0
001134     000                  $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135     000                  $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                         .WORD   0
001140  177570                 SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570                 DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560                 $TKS:   177560                  ;;TTY KBD STATUS
001146  177562                 $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564                 $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566                 $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155     002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157     000                  $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160     077                  $QUES:  .ASCII  /?/             ;;QUESTION MARK
001161     015                  $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001162  000012                 $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
001164  000000                 DRACTV: .WORD   0               ;ACTIVE DRIVE WORD

001166  000010                 LOGA:   .BLKW   10              ;TABLE OF ACTIVE DRIVE WORDS

001206  152525                 DRV0:   .WORD   152525          ;TABLE OF PATTERN = TO DRIVE #'S
001210  077777                         .WORD   077777
001212  000000                         .WORD   000000
001214  012345                         .WORD   012345
001216  125252                         .WORD   125252
001220  000001                         .WORD   000001
001222  177777                         .WORD   177777
001224  154320                         .WORD   154320

001226  000010                 RDTBL:  .BLKW   10              ;TABLE OF READ ADDRESS
001246  000004                 PASTBL: .BLKW   4               ;TABLE OF PARAMETERS FOR SYSTEM #2
```

```
     001256     377                MSKTBL: .BYTE   377            ;TABLE OF CYLINDER BASE FOR AUTO MODE
     001257     177                        .BYTE   177
     001260     077                        .BYTE   077
     001261     037                        .BYTE   037
     001262     017                        .BYTE   017
     001263     007                        .BYTE   007
     001264     003                        .BYTE   003
     001265     001                        .BYTE   001

     001266     000                BASE:   .BYTE   0              ;CYL 0  BASE CYLINDER ADDRESS
     001267     050                        .BYTE   50             ;CYL 40 BASE CYLINDER ADDRESS
     001270     120                        .BYTE   120            ;CYL 80 BASE CYLINDER ADDRESS
     001271     170                        .BYTE   170            ;CYL 120 BASE CYLINDER ADDRESS
     001272     240                        .BYTE   240            ;CYL 160 BASE CYLINDER ADDRESS
     001273     303                        .BYTE   303            ;CYL 195 BASE CYLINDER ADDRESS

     001274  000011                CYLTBL: .BLKB   11             ;TABLE OF SELECTED BASES

     001305     000                SECTBL: .BYTE   0              ;SECTOR 0
     001306     004                        .BYTE   4              ;SECTOR 4
     001307     007                        .BYTE   7              ;SECTOR 7
     001310     013                        .BYTE   13             ;SECTOR 12

     001311     000                DRCNT1: .BYTE   0              ;COUNT OF NUMBER OF DRIVES ON SYS. 1
     001312     000                MODE:   .BYTE   0              ;IF -1 START 210 SELECTED
     001313     000                PRONUM: .BYTE   0              ;IF 0 1 PROCESSOR SELECTED
     001314     000                DRIVE:  .BYTE   0              ;DRIVE # UNDER TEST (MAN+AUTO MODE)
     001315     000                CYLBAS: .BYTE   0              ;BASE SELECTED (MANUAL MODE)
     001316     000                COMND:  .BYTE   0              ;IF 0 WRITE COMMAND
     001317     000                WRTNBY: .BYTE   0              ;DRIVE WHICH DID WRITE (READ OPERATION)
     001320     000                HDRFLG: .BYTE   0              ;FLAG FOR ONE HEADER PRINTOUT
     001321     000                ECNT:   .BYTE   0              ;ERROR COUNTER
     001322     000                CNTSIN: .BYTE   0              ;SEEK INCOM. COUNTER
     001323     000                TIMR2:  .BYTE   0              ;SECOND PASS TIMER
     001324     000                IDEX:   .BYTE   0              ;CURRENT INDEX #
     001325     000                STFLG:  .BYTE   0
     001326     000                OSPFLG: .BYTE   0

             001330                        .EVEN

     001330  000000                KYTEMP: .WORD   0              ;TEMP. KEYBOARD BUFFER
     001332  000000                CONTRL: .WORD   0              ;TEMP. CONTROL+STATUS WORD
     001334  000000                DSKADR: .WORD   0              ;TEMP. DISK ADDRESS WORD
     001336  000000                BUSADR: .WORD   0              ;TEMP. BUS ADDRESS WORD
     001340  000000                WRDCNT: .WORD   0              ;TEMP. WORD COUNT
     001342  172000                CYLCNT: .WORD   -6000          ;WORD COUNT OF 1 CYLINDER
     001344  177400                SECCNT: .WORD   -400           ;WORD COUNT OF 1 SECTOR
     001346  000000                TIMR:   .WORD   0              ;TIMER FOR OPERATIONS
     001350  000000                CHKCNT: .WORD   0              ;NUMBER OF ERROR PRINTOUTS
     001352  000000                DSKTMP: .WORD   0              ;SAVE OF CURRENT DISK #
     001354  004003                WRITCS: .WORD   4003           ;IBA+WRITE+GO
     001356  000005                READCS: .WORD   5              ;READ+GO
     001360  000000                ERRFLG: .WORD   0              ;ERROR FLAG INHIBIT ADDRESS CHANGE
     001362  000000                PATTRN: .WORD   0              ;DATA PATTERN
     001364  177400                RKDS:   .WORD   177400
```



```
     001366  177402                RKER:   .WORD   177402
     001370  177404                RKCS:   .WORD   177404
     001372  177406                RKWC:   .WORD   177406
     001374  177410                RKBA:   .WORD   177410
     001376  177412                RKDA:   .WORD   177412
     001400  000000                SENDAD: .WORD   0
     001402  000000                SEEKI:  .WORD   0
     001404  000000                SEEKO:  .WORD   0

             105212                LFLF=   105212
     001406  013652                BA:     BUFF
     001410  000000                DA:     .WORD   0
     001412  000000                WC:     .WORD   0
     001414  013654                RBA:    RBUFF
     001416  000000                RWC:    .WORD   0
     001420  000000                EXTR:   .WORD   0
     001422  000000                ERRWF:  .WORD   0
     001424  000000                ERRRF:  .WORD   0
     001426  000000                ERRRFC: .WORD   0
     001430  000000                ERRWCH: .WORD   0
     001432  000000                ERRWCS: .WORD   0

                                   ;BIT DEFINITIONS

             010000                DPL=BIT12
             000100                RWS=BIT6
             000040                WPS=BIT5
             001000                SIN=BIT9
```

```
                        .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


001434                  $ERRTB:
                        ;;*****************************************************************

                        ;THE ERROR TABLE IS UNUSED IN THIS PROGRAM

                        ;;*****************************************************************
```



```
001434  105037  001312          START:  CLRB    @#MODE
001440  000005                  START:  RESET                   ;CLEAR THE BUS
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001442  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001446  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001450  022706  001140                  CMP     #SWR,R6 ;;DONE?
001454  001374                          BNE     .-6             ;;LOOP BACK IF NO
001456  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001462  012737  024010  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001470  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001476  012737  024070  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001504  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER, IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001512  013746  000004          MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001516  012737  001552  000004  MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001524  012737  177570  001140  MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001532  012737  177570  001142  MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001540  022777  177777  177372  CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001546  001012                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
001550  000403                  BR      65$             ;;BRANCH IF NO TIMEOUT
001552  012716  001560  64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001556  000002                  RTI
001560  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001566  012737  000174  001142          MOV     #DISPREG,DISPLAY
001574  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001600  004737  022526                  JSR     PC,$TKINT       ;INITIALIZE THE TTY INTERRUPT HANDLER
                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
001604  005227  177777                  INC     #-1             ;;FIRST TIME?
001610  001045                          BNE     67$             ;;BRANCH IF NO
001612  104401  001650                  TYPE    ,68$            ;;TYPE ASCIZ STRING
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
001616  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
001622  001006                          BNE     69$             ;;BRANCH IF YES
001624  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
001632  001005                          BNE     70$             ;;BRANCH IF NO
001634  104405                          GTSWR                   ;;GET SOFT-SWR SETTINGS
001636  000403                          BR      70$
001640  112737  000001  001134  69$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
001646                          70$:
001646  000426                          BR      67$             ;;GET OVER THE ASCIZ
                                ;;68$:  .ASCIZ  <CRLF>/RK11 UTILITY PACKAGE/<15><12>/MAINDEC-11-DZRKI-D/<CRLF>
001724                          67$:
001724  105737  001312                  TSTB    @#MODE
001730  100002                          BPL     1$
001732  000137  003432                  JMP     @#SECOND
001736  105737  001325          1$:     TSTB    STFLG                   ;PRINT ONLY THE FIRST TIME
001742  001402                          BEQ     10$
001744  000137  002534                  JMP     TABLTY
001750  105237  001325          10$:    INCB    STFLG
001754  105037  001326          STRT1:  CLRB    OSPFLG
```

```
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12><15><12>/       NAME                    TYPE/
                                64$:
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <15><12>/INDEX                          0/
                                66$:
                                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                                        BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <15><12>/COMPATIBILITY PACKAGE          1/
                                68$:
                                        TYPE    ,71$            ;;TYPE ASCIZ STRING
                                        BR      70$             ;;GET OVER THE ASCIZ
                                ;;71$:  .ASCIZ  <15><12>/OSCILLATING SEEK PACKAGE       2/
                                70$:
                                        TYPE    ,73$            ;;TYPE ASCIZ STRING
                                        BR      72$             ;;GET OVER THE ASCIZ
                                ;;73$:  .ASCIZ  <15><12>/FORMATTER-SURFACE VERIFIER     3/
                                72$:
                                        TYPE    ,75$            ;;TYPE ASCIZ STRING
                                        BR      74$             ;;GET OVER THE ASCIZ
                                ;;75$:  .ASCIZ  <15><12>/RK05 CONTROL PANEL TEST        4/
                                74$:
                                        TYPE    ,77$            ;;TYPE ASCIZ STRING
                                        BR      76$             ;;GET OVER THE ASCIZ
                                ;;77$:  .ASCIZ  <15><12>/RK05 CONTROL PANEL TEST #2     5/
                                76$:
                                        TYPE    ,79$            ;;TYPE ASCIZ STRING
                                        BR      78$             ;;GET OVER THE ASCIZ
                                ;;79$:  .ASCIZ  <15><12>/HEAD ALIGNMENT ROUTINE         6/
                                78$:
                                        TYPE    ,81$            ;;TYPE ASCIZ STRING
                                        BR      80$             ;;GET OVER THE ASCIZ
                                ;;81$:  .ASCIZ  <15><12>/POWER FAILURE (WRITE) TEST     7/
                                80$:
                                TABLTY:
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12><15><12>/TYPE=/
                                64$:
                                        RDOCT                   ;GET THE TEST NUMBER FROM THE OPERATOR
                                        MOV     (SP)+,R0        ;STORE IT IN R0
                                        CMP     #10,R0          ;VALID NUMBER ?
                                        BLE     NG              ;BR IF NOT
                                        ROL     R0              ;ALIGN THE NUMBER FOR DISPATCHING
                                        JMP     @BEGIN(R0)      ;GO TO THE SELECTED TEST
                                NG:     TYPE    ,SQUES          ;"?"
                                        BR      TABLTY

                                ;TEST ENTRY DISPATCH TABLE

                                BEGIN:  STRT1
                                        SECT.3
                                        SECT.2
                                        SECT.1
```



```
                                        SECT.0
                                        SECT.4
                                        SECT.5
                                        SECT.6


                                        .SBTTL  COMPATIBILITY TEST

                                ;ROUTINE TO PICK UP THE DRIVE NUMBER TO BE TESTED
                                ;ON SYSTEM 1.

                                SECT.3: NOP                     ;NO-OP
                                AUTSL2:
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/TERMINATE WITH ',<CR>"/
                                64$:
                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                        BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <15><12>/DRIVE NUMBERS ON SYSTEM 1=/
                                66$:
                                        RDLIN
                                        MOV     (SP)+,R0        ;PICK UP THE ADDRESS OF THE INPUT BUFFER
                                        MOV     #LOGA,R1        ;GET THE ADDRESS OF THE LOGICAL UNIT TBL.
                                        CLRB    @#DRCNT1        ;CLEAR THE DRIVE COUNTER
                                1$:     CLR     @#KYTEMP        ;CLEAR TEMP
                                        MOVB    (R0)+,@#KYTEMP  ;GET THE FIRST DRIVE #
                                        CMPB    #54,@#KYTEMP    ;IS IT A COMMA THAT WAS TYPED?
                                        BEQ     1$              ;IF YES GO BACK
                                        SUB     #60,@#KYTEMP    ;MAKE ASCII A DRIVE #
                                        BMI     2$              ;IF RESULT NEGATIVE BRANCH
                                        JSR     PC,STORE        ;IF RESULT POSITIVE JUMP
                                        BR      1$              ;AFTER STORING GET NEXT #
                                2$:     CMPB    #56,-(R0)       ;WAS NEGATIVE RESULT A TERMINATOR?
                                        BEQ     3$              ;IF YES BRANCH
                                        JSR     PC,ILEGAL       ;IF NO BAD CHARACTER JUMP
                                3$:     CMP     #DRV0,R1        ;IS THE TABLE FULL
                                        BEQ     SECSYS          ;IF YES BRANCH
                                        MOV     #100000,(R1)+   ;IF NO FILL TABLE WITH DOWN INDICATOR.
                                        BR      3$              ;GO BACK AND CHECK

                                ;ROUTINE TO DETERMINE IF THERE IS A SECOND
                                ;SYSTEM AND IF SO TO GET THE NUMBER OF THE
                                ;DRIVE ON THIS SYSTEM

                                SECSYS:
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/IS THERE A SECOND SYSTEM?/
                                64$:
                                        NOP                     ;***
                                        RDCHR                   ;READ A CHARACTER
                                        MOV     (SP)+,@#KYTEMP  ;GET THE RESPONSE
                                        TYPE    ,KYTEMP         ;ECHO
                                        CMP     #131,@#KYTEMP   ;WAS IT A "Y" (FOR YES)?
                                        BEQ     PRO2            ;IF YES BRANCH TO PROCESSOR 2
```

```
   482  003116  022737  000116  001330          CMP     #116,@#KYTEMP   ;WAS RESPONSE LEGAL (N FOR NO)?
   483  003124  001460                          BEQ     PRO1            ;IF LEGAL BRANCH
   484  003126  104401  003134                  TYPE    ,67$            ;;TYPE ASCIZ STRING
   485  003132  000401                          BR      66$             ;;GET OVER THE ASCIZ
   486                                  ;;67$:  .ASCIZ  /?/
   487  003136                               66$:
   488  003136  000734                          BR      SECSYS          ;GO BACK ASK AGAIN
   489  003140  152737  000377  001313  PRO2:   BISB    #377,@#PRONUM   ;SET FLAG TWO PROCESSORS
   490  003146  000240                          NOP                     ;***
   491  003150  104401  003156                  TYPE    ,65$            ;;TYPE ASCIZ STRING
   492  003154  000406                          BR      64$             ;;GET OVER THE ASCIZ
   493                                  ;;65$:  .ASCIZ  <15><12>/DRIVE # =/
   494  003172                               64$:
   495  003172  104407                          RDCHR                   ;READ A CHARACTER
   496  003174  012637  001330                  MOV     (SP)+,@#KYTEMP  ;PICK UP THE RESPONSE
   497  003200  104401  001330                  TYPE    ,KYTEMP         ;ECHO
   498  003204  162737  000060  001330          SUB     #60,@#KYTEMP    ;MAKE IT A NUMBER
   499  003212  100420                          BMI     BADINP          ;IF NOT A NUMBER, BRANCH
   500  003214  022737  000010  001330          CMP     #10,@#KYTEMP    ;IS IT A LEGAL #?
   501  003222  003414                          BLE     BADINP          ;IF NO BRANCH
   502  003224  000337  001330                  SWAB    @#KYTEMP        ;GET THE DRIVE # TO THE HIGH BYTE
   503  003230  052737  040000  001330          BIS     #BIT14,@#KYTEMP ;SET THE SECOND SYSTEM BIT
   504  003236  113705  001311                  MOVB    @#DRCNT1,R5     ;GET THE DRIVE COUNT
   505  003242  006105                          ROL     R5              ;MAKE IT AN INDEX
   506  003244  013765  001330  001166          MOV     @#KYTEMP,LOGA(R5) ;STORE THE SYSTEM #2 WORD
   507  003252  000407                          BR      GO              ;GO DO THE TEST
   508  003254                               BADINP:
   509  003254  104401  003262                  TYPE    ,65$            ;;TYPE ASCIZ STRING
   510  003260  000401                          BR      64$             ;;GET OVER THE ASCIZ
   511                                  ;;65$:  .ASCIZ  /?/
   512  003264                               64$:
   513  003264  000725                          BR      PRO2            ;GO BACK ASK AGAIN
   514  003266  105037  001313          PRO1:   CLRB    @#PRONUM        ;CLEAR THE FLAG ONE PROCESSOR
   515
   516                                  ;THIS IS THE ACTUAL PROGRAM
   517
   518  003272  012700  001166          GO:     MOV     #LOGA,R0        ;GET THE TABLE ADDRESS TO R0
   519  003276  105037  001324                  CLRB    @#IDEX          ;CLRB THE INDEX
   520  003302  000405                          BR      GO2             ;BRANCH AROUND INCREMENT ROUTINE
   521
   522  003304  062700  000002          GO1:    ADD     #2,R0           ;INDEX THRU THE TABLE
   523  003310  022700  001206                  CMP     #DRV0,R0        ;DONE?
   524  003314  001414                          BEQ     EXIT            ;IF YES GET OUT
   525  003316  005710          GO2:    TST     (R0)            ;IS THE DRIVE ACTIVE
   526  003320  100001                          BPL     GO3             ;IF YES BRANCH
   527  003322  000770                          BR      GO1             ;NO TRY THE NEXT ONE
   528  003324  011037  001164          GO3:    MOV     (R0),@#DRACTV   ;PICK UP THE ACTIVE DRIVE WORD
   529  003330  004737  004154                  JSR     PC,CYCLE        ;CALL CYCLE (PICK UP DRIVE #, CYL BASE, AND CALL MOUNT)
   530  003334  004737  005064                  JSR     PC,WRLINK       ;CALL WRITE LINK TO LOAD REGISTERS FOR WRITE
   531  003340  004737  005612                  JSR     PC,RDLINK       ;CALL READ LINK TO LOAD REGISTERS FOR READ
   532  003344  000757                          BR      GO1             ;GO GET NEXT DRIVE
   533  003346  012700  001166          EXIT:   MOV     #LOGA,R0
   534  003352  022700  001206          EXTFR2: CMP     #DRV0,R0
   535  003356  001414                          BEQ     EXITX
   536  003360  032710  040000                  BIT     #BIT14,(R0)
   537  003364  001011                          BNE     EXITX
```



```
   538  003366  005720                          TST     (R0)+
   539  003370  100770                          BMI     EXTFR2
   540  003372  014001                          MOV     -(R0),R1
   541  003374  004737  004230                  JSR     PC,DO2
   542  003400  004737  005612                  JSR     PC,RDLINK
   543  003404  005720                          TST     (R0)+
   544  003406  000761                          BR      EXTFR2
   545  003410                               EXITX:
   546  003410  104401  003416                  TYPE    ,65$            ;;TYPE ASCIZ STRING
   547  003414  000404                          BR      64$             ;;GET OVER THE ASCIZ
   548                                  ;;65$:  .ASCIZ  <15><12>/DONE!/
   549  003426                               64$:
   550  003426  000137  001440                  JMP     @#START         ;RESTART
   551
   552                                  ;THIS IS PROCESSOR #2 CODE. THIS ROUTINE ASKS FOR AND UNPACKS THE CONTROL
   553                                  ;WORDS FROM THE FIRST PROCESSOR.
   554
   555  003432                          SECOND:
   556  003432  104401  003440                  TYPE    ,65$            ;;TYPE ASCIZ STRING
   557  003436  000415                          BR      64$             ;;GET OVER THE ASCIZ
   558                                  ;;65$:  .ASCIZ  <15><12>/COMPATIBILITY-SYSTEM#2/
   559  003472                               64$:
   560  003472  012704  000200                  MOV     #200,R4         ;SET UP MASK BIT IN R4
   561  003476  012703  000001                  MOV     #1,R3           ;SET UP WORD COUNTER IN R3
   562  003502  012702  001166                  MOV     #LOGA,R2        ;GET THE TABLE ADDRESS
   563  003506                          INSYS2:
   564  003506  104401  003514                  TYPE    ,65$            ;;TYPE ASCIZ STRING
   565  003512  000405                          BR      64$             ;;GET OVER THE ASCIZ
   566                                  ;;65$:  .ASCIZ  <15><12>/WORD  /
   567  003526                               64$:
   568  003526  000240                          NOP                     ;***
   569  003530  010346                          MOV     R3,-(SP)        ;GET READY TO TYPE WORD COUNTER
   570  003532  104403                          TYPOS
   571  003534     006                          .BYTE   6
   572  003535     000                          .BYTE   0
   573  003536  104401  003544                  TYPE    ,67$            ;;TYPE ASCIZ STRING
   574  003542  000401                          BR      66$             ;;GET OVER THE ASCIZ
   575                                  ;;67$:  .ASCIZ  /=/
   576  003546                               66$:
   577  003546  104411                          RDOCT
   578  003550  012600                          MOV     (SP)+,R0        ;PICK UP THE OCTAL WORD
   579  003552  110001                          MOVB    R0,R1           ;GET THE FIRST DRIVE TO R1
   580  003554  000300                          SWAB    R0              ;GET THE SECOND DRIVE TO R0 LOW BYTE
   581  003556  042700  177400                  BIC     #177400,R0      ;CLEAR THE UNUSED BITS
   582  003562  042701  177400                  BIC     #177400,R1      ;CLEAR THE UNUSED BITS
   583  003566  006000                          ROR     R0              ;ROTATE RIGHT R0
   584  003570  103003                          BCC     2$              ;IF CARRY IS CLEAR BRANCH
   585  003572  052700  000200                  BIS     #BIT7,R0        ;SET DOWN BIT IF CARRY SET
   586  003576  000241                          CLC                     ;AND CLEAR THE CARRY BIT
   587  003600  006001                   2$:    ROR     R1              ;NOW DO THE SAME FOR R1
   588  003602  103002                          BCC     3$              ;IF NO ERROR, BRANCH
   589  003604  052701  000200                  BIS     #BIT7,R1        ;IF ERROR SET THE BIT
   590  003610  110162  000001           3$:    MOVB    R1,1(R2)        ;SET DRIVE FIRST IN TABLE
   591  003614  004737  004010                  JSR     PC,MASKER       ;CALL THE MASK CONTROL SUBROUTINE
   592  003620  110062  000001                  MOVB    R0,1(R2)        ;SET DRIVE SECOND IN TABLE (NEXT WORD)
   593  003624  004737  004010                  JSR     PC,MASKER       ;CALL THE MASK CONTROL SUBROUTINE
```

```
                                ;INCREMENT THE WORD COUNTER
                                ;GO BACK AND GET NEXT WORD
EXITA:                          ;FILL THE TABLE (LAST WORD)
                                ;WITH ALL BITS SET
                                ;GO BACK IF NOT DONE
EXITB:                          ;CLEAR DOWN AND SECOND SYSTEM BITS AT TABLE
                                ;GET ADDRESS TO R0 (CURRENT TABLE)
                                ;ADD 2 TO THE POINTER
FIL.DN:                         ;TABLE FULL?
                                ;IF YES BRANCH
                                ;FILL THE TABLE
                                ;GO BACK TRY AGAIN
2$:                             ;GET THE WORD TO R1
                                ;
                                ;
                                ;FORM DRIVE # FOR MOUNT
                                ;WRITE NEW INFO
                                ;READ SAMPLE (ALL DRIVES)
                                ;;TYPE ASCIZ STRING
                                ;;GET OVER THE ASCIZ
;;65$:                          <15><12>/DONE SYSTEM 2 RESTART SYSTEM 1, TYPE WORD /
64$:
                                ;GET WORD FOR SYSTEM 1 AND TYPE
TYPOS
                                ;
                                ;
1$:                             ;HALT (FINISHED SYSTEM #2)
                                ;GO BACK TO HALT

;THIS ROUTINE SETS UP THE MASK BITS IN THE LOG TABLE FOR
;SYSTEM #2, IF A DRIVE IS DOWN NO BIT IS SET BUT THE MASK
;IS SHIFTED.

MASKER:                         ;IS THE DRIVE UP
                                ;IF NO, BRANCH
                                ;MOVE THE MASK BIT IN THE TABLE
RETRN4:                         ;ROTATE THE MASK
                                ;DONE?,IF NO,BRANCH
                                ;SET UP FOR RETURN
                                ;
1$:                             ;IS THIS SYSTEM # 2'S DRIVE?
                                ;IF NO,BRANCH
                                ;SET UP FOR RETURN
                                ;
2$:                             ;INDEX THRU LOGA TABLE
                                ;RETURN

;ROUTINE TO BUILD THE ACTIVE LOGICAL UNIT TABLE

STORE:                          ;IS INPUT A LEGAL NUMBER?
                                ;***
                                ;IF NOT BRANCH
                                ;ALIGN DRIVE # FOR TABLE
                                ;PUT THE WORD IN THE TABLE
                                ;CLEAR THE TEMP WORD
                                ;INCREMENT THE COUNTER
```



```
                                ;IS THE TABLE FULL?
                                ;IF YES BRANCH
                                ;IF NO RETURN
TBLFUL:                         ;IF TABLE FULL SET UP FOR SYSTEM #2
                                ;RETURN
ILEGAL:                         ;IF ILLEGAL RESPONSE, GO BACK
                                ;;TYPE ASCIZ STRING
                                ;;GET OVER THE ASCIZ
;;65$:                         /ILLEGAL INPUT/
64$:
                                ;RETURN

                                ;THIS ROUTINE GETS A DRIVE # AND BUILDS A TABLE OF
                                ;CYLINDER ADDRESS FOR USE BY WRITE, (R0)=ADDRESS OF ACTIVE
                                ;WORD IN LOGICAL TABLE, IT ALSO SETS AND CLEARS THE MASK BITS
                                ;OF THE TABLE AS OPERATIONS INDICATE

CYCLE:                          ;GET THE LOGICAL UNIT ACTIVE WORD TO R1
                                ;IS IT ON SYSTEM #2
                                ;IF NO BRANCH
                                ;GO TO SYSTEM #2
CYCL2:                          ;GET THE INDEX VALUE
                                ;GET THE MASK TO R2
CYCLE2:                         ;CALL THE MASK SUBROUTINE
LDFLG:                          ;GET TABLE ADDRESS TO R3
2$:                             ;IS ACTIVE ADDRESS=TO FIRST ADDRESS
                                ;IF NO BRANCH
                                ;IF YES SET BITS TO SHOW WRITE
                                ;BRANCH TO SEE IF DONE
3$:                             ;CLEAR BITS TO SHOW OVER-WRITE
4$:                             ;DONE
                                ;IF NO GO BACK
DO2:                            ;GET DRIVE # TO LOW BYTE
                                ;***
                                ;GET IT TO R2
                                ;CLEAR THE UNUSED BITS
                                ;GET THE DRIVE #
                                ;SHIFT THE DRIVE # TO
                                ;ALIGN IT FOR THE DRIVE ADDR,
                                ;KEEP IT MOVING!
                                ;A LITTLE MORE!
                                ;THERE IT IS
                                ;
                                ;GET IT TO DISK ADDR. TEMP.
                                ;CALL MOUNT
                                ;RETURN

MOUNT:                          ;INCREMENT THE INDEX
                                ;***
                                ;SET ERROR CNTR TO 5
                                ;SET SIN CNTR TO 3
                                ;;TYPE ASCIZ STRING
                                ;;GET OVER THE ASCIZ
;;65$:                         <15><12>/MOUNT PACK ON DRIVE #/
64$:
```

```
  706  004362  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
  707  004364     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
  708  004365     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
  709  004366  104401  004374                  TYPE    ,67$            ;;TYPE ASCIZ STRING
  710  004372  000415                          BR      66$             ;;GET OVER THE ASCIZ
  711                                   ;;67$:  .ASCIZ  <15><12>/MAKE PACK WRITE ENABLE/
  712  004426                                66$:
  713  004426  104401  004434                  TYPE    ,69$            ;;TYPE ASCIZ STRING
  714  004432  000420                          BR      68$             ;;GET OVER THE ASCIZ
  715                                   ;;69$:  .ASCIZ  <15><12>/PRESS CONTINUE WHEN DRIVE RDY/
  716  004474                                68$:
  717  004474  000000                          HALT
  718  004476  004737  004504                  JSR     PC,INITIL       ;CALL INITIALIZER
  719  004502  000207                          RTS     PC              ;RETURN
  720
  721                                   ;THIS ROUTINE INITIALIZES A DRIVE AND INSURES THAT IT IS READY AND
  722                                   ;WRITE ENABLED, IT IS ENTERED FROM MOUNT OR FROM EXECUTE IF OPERATION FAILS
  723
  724  004504  010046                  INITIL: MOV     R0,-(SP)        ;SAVE R0
  725  004506  013700  001334                  MOV     @#DSKADR,R0     ;GET THE DRIVE # TO R0
  726  004512  042700  001777                  BIC     #1777,R0        ;CLEAR THE UNUSED BITS
  727  004516  005037  001346          INITI2: CLR     @#TIMR          ;CLEAR THE TIMER
  728  004522  105037  001323                  CLRB    @#TIMR2
  729  004526  000240                          NOP                     ;***
  730  004530  010077  174642                  MOV     R0,@RKDA        ;GET DRIVE # TO 'DA' REGISTER
  731  004534  012777  000001  174626          MOV     #1,@RKCS        ;ISSUE CONTROL RESET + GO
  732  004542  004737  005574                  JSR     PC,SMTME
  733  004546  105777  174616          1$:     TSTB    @RKCS           ;DID CONTROL READY SET
  734  004552  100423                          BMI     2$              ;IF YES BRANCH
  735  004554  005237  001346                  INC     @#TIMR          ;IF NO INCREMENT THE TIMER
  736  004560  001372                          BNE     1$              ;IF TIMER NOT ZERO BRANCH
  737  004562  104401  004570                  TYPE    ,65$            ;;TYPE ASCIZ STRING
  738  004566  000415                          BR      64$             ;;GET OVER THE ASCIZ
  739                                   ;;65$:  .ASCIZ  <15><12>/CONTROL RESET TIME OUT/
  740  004622                                64$:
  741  004622  010077  174550          2$:     MOV     R0,@RKDA        ;DRIVE NUMBER TO 'DA' REG.
  742  004626  105777  174532                  TSTB    @RKDS           ;IS DRIVE READY
  743  004632  100415                          BMI     3$              ;IF YES BRANCH
  744  004634  104401  004642                  TYPE    ,67$            ;;TYPE ASCIZ STRING
  745  004640  000411                          BR      66$             ;;GET OVER THE ASCIZ
  746                                   ;;67$:  .ASCIZ  <15><12>/DRIVE NOT READY/
  747  004664                                66$:
  748  004664  000714                          BR      INITI2          ;GO BACK TRY AGAIN
  749  004666  032777  000040  174470  3$:     BIT     #BIT5,@RKDS     ;IS DRIVE WRITE LOCKED?
  750  004674  001420                          BEQ     4$              ;IF NO, BRANCH
  751  004676  104401  004704                  TYPE    ,69$            ;;TYPE ASCIZ STRING
  752  004702  000414                          BR      68$             ;;GET OVER THE ASCIZ
  753                                   ;;69$:  .ASCIZ  <15><12>/DRIVE WRITE PROTECTED/
  754  004734                                68$:
  755  004734  000670                          BR      INITI2          ;YES, GO BACK TRY AGAIN
  756  004736  005037  001346          4$:     CLR     @#TIMR          ;CLEAR THE TIMER
  757  004742  010077  174430                  MOV     R0,@RKDA        ;GET THE DRIVE # TO 'DA' REGISTER
  758  004746  012777  000015  174414          MOV     #15,@RKCS       ;ISSUE DRIVE RESET + GO
  759  004754  004737  005574                  JSR     PC,SMTME
  760  004760  105777  174404          5$:     TSTB    @RKCS
  761  004764  100375                          BPL     5$
```



```
  762  004766  032777  000100  174370          BIT     #100,@RKDS      ;READ/WRITE/SEEK READY BIT SET?
  763  004774  001031                          BNE     6$              ;IF YES, BRANCH
  764  004776  005237  001346                  INC     @#TIMR          ;NO, INCREMENT THE TIMER
  765  005002  001366                          BNE     5$              ;GO BACK AND CHECK IF TIMER NOT 0
  766  005004  105737  001323                  TSTB    @#TIMR2
  767  005010  001003                          BNE     7$
  768  005012  105237  001323                  INCB    @#TIMR2
  769  005016  000760                          BR      5$
  770  005020                                7$:
  771  005020  104401  005026                  TYPE    ,71$            ;;TYPE ASCIZ STRING
  772  005024  000414                          BR      70$             ;;GET OVER THE ASCIZ
  773                                   ;;71$:  .ASCIZ  <15><12>/DRIVE RESET TIMED OUT/
  774  005056                                70$:
  775  005056  000727                          BR      4$              ;GO BACK, TRY AGAIN
  776  005060  012600                  6$:     MOV     (SP)+,R0        ;RESTORE R0
  777  005062  000207                          RTS     PC              ;RETURN TO CALLER
  778
  779                                   ;THIS ROUTINE TAKES CARE OF ALL LINKAGES FOR THE
  780                                   ;EXECUTE ROUTINE.  IT FORMS THE ADDRESS AND SETS UP ALL THE
  781                                   ;REGISTERS FOR THE WRITE OPERATION
  782
  783  005064  105037  001316          WRLINK: CLRB    @#COMND         ;INDICATE WRITE OPERATION
  784  005070  000240                          NOP                     ;***
  785  005072  113701  001314                  MOVB    @#DRIVE,R1      ;PICK UP THE DRIVE #
  786  005076  006101                          ROL     R1              ;MAKE IT A WORD INDEX
  787  005100  016137  001206  001362  1$:     MOV     DRV0(R1),@#PATTRN ;PICK UP THE DATA PATTERN
  788  005106  004737  005262                  JSR     PC,CYLADR       ;CALL CYLINDER ADDRESS
  789  005112  000401                          BR      2$              ;RETURN HERE IF NOT LAST BASE
  790  005114  000207                          RTS     PC              ;RETURN HERE IF LAST BASE
  791  005116  012737  001362  001336  2$:     MOV     #PATTRN,@#BUSADR ;GET THE ADDRESS OF THE OUTPUT
  792  005124  013737  001342  001340          MOV     @#CYLCNT,@#WRDCNT ;GET THE WORD COUNT
  793  005132  013737  001354  001332          MOV     @#WRITCS,@#CONTRL ;GET THE CONTROL + STATUS WORD
  794  005140  004737  005362                  JSR     PC,EXECUT       ;CALL EXECUTE
  795  005144  032737  000020  001334          BIT     #BIT4,@#DSKADR  ;WAS THIS WRITE SURFACE "1"?
  796  005152  001006                          BNE     3$              ;IF YES BRANCH
  797  005154  052737  000020  001334          BIS     #BIT4,@#DSKADR  ;SET SURFACE ONE BIT
  798  005162  105137  001362                  COMB    @#PATTRN        ;MAKE IT SURFACE ONE DATA
  799  005166  000753                          BR      2$              ;RELOAD REGISTERS AND EXECUTE
  800  005170  042737  000020  001334  3$:     BIC     #BIT4,@#DSKADR  ;SET UP FOR SURFACE "0"
  801  005176  105137  001362                  COMB    @#PATTRN        ;MAKE IT SURFACE 0 DATA
  802  005202  005202                          INC     R2              ;INC. THRU SELECTED CYL, OFFSET TABLE
  803  005204  004737  005272                  JSR     PC,CYLOFF       ;GET THE CYLINDER VALUE
  804  005210  000742                          BR      2$              ;RETURN HERE IF MORE TO READ
  805  005212  000207                          RTS     PC              ;RETURN HERE IF FINISHED
  806
  807                                   ;THIS ROUTINE EXPECTS TO FIND A MASK IN R2, AND FROM THIS MASK
  808                                   ;BUILDS A TABLE (AT CYLTBL) OF CYLINDER ADDRESS OFFSETS; THE TABLE
  809                                   ;IS TERMINATED BY A #377
  810
  811  005214  010546                  MASK:   MOV     R5,-(SP)        ;SAVE R5
  812  005216  042702  177400                  BIC     #177400,R2      ;CLR THE UNUSED BITS OF THE MASK
  813  005222  000240                          NOP                     ;***
  814  005224  012703  000200                  MOV     #200,R3         ;SET UP THE COMPARE MASK
  815  005230  005004                          CLR     R4              ;CLR THE INDEX COUNTER
  816  005232  012705  001274                  MOV     #CYLTBL,R5      ;GET THE CYLINDER TABLE ADDRESS
  817  005236  030203                  1$:     BIT     R2,R3           ;IS THE MASK BIT SELECTED IN BASE
```

```
000240  001401                  BEQ     2$              ;IF NO BRANCH
000242  110425                  MOVB    R4,(R5)+        ;MOVE THE CYLINDER BASE TO THE TABLE
000244  105204          2$:     INCB    R4              ;INCREMENT THE BASE
000246  006003                  ROR     R3              ;ROTATE THE COMPARE MASK
000250  103372                  BCC     1$              ;IF NOT DONE GO BACK
000252  112715  000377          MOVB    #377,(R5)       ;IF DONE LOAD FINISH FLAG
000256  012605                  MOV     (SP)+,R5        ;RESTORE R5
000260  000207                  RTS     PC              ;RETURN TO CALLER

                                ;THIS ROUTINE FORMS A CYLINDER ADDRESS FOR BOTH THE READ AND WRITE ROUTINES
                                ;WHEN THE BASE TABLE IS FULLY INDEXED IT RETURNS TO PC+2 OF CALLER

000262  012703  001266  CYLADR: MOV     #BASE,R3        ;GET THE CYLINDER TABLE ADDRESS
000266  012702  001274  CYLAD2: MOV     #CYLTBL,R2      ;GET THE SELECTED CYL BASE ADDR.
000272  111204  CYLOFF: MOVB    (R2),R4         ;GET THE SELECTED CYL VALUE TO R4
000274  000240                  NOP                     ;***
000276  122704  000377          CMPB    #377,R4         ;IS IT THE TABLE TERMINATOR?
000302  001416                  BEQ     BASINC          ;IF YES BRANCH
000304  005046                  CLR     -(SP)           ;INSURE CLEAN WORD
000306  111316                  MOVB    (R3),(SP)       ;GET THE CYL ADDRESS ON THE STACK
000310  062604                  ADD     (SP)+,R4        ;AND IT TO THE SELECTED OFFSET
000312  006104                  ROL     R4              ;SHIFT THIS RESULT
000314  006104                  ROL     R4              ;TO ALIGN THE NEWLY FORMED
000316  006104                  ROL     R4              ;CYLINDER ADDRESS WITH BITS
000320  006104                  ROL     R4              ;5 THRU 12 OF RKDA AND
000322  006104                  ROL     R4              ;STORE THIS IN DSKADR
000324  042737  017777  001334  BIC     #017777,@#DSKADR;CLEAR ALL BUT DRIVE NUMBER
000332  050437  001334          BIS     R4,@#DSKADR     ;PUT IT IN DSKADR
000336  000207                  RTS     PC
000340  005203          BASINC: INC     R3              ;PICK UP ADDRESS OF NEXT BASE CYL.
000342  000240                  NOP                     ;***
000344  027703  001274          CMP     #CYLTBL,R3      ;ARE YOU FINISHED?
000350  001401                  BEQ     RETRN3          ;IF YES, BRANCH
000352  000745                  BR      CYLAD2          ;NO GO BACK
000354  062716  000002  RETRN3: ADD     #2,(SP)         ;SET-UP FOR PC+2
000360  000207                  RTS     PC              ;RETURN

                                ;ROUTINE TO PERFORM INDICATED FUNCTION AND CHECK FOR
                                ;DONE AND ERRORS

000362  005037  001346  EXECUT: CLR     @#TIMR          ;CLEAR THE TIMER
000366  000240                  NOP                     ;HALT HERE TO CHECK COMMAND ABOUT TO BE EXECUTED
000370  105037  001323          CLRB    @#TIMR2         ;CLEAR SECOND TIMER
000374  013777  001334  173774  MOV     @#DSKADR,@RKDA  ;LOAD THE DISK ADDRESS REGISTER
000402  013777  001336  173764  MOV     @#BUSADR,@RKBA  ;LOAD THE BUS ADDRESS REGISTER
000410  013777  001340  173754  MOV     @#WRDCNT,@RKWC  ;LOAD THE WORD COUNT REGISTER
000416  013777  001332  173744  MOV     @#CONTRL,@RKCS  ;LOAD THE CONTROL REGISTER
000424  004737  005574          JSR     PC,SMTME        ;KILL TIME FOR RK11-C
000430  105777  173734  CHECK1: TSTB    @RKCS           ;IS CONTROL READY SET
000434  100011                  BPL     TIME            ;IF NO BRANCH
000436  004737  006062          JSR     PC,ERRCHK
000442  005737  001360          TST     @#ERRFLG        ;ERROR?
000446  001403                  BEQ     1$              ;IF NO BRANCH
000450  005037  001360          CLR     @#ERRFLG        ;CLEAR THE FLAG
000454  000742                  BR      EXECUT          ;TRY AGAIN
000456  000207          1$:     RTS     PC
```



```
000460  005237  001346  TIME:   INC     @#TIMR
000464  000240                  NOP                     ;***
000466  001360                  BNE     CHECK1
000470  105737  001323          TSTB    @#TIMR2         ;SECOND TIMEOUT?
000474  001003                  BNE     1$              ;IF YES BRANCH
000476  105237  001323          INCB    @#TIMR2         ;INDICATE SECOND TIMEOUT
000502  000752                  BR      CHECK1          ;GO BACK
000504  004737  004504  1$:     JSR     PC,INITIL
000510  104401  005516          TYPE    ,65$            ;;TYPE ASCIZ STRING
000514  000426                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <15><12>/TIMED OUT ON OPERATION RETRY IN PROGRESS/
000572                  64$:
000572  000673                  BR      EXECUT
000574  012737  000500  001346  SMTME:  MOV     #500,@#TIMR
000602  005337  001346  1$:     DEC     @#TIMR
000606  001375                  BNE     1$
000610  000207                  RTS     PC

                                ;THIS ROUTINE CHECKS TO SEE IF A DRIVE IS ACTIVE FROM A
                                ;WPITE OPERATION, IT GETS THE READ MASK TO R3, AND THE
                                ;EXPECTED DATA PATTERN TO "PATTERN", AND THEN CALLS ADDRESS
                                ;CONTROL.

000612  010046          RDLINK: MOV     R0,-(SP)        ;SAVE R0
000614  000240                  NOP                     ;***
000616  152737  000377  001316  BISB    #377,@#COMND    ;INDICATE READ OPERATION
000624  012701  001166          MOV     #LOGA,R1        ;GET THE TABLE ADDRESS TO R1
000630  012705  001305          MOV     #SECTBL,R5      ;GET THE SECTOR TABLE ADDRESS
000634  000405                  BR      RD2             ;SKIP OVER THE INDEX, FIRST PASS
000636  062701  000002  RD1:    ADD     #2,R1           ;ADD 2 TO THE ADDRESS
000642  022701  001206          CMP     #DRV0,R1        ;ARE YOU THRU THE ENTIRE TABLE
000646  001503                  BEQ     EXIT2           ;IF YES EXIT
000650  005711          RD2:    TST     (R1)            ;IS THE DRIVE ACTIVE
000652  100001                  BPL     RD3             ;IF YES BRANCH
000654  000770                  BR      RD1             ;IF NO GET NEXT WORD
000656  011100          RD3:    MOV     (R1),R0         ;GET THE ACTIVE WORD TO R0
000660  010002                  MOV     R0,R2           ;COPY THE WORD
000662  000300                  SWAB    R0              ;GET THE DRIVE # TO THE LOW BYTE
000664  042700  177770          BIC     #177770,R0      ;CLR ALL BUT THE DRIVE #
000670  110037  001317          MOVB    R0,@#WRINBY     ;SAVE THE DRIVE #
000674  006100                  ROL     R0              ;MAKE IT AN INDEX
000676  016037  001206  001362  MOV     DRV0(R0),@#PATTRN ;PICK UP THE DATA PATTERN
000704  004737  005214          JSR     PC,MASK         ;CALL THE MASK SUBROUTINE
000710  004737  005262          JSR     PC,CYLADR       ;GO FORM A CYLINDER ADDRESS
000714  000401                  BR      RD4             ;RETURN HERE IF NOT LAST BASE ADDR.
000716  000747                  BR      RD1             ;IF LAST BASE ADDRESS, RETURN HERE
000720  053737  001352  001334  RD4:    BIS     @#DSKTMP,@#DSKADR ;SET THE DRIVE # BITS IN DISK ADDR.
000726  152537  001334  RD5:    BISB    (R5)+,@#DSKADR  ;SET THE SECTOR BITS IN DISK ADDR.
000732  012737  007336  001336  RD6:    MOV     #RDBUFF,@#BUSADR ;GET THE BUFFER ADDR.
000740  000240                  NOP                     ;***
000742  013737  001344  001340          MOV     @#SECCNT,@#WRDCNT ;GET THE WORD COUNT
000750  013737  001356  001332          MOV     @#READCS,@#CONTRL ;GET THE READ CONTROL WORD
000756  004737  005362                  JSR     PC,EXECUT       ;DO THE READ
000762  004737  006430                  JSR     PC,RDCHK        ;CHECK THE DATA
000766  042737  000017  001334          BIC     #17,@#DSKADR    ;CLR THE SECTOR BITS IN DISK ADDR.
000774  022705  001311                  CMP     #DRCNT1,R5      ;WAS THIS THE LAST SECTOR?
```

```
  930  006000  001352                          BNE     RD5             ;IF NO GO BACK
  931  006002  012705  001305                  MOV     #SECTBL,R5      ;IF YES RESET SECTOR POINTER
  932  006006  032737  000020  001334          BIT     #BIT4,@#DSKADR  ;WAS IT SURFACE "1" THAT WAS READ?
  933  006014  001006                          BNE     RD7             ;IF YES, BRANCH
  934  006016  052737  000020  001334          BIS     #BIT4,@#DSKADR  ;NO, SET SURFACE "1" BIT
  935  006024  105137  001362                  COMB    @#PATTRN        ;MAKE HEAD "1" PATTERN
  936  006030  000736                          BR      RD5             ;GO BACK AND EXECUTE
  937  006032  042737  000020  001334  RD7:    BIC     #BIT4,@#DSKADR  ;CLEAR THE SURFACE BIT
  938  006040  105137  001362                  COMB    @#PATTRN        ;MAKE IT SURFACE Ø DATA
  939  006044  005202                          INC     R2              ;INCREMENT THE SELECTED CYL TABLE POINTER
  940  006046  004737  005272                  JSR     PC,CYLOFF       ;GO FORM NEXT ADDRESS
  941  006052  000722                          BR      RD4             ;IF HERE IT IS NOT THE LAST BASE ADDR
  942  006054  000670                          BR      RD1             ;IF HERE GET NEXT WORD-DRIVE FINISHED
  943
  944  006056  012600                  EXIT2:  MOV     (SP)+,RØ        ;RESTORE RØ
  945  006060  000207                          RTS     PC              ;RETURN TO MAIN LINE CODE
  946
  947                                   ;THE ERROR CHECK ROUTINE CHECKS IF AN ERROR OCCURRED
  948                                   ;ON WRITING OR READING.
  949
  950  006062  032777  140000  173300  ERRCHK: BIT     #140000,@RKCS   ;HARD ERROR OR ERROR SET?
  951  006070  000240                          NOP                     ;HALT HERE TO EXAMINE ERROR REG.,ECT.
  952  006072  001420                          BEQ     TSTSN1          ;IF NO, GO TEST 'SIN' BIT
  953  006074  012777  000001  173266          MOV     #1,@RKCS        ;IF YES, ISSUE CNTROL RESET + GO
  954  006102  004737  005574                  JSR     PC,SMTME
  955  006106  012777  177777  173244          MOV     #-1,@ERRFLG     ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)
  956  006114  105777  173250          1$:     TSTB    @RKCS           ;CNTROL READY BIT SET (FROM CNTROL RESET)
  957  006120  100375                          BPL     1$              ;IF NO WAIT.  (IF HUNG HERE RUN STATIC)
  958  006122  105337  001321                  DECB    @#ECNT          ;DECREMENT ERROR COUNTER
  959  006126  001002                          BNE     TSTSN1          ;HAVE ERROR BITS SET 5 TIMES?
  960  006130  000137  006212                  JMP     @#RESTRT        ;IF HERE 5 ERRORS HAVE OCCURRED
  961  006134  032777  001000  173222  TSTSN1: BIT     #1000,@RKDS     ;SEEK INCOMPLETE SET?
  962  006142  000240                          NOP                     ;***
  963  006144  001530                          BEQ     RETRN2          ;BRANCH IF NO
  964  006146  012777  000015  173214          MOV     #15,@RKCS       ;IF YES, ISSUE DRIVE RESET, GO
  965  006154  012777  177777  173176          MOV     #-1,@ERRFLG     ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)
  966  006162  004737  005574                  JSR     PC,SMTME
  967  006166  105777  173176          2$:     TSTB    @RKCS
  968  006172  100375                          BPL     2$
  969  006174  032777  000100  173162          BIT     #100,@RKDS      ;"R/W/S READY" BIT SET?
  970  006202  001771                          BEQ     2$              ;IF NO WAIT! (IF HUNG HERE RUN STATIC)
  971  006204  105337  001322                  DECB    @#CNTSIN        ;DECREMENT SEEK INCOMPLETE COUNTER
  972  006210  001106                          BNE     RETRN2          ;IF 3 'SIN' ERRORS FALL THROUGH
  973  006212  105737  001321          RESTRT: TSTB    @#ECNT
  974  006216  000240                          NOP                     ;***
  975  006220  001421                          BEQ     1$
  976  006222  104401  006230                  TYPE    ,65$            ;;TYPE ASCIZ STRING
  977  006226  000415                          BR      64$             ;;GET OVER THE ASCIZ
  978                                   ;;65$:  .ASCIZ  <15><12>/3 'SIN' ERRORS OCCURRED/
  979  006262                           64$:
  980  006262  000415                          BR      2$
  981  006264                           1$:
  982  006264  104401  006272                  TYPE    ,67$            ;;TYPE ASCIZ STRING
  983  006270  000412                          BR      66$             ;;GET OVER THE ASCIZ
  984                                   ;;67$:  .ASCIZ  <15><12>/5 ERRORS OCCURRED/
  985  006316                           66$:
```



```
  986  006316                           2$:
  987  006316  104401  006324                  TYPE    ,69$            ;;TYPE ASCIZ STRING
  988  006322  000422                          BR      68$             ;;GET OVER THE ASCIZ
  989                                   ;;69$:  .ASCIZ  / DRIVE DECLARED DOWN!! NOT TESTED !/
  990  006370                           68$:
  991  006370  105737  001316                  TSTB    @#COMND         ;TEST THE COMMAND
  992  006374  100006                          BPL     3$              ;IF WRITE, BRANCH
  993  006376  062706  000004                  ADD     #4,SP           ;POINT TO SAVE RØ
  994  006402  012600                          MOV     (SP)+,RØ        ;GET RØ BACK
  995  006404  052710  100000                  BIS     #BIT15,(RØ)     ;SET THE DOWN BIT
  996  006410  000207                          RTS     PC              ;RETURN TO MAIN CODE
  997  006412  052710  100000          3$:     BIS     #BIT15,(RØ)     ;SET THE DOWN BIT
  998  006416  012706  001100                  MOV     #STACK,SP       ;RESTORE THE STACK
  999  006422  000137  003304                  JMP     @#GO1
 1000  006426  000207                  RETRN2: RTS     PC
 1001
 1002                                   ;THIS ROUTINE CHECKS A SECTORS WORTH OF DATA ON A READ OPERATION
 1003                                   ;IT ALLOWS 5 ERROR PRINTOUTS PER SECTOR
 1004
 1005  006430  010446                  RDCHK:  MOV     R4,-(SP)        ;SAVE   R4
 1006  006432  010546                          MOV     R5,-(SP)        ;SAVE R5 FOR RDLINK
 1007  006434  105037  001320                  CLRB    @#HDRFLG        ;CLEAR THE PRINT HEADER FLAG
 1008  006440  000240                          NOP                     ;***
 1009  006442  012737  000005  001350          MOV     #5,@#CHKCNT     ;PUT ERROR COUNT IN CHECK COUNT
 1010  006450  012704  007336                  MOV     #RDBUFF,R4      ;GET THE TABLE ADDRESS TO R4
 1011  006454  013705  001362                  MOV     @#PATTRN,R5     ;GET THE EXPECTED DATA TO R5
 1012  006460  020524                  1$:     CMP     R5,(R4)+        ;ARE THEY THE SAME
 1013  006462  001515                          BEQ     3$              ;IF YES BRANCH
 1014  006464  105737  001320                  TSTB    @#HDRFLG        ;IS THE HEADER FLAG CLEAR
 1015  006470  001046                          BNE     2$              ;IF NO BRANCH
 1016  006472  104401  006500                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 1017  006476  000420                          BR      64$             ;;GET OVER THE ASCIZ
 1018                                   ;;65$:  .ASCIZ  <15><12>/ERROR! DATA WRITTEN BY DRIVE /
 1019  006540                           64$:
 1020  006540  152737  000377  001320          BISB    #377,@#HDRFLG   ;SET THE HEADER FLAG
 1021  006546  113746  001317                  MOVB    @#WRTNBY,-(SP)  ;PICK UP THE DRIVE # THAT WROTE
 1022  006552  104403                          TYPOS
 1023  006554     001                          .BYTE   1
 1024  006555     000                          .BYTE   Ø
 1025  006556  104401  006564                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 1026  006562  000411                          BR      66$             ;;GET OVER THE ASCIZ
 1027                                   ;;67$:  .ASCIZ  / CANNOT BE READ./
 1028  006606                           66$:
 1029  006606                           2$:
 1030  006606  104401  006614                  TYPE    ,69$            ;;TYPE ASCIZ STRING
 1031  006612  000405                          BR      68$             ;;GET OVER THE ASCIZ
 1032                                   ;;69$:  .ASCIZ  <15><12>/  ADDR#/
 1033  006626                           68$:
 1034  006626  013746  001334                  MOV     @#DSKADR,-(SP)  ;PICK UP THE ADDRESS THAT FAILED
 1035  006632  104403                          TYPOS
 1036  006634     006                          .BYTE   6
 1037  006635     001                          .BYTE   1
 1038  006636  104401  006644                  TYPE    ,71$            ;;TYPE ASCIZ STRING
 1039  006642  000405                          BR      70$             ;;GET OVER THE ASCIZ
 1040                                   ;;71$:  .ASCIZ  /  EXPCTD=/
 1041  006656                           70$:
```

```
1042  006656  010546                        MOV     R5,-(SP)        ;PICK UP THE EXPECTED DATA (GOOD)
1043  006660  104404                        TYPON
1044  006662  104401  006670                TYPE    ,73$            ;;TYPE ASCIZ STRING
1045  006666  000405                        BR      72$             ;;GET OVER THE ASCIZ
1046                                  ;;73$:  .ASCIZ  /  RECVD=/
1047  006702                                72$:
1048  006702  016446  177776                MOV     -2(R4),-(SP)    ;PICK UP THE RECEIVED DATA (BAD)
1049  006706  104404                        TYPON
1050  006710  005337  001350                DEC     @#CHKCNT        ;DECREMENT THE CHECK COUNT
1051  006714  001403                        BEQ     4$              ;IF ZERO, BRANCH
1052  006716  022704  010336          3$:   CMP     #MANSEL,R4      ;DONE ALL CHECKS?
1053  006722  001256                        BNE     1$              ;IF NO, GO BACK
1054  006724  012605                4$:     MOV     (SP)+,R5        ;RESTORE R5
1055  006726  012604                        MOV     (SP)+,R4        ;RESTORE R4
1056  006730  000207                        RTS     PC              ;RETURN TO CALLER
1057
1058                                  ;THIS ROUTINE BUILDS A TABLE OF PARAMETERS TO PASS TO THE SECOND SYSTEM
1059                                  ;IT PACKS THE INFO FOR TWO DRIVES INTO ONE WORD
1060
1061  006732  005003                SECONE: CLR     R3              ;CLEAR THE WORD COUNTER
1062  006734  000240                        NOP                     ;***
1063  006736  012702  001256                MOV     #MSKTBL,R2
1064  006742  005042                6$:     CLR     -(R2)
1065  006744  022702  001246                CMP     #PASTBL,R2
1066  006750  001374                        BNE     6$
1067  006752  012700  001166                MOV     #LOGA,R0        ;GET THE ACTIVE TABLE ADDRESS
1068  006756  012001                1$:     MOV     (R0)+,R1        ;PICK UP THE WORD
1069  006760  000301                        SWAB    R1              ;GET THE DRIVE # TO THE LOW BYTE
1070  006762  042701  177400                BIC     #177400,R1      ;CLEAR THE UNWANTED BITS
1071  006766  106101                        ROLB    R1              ;ROTATE THE BYTE, WAS DOWN SET?
1072  006770  103002                        BCC     2$              ;IF NO BRANCH
1073  006772  052701  000001                BIS     #BIT0,R1        ;SHOW THE DRIVE AS DOWN IN THE TABLE
1074  006776  105701                2$:     TSTB    R1              ;IS THIS THE SYSTEM #2 DRIVE?
1075  007000  100404                        BMI     3$              ;BRANCH IF YES
1076  007002  142701  000360                BICB    #360,R1         ;CLEAR THE UNUSED BITS
1077  007006  110122                        MOVB    R1,(R2)+        ;GET THIS # TO THE PASS TABLE
1078  007010  000762                        BR      1$              ;GET THE NEXT WORD FROM ACTIVE TABLE
1079  007012  110112                3$:     MOVB    R1,(R2)         ;GET THE LAST DRIVE TO THE PASS TABLE
1080  007014  012702  001246                MOV     #PASTBL,R2      ;RESTORE THE TABLE POINTER
1081  007020  104401  007026                TYPE    ,65$            ;;TYPE ASCIZ STRING
1082  007024  000425                        BR      64$             ;;GET OVER THE ASCIZ
1083                                  ;;65$:  .ASCIZ  <15><12>/LOAD AND START ADDRESS 210 ON SYSTEM #2/
1084  007100                                64$:
1085  007100  104401  007106                TYPE    ,67$            ;;TYPE ASCIZ STRING
1086  007104  000424                        BR      66$             ;;GET OVER THE ASCIZ
1087                                  ;;67$:  .ASCIZ  <15><12>/AND TYPE THE BELOW WHEN ASKED FOR IT./
1088  007156                                66$:
1089  007156  005203                4$:     INC     R3              ;INCREMENT THE WORD COUNTER
1090  007160  104401  007166                TYPE    ,69$            ;;TYPE ASCIZ STRING
1091  007164  000405                        BR      68$             ;;GET OVER THE ASCIZ
1092                                  ;;69$:  .ASCIZ  <15><12>/WORD  /
1093  007200                                68$:
1094  007200  010346                        MOV     R3,-(SP)        ;GET THE WORD COUNT ON THE STACK
1095  007202  104403                        TYPOS
1096  007204     006                        .BYTE   6
1097  007205     000                        .BYTE   0
```



```
1098  007206  104401  007214                TYPE    ,71$            ;;TYPE ASCIZ STRING
1099  007212  000401                        BR      70$             ;;GET OVER THE ASCIZ
1100                                  ;;71$:  .ASCIZ  /=/
1101  007216                                70$:
1102  007216  012246                        MOV     (R2)+,-(SP)     ;GET THE FIRST TO THE STACK
1103  007220  104403                        TYPOS
1104  007222     006                        .BYTE   6
1105  007223     001                        .BYTE   1
1106  007224  032762  100000  177776        BIT     #BIT15,-2(R2)   ;WAS THIS THE TABLE TERMINATOR
1107  007232  001004                        BNE     5$              ;BRANCH IF YES
1108  007234  032762  000200  177776        BIT     #BIT7,-2(R2)    ;TERMINATOR?
1109  007242  001745                        BEQ     4$              ;IF NO BRANCH
1110  007244  005740                5$:     TST     -(R0)
1111  007246  000000                        HALT
1112
1113  007250                                RETFR2:
1114  007250  104401  007256                TYPE    ,65$            ;;TYPE ASCIZ STRING
1115  007254  000404                        BR      64$             ;;GET OVER THE ASCIZ
1116                                  ;;65$:  .ASCIZ  <15><12>/WORD=/
1117  007266                                64$:
1118  007266  104411                        RDOCT
1119  007270  012602                        MOV     (SP)+,R2        ;GET THE WORD FROM SYSTEM 2 TO TABLE
1120  007272  042702  177400                BIC     #177400,R2
1121  007276  012704  001166                MOV     #LOGA,R4        ;SET POINTER LOOK FOR FIRST "UP" DRIVE
1122  007302  005724                1$:     TST     (R4)+           ;DRIVE UP?
1123  007304  100776                        BMI     1$              ;IF NO BRANCH
1124  007306  010437  001100                MOV     R4,@#$PASS
1125  007312  014401                        MOV     -(R4),R1
1126  007314  000240                        NOP                     ;***
1127  007316  004737  004204                JSR     PC,LDFLG        ;CALL D02+
1128  007322  004737  005612                JSR     PC,RDLINK       ;CALL READ CHECK
1129  007326  013700  001100                MOV     @#$PASS,R0
1130  007332  000137  003352                JMP     @#EXTFR2        ;GO TO END OF TEST
1131
1132  007336  000400                RDBUFF: .BLKW   400
1133
1134  010336  000240                MANSEL: NOP                     ;TABLE TERMINATOR
1135
1136
1137
1138
1139
1140  010340                                BADONE:
1141  010340  104401  010346                TYPE    ,65$            ;;TYPE ASCIZ STRING
1142  010344  000401                        BR      64$             ;;GET OVER THE ASCIZ
1143                                  ;;65$:  .ASCIZ  /?/
1144  010350                                64$:
1145
1146
1147                                          .SBTTL  OSCILLATING SEEK ROUTINE
1148
1149  010350                                SECT.2:
1150  010350  104401  010356                TYPE    ,65$            ;;TYPE ASCIZ STRING
1151  010354  000416                        BR      64$             ;;GET OVER THE ASCIZ
1152                                  ;;65$:  .ASCIZ  <15><12>/OSCILLATING SEEK PACKAGE/
1153  010412                                64$:
```

```
                                                        MOV     #DRIV0,R0       ;FIND OUT WHICH DRIVES ARE
                                                        CLR     R1              ;PRESENT AND PUT THE
                                                1$:     MOV     R1,@RKDA        ;DRIVE #'S IN A TABLE STARTING
                                                        TSTB    @RKDS           ;AT 'DRIV0'. BITS 15-13 CONTAINS
                                                        BPL     2$              ;THE DRIVE #.
                                                        MOV     R1,(R0)+
                                                2$:     ADD     #20000,R1
                                                        BNE     1$
                                                        MOV     #-1,(R0)        ;SET THE TERMINATOR TO THE TABLE

                                                INIT.2: MOV     RKDA,R2
                                                        MOV     #1,@RKCS        ;ISSUE CONTROL RESET + GO !
                                                        JSR     PC,SMTME
                                                1$:     TSTB    @RKCS           ;DID CONTROL READY SET?
                                                        BPL     1$              ;IF NO WAIT! (IF HUNG RUN STATIC)
                                                        MOV     #DRIV0,R0
                                                3$:     MOV     (R0)+,@RKDA
                                                        MOV     #15,@RKCS       ;ISSUE DRIVE RESET + GO!
                                                        JSR     PC,SMTME
                                                2$:     TSTB    @RKCS
                                                        BPL     2$
                                                        CMP     #-1,(R0)
                                                        BNE     3$
                                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                        BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <15><12>/SET SW0 TO SW7 TO SELECT DRIVES FOR TEST AND CONT/
                                                64$:
                                                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                                                        BR      66$             ;;GET OVER THE ASCIZ
                                                ;;67$:  .ASCIZ  <15><12>/RESET SW0 TO SW7 TO TEST ALL AVAIL DRIVES/
                                                66$:
                                                7$:     CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER IN USE ?
                                                        BNE     8$              ;BR IF NOT
                                                        GTSWR                   ;REQUEST NEW CONTENTS FOR SWITCH REG
                                                        BR      9$              ;CONTINUE
                                                9$:     HALT                    ;WAIT FOR OPERATOR TO ENTER NEW SWR VALUE
                                                8$:     MOVB    @SWR,R4         ;SW0 TO SW7 TO R4
                                                        BEQ     4$              ;NONE SET, SO TEST ALL
                                                        MOV     #DRIV0,R0       ;TABLE TO STORE DRIVE ADDRS
                                                        CLR     R1              ;ADDR OF DRIVE
                                                6$:     ROR     R4              ;NEXT SWITCH TO CARRY
                                                        BCC     5$              ;SWITCH NOT SET
                                                        MOV     R1,(R0)+        ;SWITCH SET, SO MOVE ADDR TO TABLE
                                                5$:     ADD     #20000,R1       ;ADDR OF NEXT DRIVE
                                                        BNE     6$              ;ALL DONE WHEN ZERO
                                                        MOV     #-1,(R0)+       ;TABLE TERMINATOR
                                                4$:     TSTB    OSPFLG          ;TYPED ONCE?
                                                        BNE     RWSRDY          ;IF YES, DONT RETYPE
                                                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                                                        BR      68$             ;;GET OVER THE ASCIZ
                                                ;;69$:  .ASCIZ  <15><12>/TOGGLE THE "FIRST CYLINDER ADDRESS" (OUTER LIMIT)/
                                                68$:
                                                        TYPE    ,71$            ;;TYPE ASCIZ STRING
                                                        BR      70$             ;;GET OVER THE ASCIZ
                                                ;;71$:  .ASCIZ  <15><12>/INTO THE LOW BYTE (BIT0-7) OF THE SWITCH REGISTER AND THE "LAST
```



```
                                                70$:
                                                        TYPE    ,73$            ;;TYPE ASCIZ STRING
                                                        BR      72$             ;;GET OVER THE ASCIZ
                                                ;;73$:  .ASCIZ  <15><12>/CYLINDER ADDRESS" (INNER LIMIT) INTO THE HIGH/
                                                72$:
                                                        TYPE    ,75$            ;;TYPE ASCIZ STRING
                                                        BR      74$             ;;GET OVER THE ASCIZ
                                                ;;75$:  .ASCIZ  <15><12>/       BYTE (BIT8-15), THEN PRESS CONTINUE./
                                                74$:
                                                        INCR    OSPFLG
                                                RWSRDY: BIT     #100,@RKDS      ;"READ/WRITE/SEEK READY" BIT SET?
                                                        BEQ     RWSRDY          ;IF NO WAIT! (IF HUNG RUN STATIC)
                                                TRYAGN: CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER  IN USE ?
                                                        BNE     1$              ;BR IF NOT
                                                        GTSWR                   ;GET NEW CONTENTS
                                                        BR      CONTIN          ;CONTINUE
                                                1$:     HALT
                                                CONTIN: MOV     #1,@RKCS        ;CONTROL RESET
                                                        TSTB    @RKCS
                                                        BPL     .-4
                                                        MOV     SWR,R5          ;GET THE ADDRESS OF THE SWITCH REG. TO R5
                                                1$:     MOVB    (R5)+,R1        ;GET A BYTE TO R1
                                                        BIC     #177400,R1      ;CLEAR THE UNUSED BITS
                                                        CMP     #312,R1         ;IS ADDRESS LEGAL?
                                                        BPL     2$              ;BRANCH IF YES
                                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                        BR      64$             ;;GET OVER THE ASCIZ
                                                ;;65$:  .ASCIZ  <15><12>/INVALID ADDRESS IN SWITCH REGISTER TRY AGAIN/
                                                64$:
                                                        BR      TRYAGN          ;GO BACK FOR NEW PARAMETERS
                                                2$:     ROL     R1              ;ROTATING THIS REGISTER
                                                        ROL     R1              ;MAKES THE BYTE FROM THE
                                                        ROL     R1              ;SWITCH REGISTER LINE UP
                                                        ROL     R1              ;WITH THE CYLINDER BITS OF
                                                        ROL     R1              ;THE RKDA REGISTER.
                                                        BIC     #160037,R1      ;CLEAR THE UNUSED BITS
                                                        BIT     #1,R5           ;IS THIS THE LOW BYTE?
                                                        BNE     3$              ;BRANCH IF YES
                                                        MOV     R1,@#SEEKI      ;STORE THE INNER LIMIT OF THE SEEK
                                                        BR      SEKSET          ;START SEEKS
                                                3$:     MOV     R1,@#SEEKO      ;STORE THE OUTER LIMIT OF THE SEEK
                                                        BR      1$              ;GO BACK AND GET HIGH BYTE.

                                                SEKSET: MOV     #50,R3          ;GET THE NUMBER OF SEEK CYCLES TO R3
                                                        MOV     @#SEEKO,R4      ;ADD THE OUTER LIMIT CYLINDER ADDRESS
                                                        MOV     @#SEEKI,R5      ;ADD THE INNER LIMIT CYLINDER ADDRESS

                                                LDSEEK: MOV     #DRIV0,R0       ;INITIALIZE POINTER
                                                5$:     MOV     R4,@RKDA        ;GET INNER LIMIT CYL ADRES
                                                        BIS     (R0)+,@RKDA     ;SET DRIVE ADRES
                                                        MOV     #11,@RKCS       ;ISSUE SEEK+GO! (FOR INNER LIMIT)
                                                        JSR     PC,SMTME
                                                3$:     TSTB    @RKCS
                                                        BPL     3$

                                                        CMP     #-1,(R0)        ;ALL DRIVES DONE?
```

```
011624  001361                          BNE     5$              ;NO
011626  012700  020470                  MOV     #DRIV0,R0
011632  012077  167540          6$:     MOV     (R0)+,@RKDA     ;ADRES THE DRIVE
011636  032777  000100  167520  7$:     BIT     #RWS,@RKDS      ;SEEK DONE?
011644  001774                          BEQ     7$              ;NO, WAIT
011646  022710  177777                  CMP     #-1,(R0)        ;ALL DRIVES DONE?
011652  001367                          BNE     6$              ;NO

011654  012700  020470                  MOV     #DRIV0,R0
011660  010577  167512          8$:     MOV     R5,@RKDA        ;SET OUTER CYL ADRES
011664  052077  167506                  BIS     (R0)+,@RKDA     ;SET DRIVE ADRES
011670  012777  000011  167472          MOV     #11,@RKCS       ;ISSUE SEEK+GO! (FOR OUTER LIMIT)
011676  004737  005574                  JSR     PC,SMTME
011702  105777  167462          9$:     TSTB    @RKCS
011706  100375                          BPL     9$

011710  022710  177777                  CMP     #-1,(R0)        ;ALL DONE?
011714  001361                          BNE     8$              ;NO

011716  012700  020470                  MOV     #DRIV0,R0
011722  012077  167450          10$:    MOV     (R0)+,@RKDA     ;SET DRIVE ADRES
011726  032777  000100  167430  11$:    BIT     #RWS,@RKDS              ;SEEK DONE?
011734  001774                          BEQ     11$
011736  022710  177777                  CMP     #-1,(R0)        ;ALL DONE?
011742  001367                          BNE     10$
011744  005303                          DEC     R3              ;DONE 50 SEEK CYCLES (100 SEEKS)
011746  001306                          BNE     LOOSEEK         ;IF NO BRANCH (KEEP CYCLING)
011750  000605                          BR      CONTIN          ;CHECK SWR FOR CHANGE AND CONTINUE



                                        .SBTTL  FORMATTER-SURFACE VERIFIER

011752                          BAD.IN:
011752  104401  011760                  TYPE    ,65$            ;;TYPE ASCIZ STRING
011756  000401                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /?/
011762                          64$:
011762                          SECT.1:
011762  104401  011770                  TYPE    ,65$            ;;TYPE ASCIZ STRING
011766  000441                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/FORMATTER-SURFACE VERIFIER,SET SW REG FOR DRV #'S, PRESS CONT./
012072                          64$:
012072  022737  000176  001140          CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER IN USE ?
012100  001002                          BNE     1$              ;BR IF NOT
012102  104405                          GTSWR                   ;GET SWITCH REGISTER VALUE
012104  000401                          BR      2$              ;CONTINUE
012106  000000                  1$:     HALT                    ;WAIT FOR 'CONTINUE'
012110  012737  000001  020320  2$:     MOV     #1,SHFCNT       ;SET SHIFT COUNT
012116  005037  020322                  CLR     DRVCNT          ;CLEAR DRIVE COUNT
012122  033777  020320  167010  S13:    BIT     SHFCNT,@SWR     ;IS THIS SW SET?
012130  001011                          BNE     S10             ;YES FORMAT THIS DRIVE
012132  006337  020320          S11:    ASL     SHFCNT
012136  005237  020322                  INC     DRVCNT
```



```
012142  022737  000010  020322          CMP     #10,DRVCNT      ;ALL DONE?
012150  001556                          BEQ     GD1
012152  000763                          BR      S13
012154  013700  020322          S10:    MOV     DRVCNT,R0
012160  104401  001161                  TYPE    ,SCRLF
012164  104401  020324                  TYPE    ,FM1            ;TYPE 'DRIVE'
012170  010046                          MOV     R0,-(SP)        ;TYPE DRIVE #
012172  104402                          TYPOC
012174  000300                          SWAB    R0
012176  006100                          ROL     R0
012200  006100                          ROL     R0
012202  006100                          ROL     R0
012204  006100                          ROL     R0
012206  006100                          ROL     R0
012210  010037  001352                  MOV     R0,@#DSKTMP
012214  012737  000000  001422          MOV     #0,ERRWF
012222  012737  000000  001424          MOV     #0,ERRRF        ;CLEAR OUT ERROR COUNTS.
012230  012737  000000  001426          MOV     #0,ERRRFC
012236  012737  000000  001430          MOV     #0,ERRWCH
012244  012737  000000  001432          MOV     #0,ERRWCS
012252  012737  177750  001416  S12:    MOV     #-24.,RWC       ;SET WORD COUNT FOR READ FORMAT.
012260  012737  164000  001412          MOV     #-6144.,WC      ;SET UP WORD COUNT FOR WRITES.
012266  012737  000000  001420          MOV     #0,EXTR         ;CLEAR EXTRA BIT FOR 12 SECTOR PACK.
012274  012701  177772          COMMON: MOV     #-6.,R1         ;SET UP LOOP COUNT FOR THE CLEANER.
012300  012777  014500  167070  COM:    MOV     #14500,@RKDA    ;SET UP FOR A SEEK TO 202.
012306  053777  001352  167062          BIS     @#DSKTMP,@RKDA
012314  105777  167050          1$:     TSTB    @RKCS           ;IS THE CONTROLER READY?
012320  100375                          BPL     1$              ;NO SO WAIT.
012322  012777  000011  167040          MOV     #11,@RKCS       ;DO THE SEEK.
012330  032777  000100  167026  2$:     BIT     #BIT6,@RKDS     ;IS THE SEEK DONE?
012336  001774                          BEQ     2$              ;NO SO WAIT.
012340  105777  167024          3$:     TSTB    @RKCS           ;IS CONTROLER READY?
012344  100375                          BPL     3$              ;NO
012346  012777  000015  167014          MOV     #15,@RKCS       ;DO A DRIVE RESET.
012354  032777  000100  167002  4$:     BIT     #BIT6,@RKDS     ;IS DRIVE RESET DONE.
012362  001774                          BEQ     4$              ;NO
012364  005201                          INC     R1              ;COUNT THE CLEANER LOOP.
012366  001344                          BNE     COM             ;MORE TO GO.
012370  012737  000000  001410          MOV     #0,DA           ;START OUT AT CYL. 0.
012376  012777  177777  167002  NEXT:   MOV     #177777,@BA     ;PUT ALL ONE'S IN BUFFER.
012404  004137  012512                  JSR     R1,IO           ;GO DO THE DISK THING.
012410  012777  000000  166770          MOV     #0,@BA          ;PUT ALL ZERO'S IN BUFFER.
012416  004137  012512                  JSR     R1,IO           ;GO DO IT AGIAN.
012422  012777  125252  166756          MOV     #125252,@BA     ;PUT A ALT. PATTERN IN BUFFER.
012430  004137  012512                  JSR     R1,IO           ;ONCE MORE.
012434  005177  166746                  COM     @BA             ;COMPLEMENT THE LAST PATTERN.
012440  004137  012512                  JSR     R1,IO           ;AND AGAIN.
012444  062737  000040  001410          ADD     #40,DA          ;INCREMENT TO THE NEXT CYL.
012452  022737  014540  001410          CMP     #14540,DA       ;ARE WE DONE WITH THIS ONE?
012460  001346                          BNE     NEXT            ;NO SO DO THE NEXT CYL.
012462                          GOOT:
012462  104401  012470                  TYPE    ,65$            ;;TYPE ASCIZ STRING
012466  000406                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CR><LF>/PACK GOOD/
012504                          64$:
012504  000612                          BR      S11
```

```
1378  012506  000137  001440          GO1:    JMP     @#START         ;RESTART
1379                                  ;
1380                                  ;
1381                                  ;DISK I/O SUBROUTINE.
1382                                  ;
1383                                  ;
1384                                  ;SET UP FOR A WRITE/FORMAT.
1385  012512  013777  001412  166652  IO:      MOV     WC,@RKWC        ;SET UP THE WORD COUNT REG.
1386  012520  013777  001410  166650          MOV     DA,@RKDA        ;SET UP THE DISK ADDRESS.
1387  012526  053777  001352  166642          BIS     @#DSKTMP,@RKDA  ;SET THE UNIT NUMBER UP.
1388  012534  013777  001406  166632          MOV     BA,@RKBA        ;SET UP THE BUSS ADDRESS.
1389  012542  012777  000000  166620          MOV     #0,@RKCS        ;CLEAR THE CONTROL REG. FOR SET UP.
1390  012550  052777  006000  166612          BIS     #BIT10+BIT11,@RKCS ;SET FORMAT&INHIBT INC. BITS.
1391  012556  052777  000002  166604          BIS     #BIT1,@RKCS     ;SET UP WRITE FUN.
1392  012564  053777  001420  166576          BIS     EXTR,@RKCS      ;SET UP 12OR16 SECTOR PACK.
1393  012572  052777  000001  166570          BIS     #BIT0,@RKCS     ;GO DO THE WRITE FORMAT.
1394  012600  105777  166564          1$:     TSTB    @RKCS           ;IS WRITE FORMAT DONE?
1395  012604  100375                          BPL     1$              ;NO SO WAIT.
1396  012606  005777  166556                  TST     @RKCS           ;WAS THERE A ERROR?
1397  012612  100541                          BMI     WFERR           ;YES GO SERVICE IT.
1398  012614  012737  000000  001422          MOV     #0,ERRWF        ;CLEAR OUT THE ERROR COUNTER.
1399                                  ;
1400                                  ;SET UP FOR A READ/FORMAT
1401                                  ;
1402  012622  013777  001416  166542          MOV     RWC,@RKWC       ;SET UP WORD COUNT REG.
1403  012630  013777  001410  166540          MOV     DA,@RKDA        ;SET UP DISK ADDRESS.
1404  012636  053777  001352  166532          BIS     @#DSKTMP,@RKDA  ;SET THE UNIT NUMBER .
1405  012644  013777  001414  166522          MOV     RBA,@RKBA       ;SET UP THE BUSS ADDRESS.
1406  012652  012777  000000  166510          MOV     #0,@RKCS        ;CLEAR THE CONTROL REG.
1407  012660  052777  002000  166502          BIS     #BIT10,@RKCS    ;SET THE FORMAT BIT.
1408  012666  052777  000004  166474          BIS     #BIT2,@RKCS     ;SET UP READ FUN.
1409  012674  053777  001420  166466          BIS     EXTR,@RKCS      ;SET UP 12 OR 16 SECTOR PACK.
1410  012702  052777  000001  166460          BIS     #BIT0,@RKCS     ;GO DO THE READ FORMAT.
1411  012710  105777  166454          2$:     TSTB    @RKCS           ;IS THE READ FORMAT DONE?
1412  012714  100375                          BPL     2$              ;NO SO WAIT.
1413  012716  032777  040000  166444          BIT     #BIT14,@RKCS    ;WAS TRERE A ERROR?
1414  012724  001134                          BNE     RFERR           ;YES GO SERVICE IT.
1415  012726  012737  000000  001424          MOV     #0,ERRRF        ;CLEAR OUT THE ERROR COUNT.
1416                                  ;
1417                                  ;SET UP FOR A WRITE CHECK.
1418                                  ;
1419  012734  013777  001412  166430          MOV     WC,@RKWC        ;SET UP WORD COUNT REG.
1420  012742  013777  001410  166426          MOV     DA,@RKDA        ;SET UP DISK ADDRESS.
1421  012750  053777  001352  166420          BIS     @#DSKTMP,@RKDA  ;SET UP THE UNIT NUMBER
1422  012756  013777  001406  166410          MOV     BA,@RKBA        ;SET UP BUSS ADDRESS.
1423  012764  012777  000000  166376          MOV     #0,@RKCS        ;CLEAR THE CONTROL REG.
1424  012772  052777  004400  166370          BIS     #BIT11+BIT8,@RKCS ;SET INHIBIT INCR.&STOP ON SOFT ERROR BITS.
1425  013000  053777  001420  166362          BIS     EXTR,@RKCS      ;SET 12 OR 16 SECTOR PACK.
1426  013006  052777  000006  166354          BIS     #BIT1+BIT2,@RKCS ;SET UP WRITE CHECK FUN.
1427  013014  052777  000001  166346          BIS     #BIT0,@RKCS     ;GO DO THE WRITE CHECK.
1428                                  ;
1429                                  ;CHECK HEADERS READ BY THE READ/FORMAT.
1430                                  ;
1431  013022  013703  001416                  MOV     RWC,R3          ;PUT NUMBER OF WORDS TO
1432  013026  005403                          NEG     R3              ;CHECK IN REG 3.
1433  013030  063703  001414                  ADD     RBA,R3          ;SET REG 3 TO THE LAST WORD TO BE CHECKED.
```



```
1434  013034  013702  001414                  MOV     RBA,R2          ;SET REG 2 TO STARTING ADD. OF BUFF.
1435  013040  023722  001410          MORE:   CMP     DA,(2)+         ;CHECK THAT HEADER IS RIGHT.
1436  013044  001073                          BNE     RFCERR          ;THIS HEADER WAS WRONG GO SERVICE IT.
1437  013046  020302                          CMP     R3,R2           ;ARE WE DONE?
1438  013050  001373                          BNE     MORE            ;NO GO CHECK THE NEXT ONE.
1439  013052  012737  000000  001426          MOV     #0,ERRRFC       ;CLEAR OUT THE ERROR COUNT.
1440                                  ;
1441                                  ;LETS CHECK ON THE WRITE CHECK WE STARTED.
1442                                  ;
1443  013060  105777  166304          1$:     TSTB    @RKCS
1444  013064  100375                          BPL     1$              ;THE CONTROLER IS STILL BUSY.
1445  013066  005777  166276                  TST     @RKCS           ;WAS THERE A ERROR?
1446  013072  100407                          BMI     WCERRR          ;YES GO SERVICE IT.
1447  013074  012737  000000  001430          MOV     #0,ERRWCH       ;CLEAR OUT THE
1448  013102  012737  000000  001432          MOV     #0,ERRWCS       ;ERROR COUNTERS.
1449  013110  000201                          RTS     R1              ;RETURN TO THE MAIN LINE.
1450  013112  000137  013576          WCERRR: JMP     WCERR
1451                                  ;
1452                                  ;ERRORS FOR WRITE FORMAT.
1453                                  ;
1454  013116  005237  001422          WFERR:  INC     ERRWF           ;ADD ONE TO THE ERROR COUNT.
1455  013122  022737  000004  001422          CMP     #4,ERRWF        ;HAS IT HAPPEND 4 TIMES ON THIS CYL.
1456  013130  001016                          BNE     RETRY           ;NO.
1457  013132                          SYSER:
1458  013132  104401  013140                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1459  013136  000410                          BR      64$             ;;GET OVER THE ASCIZ
1460                                  ;;65$:  .ASCIZ  <15><12>/SYSTEM ERROR/
1461  013160                          64$:
1462  013160  000000                          HALT                    ;LET THE TECH. BREATH.
1463  013162  000137  001440                  JMP     START           ;RESTART THE TEST.
1464  013166  012777  000000  166174  RETRY:  MOV     #0,@RKCS        ;CLEAR OUT THE CONTROL REG.
1465  013174  012777  000015  166166          MOV     #15,@RKCS       ;DO A DRIVE RESET.
1466  013202  032777  000100  166154  1$:     BIT     #BIT6,@RKDS     ;IS IT DONE.
1467  013210  001774                          BEQ     1$              ;NO SO WAIT.
1468  013212  000137  012512                  JMP     IO              ;TRY AGAIN.
1469                                  ;
1470                                  ;ERRORS FOR READ/FORMAT.
1471                                  ;
1472  013216  005237  001424          RFERR:  INC     ERRRF           ;ADDONE TO ERROR COUNT.
1473  013222  022737  000004  001424          CMP     #4,ERRRF        ;HAS IT HAPPEND 4 TIMES ON THIS CYL?
1474  013230  001356                          BNE     RETRY           ;NO DO IT AGAIN.
1475  013232  000737                          BR      SYSER           ;YES SO TELL HIM SO.
1476                                  ;
1477                                  ;READ/FORMAT ERRORS FOUND BY SOFTWARE CHECKS.
1478                                  ;
1479  013234  005237  001426          RFCERR: INC     ERRRFC          ;ADD ONE TO ERROR COUNT.
1480  013240  105777  166124          1$:     TSTB    @RKCS           ;WAIT FOR THE WRITE CHECK.
1481  013244  100375                          BPL     1$
1482  013246  022737  000004  001426          CMP     #4,ERRRFC       ;IS IT 4?
1483  013254  001401                          BEQ     FAILED          ;PUT OUT FAILED MESSAGE.
1484  013256  000743                          BR      RETRY           ;NO SO GO TRY IT AGAIN.
1485  013260  042777  000037  166110  FAILED: BIC     #37,@RKDA       ;PUT WHICH SECTORS HEADER
1486  013266  042702  177740                  BIC     #177740,R2
1487  013272  060277  166100                  ADD     R2,@RKDA        ;FAILED IN RKDA FOR THE MESSAGE.
1488  013276                          FAIL:
1489  013276  104401  013304                  TYPE    ,65$            ;;TYPE ASCIZ STRING
```

```
1490  013302  000417                        BR      64$             ;;GET OVER THE ASCIZ
1491                                  ;;65$:  .ASCIZ  <15><12>/PACK FAILED AT (IN OCTAL) /
1492  013342                          64$:
1493  013342  017701  166030                MOV     @RKDA,R1        ;GENERAT THE CYL,SECTOR,SURFACE
1494  013346  010102                        MOV     R1,R2   ;MESSAGE FROM RKDA
1495  013350  042702  177770                BIC     #177770,R2
1496  013354  062702  000260                ADD     #260,R2
1497  013360  110237  013537                MOVB    R2,SEC+1
1498  013364  004337  013564                JSR     R3,SHF3
1499  013370  042702  177776                BIC     #177776,R2
1500  013374  062702  000260                ADD     #260,R2
1501  013400  110237  013536                MOVB    R2,SEC
1502  013404  004337  013570                JSR     R3,SHF1
1503  013410  042702  177776                BIC     #177776,R2
1504  013414  062702  000260                ADD     #260,R2
1505  013420  110237  013550                MOVB    R2,SUR
1506  013424  004337  013570                JSR     R3,SHF1
1507  013430  042702  177770                BIC     #177770,R2
1508  013434  062702  000260                ADD     #260,R2
1509  013440  110237  013526                MOVB    R2,CYL+2
1510  013444  004337  013564                JSR     R3,SHF3
1511  013450  042702  177770                BIC     #177770,R2
1512  013454  062702  000260                ADD     #260,R2
1513  013460  110237  013525                MOVB    R2,CYL+1
1514  013464  004337  013564                JSR     R3,SHF3
1515  013470  042702  177774                BIC     #177774,R2
1516  013474  062702  000260                ADD     #260,R2
1517  013500  110237  013524                MOVB    R2,CYL
1518  013504  104401  013514                TYPE    ,1$             ;TYPE OUT THE GENERATED MESSAGE
1519  013510  000137  013556                JMP     STOP            ;BYPASS THE INLINE MESSAGE
1520  013514  005015  054503  027114  1$:   .ASCII  <15><12>/CYL.  /
1521  013522  020040
1522  013524  030060  020060          CYL:  .ASCII  /000 /
1523  013530  042523  027103  020040        .ASCII  /SEC.  /
1524  013536  030060     040     SEC:  .ASCII  /00 /
1525  013541     123  051125  027106        .ASCII  /SURF.  /
1526  013546  020040
1527  013550  020060  005015     000  SUR:  .ASCIZ  /0 /<15><12>
1528          013556                        .EVEN
1529  013556  000000                  STOP: HALT                    ;LET OPER DO HIS THING.
1530  013560  000137  001440                JMP     START           ;RESTART THE TEST.
1531                                        ;
1532                                        ;SHIFT SUBROUTINES.
1533                                        ;
1534  013564  006201                  SHF3: ASR     R1              ;HERE FOR A SHIFT OF 3.
1535  013566  006201                  SHF2: ASR     R1              ;HERE FOR A SHIFT OF 2.
1536  013570  006201                  SHF1: ASR     R1              ;HERE FOR A SHIFT OF 1.
1537  013572  010102                        MOV     R1,R2           ;PUT RESULTES IN THE WORKING REG.
1538  013574  000203                        RTS     R3
1539                                        ;
1540                                        ;ERRORS FOR WRITE CHECK.
1541                                        ;
1542  013576  032777  040000  165564  WCERR: BIT     #BIT14,@RKCS    ;WAS IT A HARD ERROR.
1543  013604  001010                        BNE     WCHERR          ;YES GO PROSSES IT.
1544  013606  005237  001432                INC     ERRWCS          ;ADD 1 TO THE SOFT ERROR COUNT.
1545  013612  022737  000004  001432        CMP     #4,ERRWCS       ;HAS THERE BEEN 4 OF THEM?
```



```
1546  013620  001626                        BEQ     FAIL            ;YES PUT OUT FAILED MESSAGE.
1547  013622  000137  012512                JMP     IO              ;NO SO TRY AGAIN.
1548  013626  005237  001430          WCHERR: INC     ERRWCH          ;ADD 1 TO THE HARD ERROR COUNT.
1549  013632  022737  000004  001430        CMP     #4,ERRWCH       ;HAS THERE BEEN 4 OF THEM?
1550  013640  001402                        BEQ     SYSERR          ;YES PUT OUT SYSTEM ERROR MESSAGE.
1551  013642  000137  013166                JMP     RETRY           ;NO SO TRY AGAIN.
1552  013646  000137  013132          SYSERR: JMP     SYSER
1553                                        ;
1554                                        ;BUFFERS.
1555                                        ;
1556  013652  000000                  BUFF: .WORD   0
1557  013654  000030                  RBUFF: .BLKW   30
1558
1559
1560
1561
1562
1563                                        .SBTTL  RK05 CONTROL PANEL TEST
1564
1565  013734                          RAD.ON:
1566  013734  104401  013742                TYPE    ,65$            ;;TYPE ASCIZ STRING
1567  013740  000401                        BR      64$             ;;GET OVER THE ASCIZ
1568                                  ;;65$:  .ASCIZ  /?/
1569  013744                          64$:
1570  013744                          SECT.0:
1571  013744  104401  013752                TYPE    ,65$            ;;TYPE ASCIZ STRING
1572  013750  000424                        BR      64$             ;;GET OVER THE ASCIZ
1573                                  ;;65$:  .ASCIZ  <15><12>/RK05 CONTROL PANEL TEST, WHICH DRIVE?/
1574  014022                          64$:
1575  014022  104407                        RDCHR
1576  014024  011600                        MOV     (SP),R0
1577  014026  104403                        TYPOS
1578  014030     001                        .BYTE   1
1579  014031     000                        .BYTE   0
1580  014032  162700  000060                SUB     #60,R0
1581  014036  100736                        BMI     BAD.ON
1582  014040  022700  000010                CMP     #10,R0
1583  014044  003733                        BLE     BAD.ON
1584  014046  110037  001314                MOVB    R0,@#DRIVE
1585  014052  000300                        SWAB    R0
1586  014054  006100                        ROL     R0
1587  014056  006100                        ROL     R0
1588  014060  006100                        ROL     R0
1589  014062  006100                        ROL     R0
1590  014064  006100                        ROL     R0
1591  014066  010037  001352                MOV     R0,@#DSKTMP
1592  014072  104401  014100                TYPE    ,67$            ;;TYPE ASCIZ STRING
1593  014076  000414                        BR      66$             ;;GET OVER THE ASCIZ
1594                                  ;;67$:  .ASCIZ  <15><12>/MOUNT PACK ON DRIVE#/
1595  014130                          66$:
1596  014130  013746  001314                MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
1597  014134  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
1598  014136     001                        .BYTE   1               ;;TYPE 1 DIGIT(S)
1599  014137     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
1600  014140  104401  014146                TYPE    ,69$            ;;TYPE ASCIZ STRING
1601  014144  000425                        BR      68$             ;;GET OVER THE ASCIZ
```

```
1602                                    ;;69$:  .ASCIZ  <15><12>/PLACE DRIVE IN RUN ;SHOULD SEE THE RUN,/
1603  014220                            68$:
1604  014220  104401  014226                    TYPE    ,71$            ;;TYPE ASCIZ STRING
1605  014224  000423                            BR      70$             ;;GET OVER THE ASCIZ
1606                                    ;;71$:  .ASCIZ  <15><12>/POWER, AND ON CYLINDER LAMPS LIGHT./
1607  014274                            70$:
1608  014274  104401  014302                    TYPE    ,73$            ;;TYPE ASCIZ STRING
1609  014300  000425                            BR      72$             ;;GET OVER THE ASCIZ
1610                                    ;;73$:  .ASCIZ  <15><12>/MAKE DRIVE WRITE ENABLE PRESS CONTINUE/
1611  014354                            72$:
1612  014354  000000                            HALT
1613  014356  004737  016134                    JSR     PC,WRRDCK       ;GO WRITE 0'S, RD 0'S, CHECK
1614  014362  104401  014370                    TYPE    ,75$            ;;TYPE ASCIZ STRING
1615  014366  000430                            BR      74$             ;;GET OVER THE ASCIZ
1616                                    ;;75$:  .ASCIZ  <15><12><15><12>/WRITE PROTECT THE DRIVE THEN PRESS CONTINUE/
1617  014450                            74$:
1618  014450  000000                            HALT
1619  014452  004737  016344                    JSR     PC,WRPRO        ;GO TRY OVERWRITE
1620  014456  104401  014464                    TYPE    ,77$            ;;TYPE ASCIZ STRING
1621  014462  000426                            BR      76$             ;;GET OVER THE ASCIZ
1622                                    ;;77$:  .ASCIZ  <15><12><15><12>/CLEAR WRITE PROTECT THEN PRESS CONTINUE/
1623  014540                            76$:
1624  014540  000000                            HALT
1625  014542  004737  016134                    JSR     PC,WRRDCK       ;GO WRITE 0'S, RD 0'S, CHECK
1626  014546  013777  001352  164622            MOV     @#DSKTMP,@RKDA  ;SET UP DRIVE#
1627  014554  012777  000017  164606            MOV     #17,@RKCS       ;FUNCTION WRITE PROTECT
1628  014562  004737  005574                    JSR     PC,SMTME        ;GO WAIST TIME FOR RK11-C
1629  014566  105777  164576            1$:     TSTB    @RKCS           ;IS CONTROL READY SET
1630  014572  100375                            BPL     1$              ;IF NO, BRANCH
1631  014574  004737  016344                    JSR     PC,WRPRO        ;GO TRY OVERWRITE OF IERO'S
1632  014600                            PRTTWO:
1633  014600  104401  014606                    TYPE    ,65$            ;;TYPE ASCIZ STRING
1634  014604  000431                            BR      64$             ;;GET OVER THE ASCIZ
1635                                    ;;65$:  .ASCIZ  <15><12><15><12>/CAUTION! TRY TO OPEN THE DOOR, DO NOT FORCE!/
1636  014670                            64$:
1637  014670  104401  014676                    TYPE    ,67$            ;;TYPE ASCIZ STRING
1638  014674  000414                            BR      66$             ;;GET OVER THE ASCIZ
1639                                    ;;67$:  .ASCIZ  <15><12>/DOOR SHOULD NOT OPEN!/
1640  014726                            66$:
1641  014726  104401  014734                    TYPE    ,69$            ;;TYPE ASCIZ STRING
1642  014732  000420                            BR      68$             ;;GET OVER THE ASCIZ
1643                                    ;;69$:  .ASCIZ  <15><12>/PRESS CONTINUE WHEN FINISHED/
1644  014774                            68$:
1645  014774  000000                            HALT
1646  014776  104401  015004                    TYPE    ,71$            ;;TYPE ASCIZ STRING
1647  015002  000426                            BR      70$             ;;GET OVER THE ASCIZ
1648                                    ;;71$:  .ASCIZ  <15><12><15><12>/PUT DRIVE IN LOAD, WAIT FOR LOAD LIGHT/
1649  015060                            70$:
1650  015060  104401  015066                    TYPE    ,73$            ;;TYPE ASCIZ STRING
1651  015064  000420                            BR      72$             ;;GET OVER THE ASCIZ
1652                                    ;;73$:  .ASCIZ  <15><12>/PRESS CONTINUE WHEN FINISHED/
1653  015126                            72$:
1654  015126  000000                            HALT
1655  015130  013777  001352  164240            MOV     @#DSKTMP,@RKDA  ;SET UP DISK ADDRESS
1656  015136  012777  000015  164224            MOV     #15,@RKCS       ;ISSUE A DRIVE RESET
1657  015144  004737  005574                    JSR     PC,SMTME        ;KILL TIME FOR RK11-C
```



```
1658  015150  105777  164214            1$:     TSTB    @RKCS           ;CONTROL READY SET?
1659  015154  100375                            BPL     1$              ;IF NO, BRANCH
1660  015156  032777  100000  164202            BIT     #BIT15,@RKER    ;DRE SET
1661  015164  001023                            BNE     2$              ;IF YES BRANCH
1662  015166  104401  015174                    TYPE    ,75$            ;;TYPE ASCIZ STRING
1663  015172  000420                            BR      74$             ;;GET OVER THE ASCIZ
1664                                    ;;75$:  .ASCIZ  <15><12>/DRE=BIT15 OF RKER DID NOT SET/
1665  015234                            74$:
1666  015234  032777  140000  164126    2$:     BIT     #140000,@RKCS   ;DID HARD ERROR ON ERROR SET
1667  015242  001015                            BNE     3$              ;BRANCH IF YES
1668  015244  104401  015252                    TYPE    ,77$            ;;TYPE ASCIZ STRING
1669  015250  000412                            BR      76$             ;;GET OVER THE ASCIZ
1670                                    ;;77$:  .ASCIZ  <15><12>/ERROR DID NOT SET/
1671  015276                            76$:
1672  015276  012777  000001  164064    3$:     MOV     #1,@RKCS        ;ISSUE A CONTROL RESET
1673  015304  004737  005574                    JSR     PC,SMTME        ;WAIST TIME
1674  015310  105777  164054            4$:     TSTB    @RKCS           ;CONTROL READY SET
1675  015314  100375                            BPL     4$              ;IF NO, BRANCH
1676  015316  032777  100000  164042            BIT     #BIT15,@RKER    ;'DRE' CLEAR
1677  015324  001425                            BEQ     5$              ;IF YES BRANCH
1678  015326  104401  015334                    TYPE    ,79$            ;;TYPE ASCIZ STRING
1679  015332  000422                            BR      78$             ;;GET OVER THE ASCIZ
1680                                    ;;79$:  .ASCIZ  <15><12>/CONTROL RESET DID NOT CLEAR 'DRE'/
1681  015400                            78$:
1682  015400  032777  140000  163762    5$:     BIT     #140000,@RKCS   ;ERROR BITS CLEAR
1683  015406  001431                            BEQ     X               ;IF YES BRANCH
1684  015410  104401  015416                    TYPE    ,81$            ;;TYPE ASCIZ STRING
1685  015414  000426                            BR      80$             ;;GET OVER THE ASCIZ
1686                                    ;;81$:  .ASCIZ  <15><12>/CONTROL RESET DID NOT CLEAR 'ERROR', RKCS/
1687  015472                            80$:
1688  015472                            X:
1689  015472  104401  015500                    TYPE    ,65$            ;;TYPE ASCIZ STRING
1690  015476  000422                            BR      64$             ;;GET OVER THE ASCIZ
1691                                    ;;65$:  .ASCIZ  <15><12><15><12>/OPEN THE DOOR, PUT DRIVE IN RUN/
1692  015544                            64$:
1693  015544  104401  015552                    TYPE    ,67$            ;;TYPE ASCIZ STRING
1694  015550  000425                            BR      66$             ;;GET OVER THE ASCIZ
1695                                    ;;67$:  .ASCIZ  <15><12>/CAUTION! IF RUN LIGHT ON ERROR! DEPRESS/
1696  015624                            66$:
1697  015624  104401  015632                    TYPE    ,69$            ;;TYPE ASCIZ STRING
1698  015630  000426                            BR      68$             ;;GET OVER THE ASCIZ
1699                                    ;;69$:  .ASCIZ  <15><12>/LOAD IMMEDIATELY, CONTINUE WHEN FINISHED/
1700  015706                            68$:
1701  015706  000000                    XX:     HALT
1702  015710  104401  015716                    TYPE    ,65$            ;;TYPE ASCIZ STRING
1703  015714  000422                            BR      64$             ;;GET OVER THE ASCIZ
1704                                    ;;65$:  .ASCIZ  <15><12><15><12>/REMOVE THE PACK, CLOSE THE DOOR/
1705  015762                            64$:
1706  015762  104401  015770                    TYPE    ,67$            ;;TYPE ASCIZ STRING
1707  015766  000423                            BR      66$             ;;GET OVER THE ASCIZ
1708                                    ;;67$:  .ASCIZ  <15><12>/PUT DRIVE IN RUN, DRIVE SHOULD NOT/
1709  016036                            66$:
1710  016036  104401  016044                    TYPE    ,69$            ;;TYPE ASCIZ STRING
1711  016042  000423                            BR      68$             ;;GET OVER THE ASCIZ
1712                                    ;;69$:  .ASCIZ  <15><12>/RUN...INTERCLOCKS HAVE BEEN CHECKED/
1713  016112                            68$:
```

```
1714  016112  104401  016120          TYPE    ,71$            ;;TYPE ASCIZ STRING
1715  016116  000404                  BR      70$             ;;GET OVER THE ASCIZ
1716                          ;;71$:  .ASCIZ  <15><12>/DONE!/
1717  016130                  70$:
1718  016130  000137  001440          JMP     @#START
1719  016134  005037  001362  WRRDCK: CLR     @#PATTRN        ;MAKE A PATTERN OF ZERO'S
1720  016140  013777  001352  163230  MOV     @#DSKTMP,@RKDA  ;SET UP DRIVE ADDRESS
1721  016146  012777  000001  163214  MOV     #1,@RKCS        ;ISSUE A CONTROL RESET
1722  016154  004737  005574          JSR     PC,SMTME
1723  016160  105777  163204  5$:     TSTB    @RKCS           ;CONTROL READY SET
1724  016164  100375                  BPL     5$              ;IF NO BRANCH
1725  016166  013777  001352  163202  MOV     @#DSKTMP,@RKDA  ;SET UP DRIVE ADDRESS
1726  016174  012777  001362  163172  MOV     #PATTRN,@RKBA   ;GET BUSS ADDRESS
1727  016202  013777  001344  163162  MOV     @#SECCNT,@RKWC  ;WORD COUNT 1 SECTOR
1728  016210  013777  001354  163152  MOV     @#WRITCS,@RKCS  ;IBA + WRITE + GO
1729  016216  004737  005574          JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1730  016222  105777  163142  1$:     TSTB    @RKCS           ;CONTROL READY SET
1731  016226  100375                  BPL     1$              ;IF NO, BRANCH
1732  016230  013777  001352  163140  MOV     @#DSKTMP,@RKDA  ;SET UP RK REGISTERS
1733  016236  012777  007336  163130  MOV     #RDBUFF,@RKBA   ;TO READ ONE SECTOR
1734  016244  013777  001344  163120  MOV     @#SECCNT,@RKWC  ;TO THE READ BUFFER
1735  016252  013777  001356  163110  MOV     @#READCS,@RKCS
1736  016260  004737  005574          JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1737  016264  105777  163100  2$:     TSTB    @RKCS           ;CONTROL READY SET
1738  016270  100375                  BPL     2$              ;IF NO, BRANCH
1739  016272  012704  007336          MOV     #RDBUFF,R4      ;GET BUFFER TO R4
1740  016276  005005                  CLR     R5              ;SET UP TO COMPARE
1741  016300  020524  3$:     CMP     R5,(R4)+        ;FOR ZERO'S
1742  016302  001414                  BEQ     4$              ;IF OK, BRANCH
1743  016304  104401  016312          TYPE    ,65$            ;;TYPE ASCIZ STRING
1744  016310  000410                  BR      64$             ;;GET OVER THE ASCIZ
1745                          ;;65$:  .ASCIZ  <15><12>/WRITE FAILED/
1746  016332                  64$:
1747  016332  000700                  BR      WRRDCK          ;GO BACK TRY AGAIN
1748  016334  022704  010336  4$:     CMP     #MANSEL,R4      ;DONE ALL CHECKS
1749  016340  001357                  BNE     3$              ;IF NO, BRANCH
1750  016342  000207                  RTS     PC              ;IF YES, RETURN
1751  016344  032777  000040  163012  WRPRO:  BIT     #BIT5,@RKDS     ;BIT 5 ON
1752  016352  001021                  BNE     1$              ;IF YES, BRANCH
1753  016354  104401  016362          TYPE    ,65$            ;;TYPE ASCIZ STRING
1754  016360  000416                  BR      64$             ;;GET OVER THE ASCIZ
1755                          ;;65$:  .ASCIZ  <15><12>/WPS=BIT5 OF RKDS NOT SET/
1756  016416                  64$:
1757  016416  012737  177777  001362  1$:     MOV     #177777,@#PATTRN ;GO LOAD ALL
1758  016424  013777  001352  162744          MOV     @#DSKTMP,@RKDA  ;RK REGISTERS
1759  016432  012777  001362  162734          MOV     #PATTRN,@RKBA   ;TO WRITE
1760  016440  013777  001344  162724          MOV     @#SECCNT,@RKWC  ;ALL ONES (WITH WRITE LOCK)
1761  016446  013777  001354  162714          MOV     @#WRITCS,@RKCS  ;OVER THE ZERO'S
1762  016454  004737  005574          JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1763  016460  105777  162704  2$:     TSTB    @RKCS           ;CONTROL READY SET?
1764  016464  100375                  BPL     2$              ;IF NO, BRANCH
1765  016466  032777  020000  162672  BIT     #BIT13,@RKER    ;WLO BIT SET
1766  016474  001032                  BNE     3$              ;IF YES BRANCH
1767  016476  104401  016504          TYPE    ,67$            ;;TYPE ASCIZ STRING
1768  016502  000427                  BR      66$             ;;GET OVER THE ASCIZ
1769                          ;;67$:  .ASCIZ  <15><12>/EXPECTED WLO=BIT13 OF RKER BUT DID NOT SET/
```



```
1770  016562                  66$:
1771  016562  012777  000001  162600  3$:     MOV     #1,@RKCS        ;DO A CONTROL RESET
1772  016570  004737  005574          JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1773  016574  105777  162570  4$:     TSTB    @RKCS           ;CONTROL READY SET?
1774  016600  100375                  BPL     4$              ;IF NO BRANCH
1775  016602  032777  020000  162556  BIT     #BIT13,@RKER    ;WLO BIT CLEAR
1776  016610  001431                  BEQ     RDCHK0          ;IF YES, BRANCH
1777  016612  104401  016620          TYPE    ,69$            ;;TYPE ASCIZ STRING
1778  016616  000426                  BR      68$             ;;GET OVER THE ASCIZ
1779                          ;;69$:  .ASCIZ  <15><12>/CONTROL RESET DID NOT CLEAR 'WLO' OF RKER/
1780  016674                  68$:
1781  016674  013777  001352  162474  RDCHK0: MOV     @#DSKTMP,@RKDA  ;SET UP RK REGISTERS
1782  016702  012777  007336  162464          MOV     #RDBUFF,@RKBA   ;TO READ SECTOR 0,
1783  016710  013777  001344  162454          MOV     @#SECCNT,@RKWC  ;CYLINDER 0, HEAD 0
1784  016716  013777  001356  162444          MOV     @#READCS,@RKCS  ;TO ENSURE NO WRITE TOOK PLACE
1785  016724  004737  005574          JSR     PC,SMTME        ;KILL TIME
1786  016730  105777  162434  3$:     TSTB    @RKCS
1787  016734  100375                  BPL     3$
1788  016736  012703  000005          MOV     #5,R3
1789  016742  012704  007336          MOV     #RDBUFF,R4      ;CHECK TO INSURE NO WRITE
1790  016746  005005                  CLR     R5              ;TOOK PLACE
1791  016750  020524  1$:     CMP     R5,(R4)+        ;WITH WRITE LOCK
1792  016752  001474                  BEQ     2$
1793  016754  005303                  DEC     R3              ;DEC THE ERROR COUNT
1794  016756  001475                  BEQ     4$              ;IF ZERO BRANCH
1795  016760  104401  016760          TYPE    ,65$            ;;TYPE ASCIZ STRING
1796  016764  000422                  BR      64$             ;;GET OVER THE ASCIZ
1797                          ;;65$:  .ASCIZ  <15><12>/WRITE OCCURRED WITH WRITE PROTECT/
1798  017032                  64$:
1799  017032  005744                  TST     -(R4)
1800  017034  104401  017042          TYPE    ,67$            ;;TYPE ASCIZ STRING
1801  017040  000410                  BR      66$             ;;GET OVER THE ASCIZ
1802                          ;;67$:  .ASCIZ  <15><12>/BUFFER ADDR=/
1803  017062                  66$:
1804  017062  010446                  MOV     R4,-(SP)
1805  017064  104403                  TYPOS
1806  017066     006                  .BYTE   6
1807  017067     001                  .BYTE   1
1808  017070  104401  017076          TYPE    ,69$            ;;TYPE ASCIZ STRING
1809  017074  000406                  BR      68$             ;;GET OVER THE ASCIZ
1810                          ;;69$:  .ASCIZ  /    EXPCTD=/
1811  017112                  68$:
1812  017112  010546                  MOV     R5,-(SP)
1813  017114  104404                  TYPON
1814  017116  104401  017124          TYPE    ,71$            ;;TYPE ASCIZ STRING
1815  017122  000406                  BR      70$             ;;GET OVER THE ASCIZ
1816                          ;;71$:  .ASCIZ  /    RECVD=/
1817  017140                  70$:
1818  017140  012446                  MOV     (R4)+,-(SP)
1819  017142  104404                  TYPON
1820  017144  022704  010336  2$:     CMP     #MANSEL,R4      ;FINISHED ALL CHECKS
1821  017150  001277                  BNE     1$              ;IF NO, BRANCH
1822  017152  000207          4$:     RTS     PC              ;RETURN
1823
1824
1825
```

1826


```
1827                                    ;THE FOLLOWING REVISION WAS MADE BY JIM KAPADIA
1828
1829                                            .SBTTL  CONTROL PANEL TEST # 2
1830
1831                                    ;THIS IS THE ENTRY POINT INTO CONTROL PANEL TEST #2.  ALL
1832                                    ;THE DRIVES THAT ARE PRESENT AND IN 'RDY' CONDITION ARE
1833                                    ;REPORTED (ON LINE).
1834
1835  017154  000240                    SECT.4: NOP
1836  017156  012777  000001  162204            MOV     #1,@RKCS
1837  017164  105777  162200            3$:     TSTB    @RKCS
1838  017170  100375                            BPL     3$
1839  017172  012700  020470                    MOV     #DRIV0,R0
1840  017176  005001                            CLR     R1
1841  017200  005002                            CLR     R2
1842
1843  017202  010210                    1$:     MOV     R2,(R0)         ;SET UP ADDRESS TABLE
1844  017204  010277  162166                    MOV     R2,@RKDA        ;ADDRESS THE DRIVE
1845  017210  105777  162150                    TSTB    @RKDS           ;IS IT PRESENT?
1846  017214  100021                            BPL     2$              ;NO
1847
1848  017216  104401  020324                    TYPE    ,EM1            ;TYPE 'DRIVE'
1849  017222  010146                            MOV     R1,-(SP)        ;TYPE OUT DRIVE #
1850  017224  104402                            TYPOC
1851  017226  104401  020335                    TYPE    ,EM2            ;TYPE 'ON LINE'
1852  017232  052710  000300                    BIS     #BIT6+BIT7,(R0) ;SET BITS INDICATING THIS
1853                                                                    ;DRIVE PRESENT
1854
1855  017236  012777  000015  162124            MOV     #15,@RKCS       ;ISSUE A DRIVE RESET
1856  017244  004737  005574                    JSR     PC,SMTME        ;ALLOW SOME TIME
1857  017250  032777  000100  162106  4$:     BIT     #RWS,@RKDS      ;WAIT FOR RWS RDY
1858  017256  001774                            BEQ     4$
1859
1860  017260  005720                    2$:     TST     (R0)+
1861  017262  005201                            INC     R1
1862  017264  062702  020000                    ADD     #20000,R2       ;NXT DRIVE
1863  017270  001344                            BNE     1$              ;ALL DONE?
1864
1865  017272  104401  001161                    TYPE    ,$CRLF
1866
1867                                    ;THIS CODE CHECKS THE CONDITION OF 'DRY' BIT IN RKDS FOR EVERY
1868                                    ;DRIVE.  IF 'DRY' IS SET DRIVE IS SAID TO BE 'ON LINE', OTHERWISE IT
1869                                    ;IS OFFLINE.  IF THE 'DRY' BIT HAS CHANGED FROM LAST TIME, THEN
1870                                    ;IT IS REPORTED.  IF THERE IS NO CHANGE NOTHING IS REPORTED.
1871
1872  017276  012700  020470    BEGCT1: MOV     #DRIV0,R0       ;INITIALIZE POINTERS
1873  017302  005001                            CLR     R1
1874
1875  017304  011077  162066    BEGCT1: MOV     (R0),@RKDA      ;ADDRESS A DRIVE
1876  017310  042777  017777  162060            BIC     #17777,@RKDA    ;MASK OUT NON DR# BITS
1877  017316  105777  162042                    TSTB    @RKDS           ;IS THIS DRIVE ON LINE?
1878  017322  100044                            BPL     1$              ;NO
1879                                                                    ;YES
1880  017324  105710                            TSTB    (R0)            ;WAS IT 'ON LINE' LAST TIME?
1881  017326  100454                            BMI     NXT1            ;YES, NO MESSAGE TO REPORT
1882  017330  052710  000200                    BIS     #BIT7,(R0)      ;IT CHANGED FROM OFF LINE TO ON
```

```
1893  017334  104401  020324          TYPE    ,EM1            ;LINE, REPORT MESSAGE
1894  017340  010146                  MOV     R1,-(SP)
1895  017342  104402                  TYPOC
1896  017344  104401  020335          TYPE    ,EM2            ;TYPE 'ON LINE'
1897  017350  032777  000040  162006  BIT     #WPS,@RKDS      ;WRITE ENABLED?
1898  017356  001417                  BEQ     2$              ;YES, OK
1899  017360  104401  017366          TYPE    ,65$            ;;TYPE ASCIZ STRING
1900  017364  000414                  BR      64$             ;;GET OVER THE ASCIZ
1901                          ;;65$:  .ASCIZ  <15><12>/EROR,NOT WRT ENABLED/
1902  017416                  64$:
1903  017416  012777  000017  161744  2$:     MOV     #17,@RKCS       ;WRITE PROT THE DISK
1904  017424  105777  161740          3$:     TSTB    @RKCS
1905  017430  100375                  BPL     3$
1906  017432  000412                  BR      NXT1
1907
1908  017434  105710          1$:     TSTB    (R0)            ;WAS THIS DRIVE OFF LINE LAST
1909  017436  100010                  BPL     NXT1            ;TIME?  BRNCH IF YES
1910  017440  104401  020324          TYPE    ,EM1            ;IF NOT, REPORT THE CHANGE
1911  017444  010146                  MOV     R1,-(SP)        ;TYPE DRIVE #
1912  017446  104402                  TYPOC
1913  017450  104401  020347          TYPE    ,EM3            ;TYPE 'OFF LINE'
1914  017454  042710  000200          BIC     #BIT7,(R0)      ;CLEAR BIT TO INDICATE THIS
1915                                                          ;DRIVE 'OFF LINE'
1916
1917                          ;THIS CODE CHECKS 'WPS' BIT FOR EVERY DRIVE THAT IS IN 'DRY'
1918                          ;CONDITION (ON LINE).  IT REPORTS ANY CHANGE IN THE CONDITION OF
1919                          ;THE 'WPS' BIT.  IF THERE WAS NO CHANGE FROM LAST TIME NOTHING
1920                          ;IS REPORTED.  AT THE TIME OF ENTRY R0 POINTS TO DRIVE FLAG.
1921
1922  017460  105777  161700  NXT1:   TSTB    @RKDS           ;IS THIS DRIVE PRESENT?
1923                                                          ;RKDA CONTAINS THE DRV #
1924  017464  100033                  BPL     NXT2            ;NO, SKIP CHECKING
1925
1926  017466  032777  000040  161670  BIT     #WPS,@RKDS      ;WPS BIT SET?
1927  017474  001014                  BNE     1$              ;YES
1928                                                          ;WPS BIT CLEAR
1929  017476  032710  000004          BIT     #BIT2,(R0)      ;WAS IT CLR LAST TIME ALSO?
1930  017502  001424                  BEQ     NXT2            ;YES, NOTHING TO REPORT.
1931                                                          ;WPS CHANGED FROM 'SET'
1932  017504  104401  020324          TYPE    ,EM1            ;TO 'CLR', REPORT IT
1933  017510  010146                  MOV     R1,-(SP)        ;TYPE DRIVE #
1934  017512  104402                  TYPOC
1935  017514  042710  000004          BIC     #BIT2,(R0)      ;INDICATE THAT 'WPS' IS CLEAR
1936  017520  104401  020375          TYPE    ,EM5            ;TYPE 'WPS CLEAR'
1937  017524  000413                  BR      NXT2
1938                                                          ;WPS BIT IS SET
1939  017526  032710  000004          1$:     BIT     #BIT2,(R0)      ;WAS IT SET LAST TIME ALSO?
1940  017532  001010                  BNE     NXT2            ;YES, NOTHING TO REPORT.
1941                                                          ;WPS CHANGED, FROM 'CLR' TO
1942                                                          ;'SET', REPORT THIS CHANGE
1943  017534  104401  020324          TYPE    ,EM1            ;TYPE 'DRIVE'
1944  017540  010146                  MOV     R1,-(SP)        ;TYPE DRIVE #
1945  017542  104402                  TYPOC
1946  017544  104401  020362          TYPE    ,EM4            ;TYPE 'WPS SET'
1947  017550  052710  000004          BIS     #BIT2,(R0)      ;SET FLAG BIT INDICATING WPS SET
1948
```



```
1949
1950                          ;THIS CODE PERFORMS A SEEK FUNCTION ON A DRIVE AND CHECKS IF
1951                          ;THE 'DPL' BIT SET AS A RESULT, (IF THE POWER WAS CUT OFF
1952                          ;FROM THE DRIVE).  NOTE THAT ONLY THOSE DRIVES ARE
1953                          ;CHECKED WHICH WERE FOUND TO BE PRESENT AT BEGINNING (WHEN
1954                          ;THIS TEST WAS ENTERED).  SEEK IS DONE TO CYLINDER 1.
1955                          ;AT THE TIME OF ENTRY R0 POINTS TO THE DRIVE FLAG.
1956
1957  017554  032710  000100  NXT2:   BIT     #BIT6,(R0)      ;WAS THIS DRIVE PRESENT AT BEGNG
1958  017560  001403                  BEQ     4$              ;NO
1959  017562  105777  161576          TSTB    @RKDS           ;IS IT PRESENT NOW?
1960  017566  100402                  BMI     3$              ;YES
1961  017570  000137  020226  4$:     JMP     DN1DRV          ;IF NOT SKIP THIS CHECK
1962
1963  017574  052777  000040  161574  3$:     BIS     #40,@RKDA       ;RKDA ALREADY HAS THE DRV #
1964                                                          ;SET CYL 1 ADDRESS
1965  017602  012777  000011  161560          MOV     #11,@RKCS       ;SEEK, GO
1966
1967  017610  105777  161554          1$:     TSTB    @RKCS           ;WAIT FOR CONTROL RDY?
1968  017614  100375                  BPL     1$              ;SOMETHING WRONG IF CNTAL RDY
1969                                                          ;DOES NOT COME BACK
1970  017616  032777  010000  161540          BIT     #DPL,@RKDS      ;DPL BIT SET?
1971  017624  001414                  BEQ     2$              ;NO
1972                                                          ;YES, DPL SET
1973  017626  032710  000001          BIT     #BIT0,(R0)      ;WAS 'DPL' SET LAST TIME ALSO?
1974  017632  001167                  BNE     CLRDPL          ;YES, NOTHING TO REPORT.
1975                                                          ;DPL CHANGED, GOT SET THIS
1976  017634  104401  020324          TYPE    ,EM1            ;TIME, REPORT IT
1977  017640  010146                  MOV     R1,-(SP)
1978  017642  104402                  TYPOC
1979  017644  104401  020413          TYPE    ,EM6            ;TYPE 'POWER LO'
1980  017650  052710  000001          BIS     #BIT0,(R0)      ;SET FLAG BIT INDICATING THAT
1981                                                          ;DPL SET THIS TIME
1982  017654  000556                  BR      CLRDPL
1983                                                          ;'DPL' BIT IS CLEAR
1984  017656  032710  000001  2$:     BIT     #BIT0,(R0)      ;WAS 'DPL' CLEAR LAST TIME ALSO?
1985  017662  001410                  BEQ     WATSK           ;YES, NOTHING TO REPORT
1986                                                          ;REPORT THAT 'DPL' BIT CHANGED,
1987  017664  104401  020324          TYPE    ,EM1            ;FROM SET TO CLEAR
1988  017670  010146                  MOV     R1,-(SP)
1989  017672  104402                  TYPOC
1990  017674  104401  020434          TYPE    ,EM7            ;TYPE 'POWER UP'
1991  017700  042710  000001          BIC     #BIT0,(R0)      ;SET FLAG BIT INDICATING THAT DPL
1992                                                          ;IS CLEAR THIS TIME
1993
1994                          ;THIS CODE WAITS FOR THE SEEK (DONE ABOVE) TO FINISH.  WAITING
1995                          ;TIME IS APPROX. 50 MS (FOR THE WORST CASE).  IF R/W/S RDY
1996                          ;DOES NOT SET WITHIN 50 MS, THEN IT IS ASSUMED THAT A 'SIN'
1997                          ;IS POSSIBLE AND THE PROGRAM WAITS FOR 1450 MS MORE, SO THAT
1998                          ;THE 'SIN' CAN SET.  IF 'SIN' DOES NOT SET WITHIN THIS
1999                          ;TIME AN ERROR IS REPORTED:
2000                          ;       SIN DIDN'T OCCUR
2001                          ;IF R/W/S RDY SETS WITHIN 50 MS THE PROGRAM PROCEEDS TO
2002                          ;CHECK THE NEXT DRIVE.
2003
2004  017704  012705  164220  WATSK:  MOV     #-6000.,R5      ;SET COUNT TO WAIT FOR
2005                                                          ;50 MS
```

```
                                        ;R/W/S, RDY SET?
                        BNE     3$      ;YES
                        INC     R5      ;WAIT
                        BNE     1$
                                        ;50 MS OVER, R/W/S RDY
                                        ;DIDN'T SET, WAIT FOR
                                        ;SIN TO SET.
                        CLR     R4
                        MOV     #177777,R5      ;SET UP COUNT
2$:                     BIT     #SIN,@RKDS      ;SIN SET?
                        BNE     SINST           ;YES
                        DEC     R5              ;WAIT
                        BNE     2$
                        TST     R4
                        BNE     4$
                        INC     R4
                        BR      2$
                                        ;1500 MS ELAPSED, BUT SIN
                                        ;DIDN'T SET, ERROR!
4$:
                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                        BR      64$             ;;GET OVER THE ASCIZ
;;65$:                  .ASCIZ  <15><12>/SIN DIDN'T OCCUR, DRVE/
64$:
                        MOV     R1,-(SP)                ;TYPE DRIVE #
                        TYPOC
                        BR      DN1DRV
3$:                     BIT     #BIT1,(R0)      ;DID R/W/S RDY SET LAST TIME?
                        BEQ     DN1DRV          ;YES
                        BIC     #BIT1,(R0)      ;CLR FLAG INDICATING THAT SEEK IS OK
                        TYPE    ,EM1            ;REPORT THAT SEEK IS OK
                        MOV     R1,-(SP)
                        TYPOC
                        TYPE    ,EM9
                        BR      DN1DRV

;IF SIN SET, DO DRIVE RESET AND CLEAR IT

SINST:  BIT     #BIT1,(R0)      ;DID 'SIN' SET LAST TIME ALSO?
                        BNE     4$              ;YES, NOTHING TO REPORT
                        BIS     #BIT1,(R0)      ;SET FLAG INDICATING THAT
                        TYPE    ,EM1            ;'SIN' SET, AND REPORT THE CHANGE
                        MOV     R1,-(SP)
                        TYPOC
                        TYPE    ,EM8            ;TYPE 'SIN'

4$:     MOV     @RKDA,R5        ;SAVE RKDA
                        MOV     #1,@RKCS        ;DO CONTROL RESET
1$:     TSTB    @RKCS           ;WAIT FOR CONTROL RDY
                        BPL     1$
                        MOV     R5,@RKDA
                        MOV     #15,@RKCS       ;DO DRIVE RESET, RKDA
                                        ;ALREADY HAS THE DRVE #
2$:     TSTB    @RKCS           ;WAIT FOR CNTRL RDY
                        BPL     2$
                        CLR     R5
```



```
3$:     BIT     #RWS,@RKDS      ;R/W/S SET?
                        BNE     DN1DRV          ;YES
                        INC     R5
                        BNE     3$              ;WAIT FOR R/W/S RDY
                                        ;REPORT ERROR, R/W/S RDY CLR
                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                        BR      64$             ;;GET OVER THE ASCIZ
;;65$:                  .ASCIZ  <15><12>/RWS RDY NOT SET/
64$:

                        BR      DN1DRV

;IF DPL SET CLEAR THE ERROR BY DOING CONTROL RESET.

CLRDPL: MOV     #1,@RKCS        ;CONTROL RESET
1$:     TSTB    @RKCS           ;WAIT FOR CNTRL RDY
                        BPL     1$
;AT THIS STAGE THE DRIVE (# IN RKDA) HAS BEEN CHECKED
;FOR DRY, WPS, DPL, & SIN. THE POINTERS ARE INCREMENTED
;AND THE SAME CHECKS WILL BE DONE ON THE NEXT
;DRIVE & THEN THE NEXT ONE & SO ON. NOTE THAT
;THIS SUB-PROGRAM KEEPS ON CYCLING THROUGH
;ALL THE DRIVES, AT THE TIME OF ENTRY (HERE)
;R0 POINTS TO THE FLAG FOR THE DRIVE THAT WAS
;JUST CHECKED, BEFORE GOING ON TO THE NEXT
;DRIVE THE HEADS ARE BROUGHT BACK TO CYLINDER
;0 (FOR THE NEXT CYCLE).

DN1DRV: MOV     (R0),@RKDA      ;GET DRIVE #
                        TSTB    @RKDS           ;DRIVE PRESENT?
                        BPL     3$              ;NO
                        BIC     #17777,@RKDA    ;CYL ADRES = 0
                        MOV     #11,@RKCS       ;GO, SEEK

1$:     TSTB    @RKCS           ;WAIT FOR CNTRL RDY
                        BPL     1$

                        JSR     PC,SMTME
4$:     BIT     #RWS,@RKDS
                        BEQ     4$

3$:     TST     (R0)+           ;INCREMENT POINTERS TO
                        INC     R1              ;NEXT DRIVE
                        CMP     R1,#10          ;ALL DONE, THIS CYCLE?
                        BEQ     2$              ;YES
                        JMP     BEGCT1          ;GO DO NEXT DRIVE
2$:     JMP     BEGCT           ;RESTART THE CYCLE OVER
                                        ;AGAIN


SHFCNT: .WORD   0
DRVCNT: .WORD   0
;MESSAGES
EM1:    .ASCIZ  <15><12>/DRIVE /
```

```
2107  020335  040  047440  020116  EM2:    .ASCIZ  /  ON LINE/
2108  020342  044514  042516  000
2109  020347  040  047440  043106  EM3:    .ASCIZ  /  OFF LINE/
2110  020354  046040  047111  000105
2111  020362  020040  051127  020124  EM4:    .ASCIZ  /  WRT PROT/
2112  020370  051120  052117  000
2113  020375  040  053440  052122  EM5:    .ASCIZ  /  WRT ENABLED/
2114  020402  042440  040516  046102
2115  020410  042105  000
2116  020413  040  042040  044522  EM6:    .ASCIZ  /  DRIVE POWER LO/
2117  020420  042526  050040  053517
2118  020426  051105  046040  000117
2119  020434  020040  047520  042527  EM7:    .ASCIZ  /  POWER UP/
2120  020442  020122  050125  000
2121  020447  040  051440  047111  EM8:    .ASCIZ  /  SIN/
2122  020454  000
2123  020455  040  051440  042505  EM9:    .ASCIZ  /  SEEK OK/
2124  020462  020113  045517  000
2125
2126                                  ;DRIVE FLAGS FOR CONTROL PANEL TEST #2
2127                                  ;BITS 15,14,13 GIVE THE DRIVE NO (EX: 0,1,2,---)
2128                                  ;BIT 7 IS SET WHEN 'DRY' BIT IS SET FOR THE DRIVE (ON LINE)
2129                                  ;BIT 7 IS CLEAR WHEN 'DRY' IS CLEAR (WHEN DRIVE IS IN LOAD/OFF LINE,
2130                                  ;DRIVE POWER IS CUT OFF)
2131                                  ;BIT 6 IS SET IF A DRIVE IS FOUND TO BE PRESENT (DRY) AT
2132                                  ;THE BEGINING, UNLIKE BIT 7 THIS BIT DOES NOT GET SET OR
2133                                  ;CLEARED AS THE DRIVE CONDITIONS CHANGE, IT JUST INDICATES
2134                                  ;THAT THE DRIVE IS AVAILABLE FOR CHECKING.
2135                                  ;BIT 0 IS SET IF 'DPL' BIT GETS SET, IE: DRIVE POWER OFF
2136                                  ;BIT 0 IS CLEARED WHEN DRIVE POWER IS ON.
2137                                  ;BIT 1 IS SET WHEN SEEK INCOMPLETE 'SIN' OCCURS.
2138                                  ;BIT 1 IS CLEAR WHEN SEEK IS OK.
2139                                  ;BIT 2 IS SET WHEN WRT PROT IS SET FROM CONSOLE.
2140                                  ;BIT 2 IS CLEARED WHEN DRIVE IS WRITE ENABLED.
2141
2142          020470                      .EVEN
2143  020470  000000                  DRIV0:  .WORD   0               ;DRIVE FLAGS
2144  020472  000000                  DRIV1:  .WORD   0
2145  020474  000000                  DRIV2:  .WORD   0
2146  020476  000000                  DRIV3:  .WORD   0
2147  020500  000000                  DRIV4:  .WORD   0
2148  020502  000000                  DRIV5:  .WORD   0
2149  020504  000000                  DRIV6:  .WORD   0
2150  020506  000000                  DRIV7:  .WORD   0
2151
2152
```



```
2153                                        .SBTTL  HEAD ALIGNMENT ROUTINE
2154
2155                                  ;HEAD ALIGNMENT ROUTINE
2156                                  ;THIS MAINTAINANCE ROUTINE IS HELPFUL IN HEAD ALIGNMENT. UPON ENTRY
2157                                  ;THE QUESTION - DRIVE? - IS ASKED , THE USER SHOULD REPLY WITH THE
2158                                  ;DRIVE NUMBER THAT IS TO BE ALIGNED. IF THE DRIVE IS AN RK-05F
2159                                  ;THE LETTER 'F' IS ADDED AS A SUFFIX, FOR SELECTING SURFACE 0
2160                                  ;PUT SW0=0, FOR SELECTING SURFACE 1 PUT SW0=1. SET SW1 =1 TO SELECT
2161                                  ;CYLINDER 64.  SET SW1=0 TO SELECT CYLINDER 105.
2162                                  ;IF THE DRIVE IS AN RK-05F, CYLINDER 64 BECOMES CYLINDER 130
2163                                  ;OF THE EVEN DRIVE, AND CYLINDER 105 BECOMES CYLINDER 5 OF THE ODD DRIVE
2164                                  ;THE HEADS ARE PLACED ON THE SELECTED CYLINDER AND DATA IS READ
2165                                  ;CONTINUOSLY FROM THE CYLINDER (SECTOR 0)
2166                                  ; THE UPPER OR LOWER HEAD AND CYLINDER CAN BE SELECTED
2167                                  ;DYNAMICALLY, IE. THE PROGRAM DOES NOT HAVE TO BE STOPPED TO SELECT THE
2168                                  ;UPPER OR LOWER HEAD OR CYLINDER.
2169                                  ;IN ORDER TO SELECT ANOTHER DRIVE, PUT ANY SWITCH FROM SW2 TO SW15 UP AND
2170                                  ;THE PROGRAM WILL AGAIN ASK THE QUESTION (DRIVE?).
2171
2172  020510  000240                  SECT.5: NOP
2173  020512  104401  020520                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2174  020516  000426                          BR      64$             ;;GET OVER THE ASCIZ
2175                                  ;;65$:  .ASCIZ  <15><12>/SET SW0=0 FOR SURFACE 0, SW0=1 FOR SUR 1./
2176  020574                          64$:
2177  020574  104401  020602                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2178  020600  000432                          BR      66$             ;;GET OVER THE ASCIZ
2179                                  ;;67$:  .ASCIZ  <15><12>/SET SW1=0 FOR CYLINDER 105, SW1=1 FOR CYLINDER 64/
2180  020666                          66$:
2181  020666  104401  020674                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2182  020672  000427                          BR      68$             ;;GET OVER THE ASCIZ
2183                                  ;;69$:  .ASCIZ  <15><12>/PUT ANY SW FROM 2-15 HI TO SELECT NEW DRVE/
2184  020752                          68$:
2185  020752  104401  021370          HDALGN: TYPE    ,EM10                   ;ASK FOR DRIVE #
2186  020756  005037  021364                  CLR     FFLAG                   ;FLAG FOR RK-05F
2187  020762  104410                          RDLIN                           ;GET OPR INPUT
2188  020764  012601                          MOV     (SP)+,R1                ;ADDR OF COMMAND STRING
2189  020766  112100                          MOVB    (R1)+,R0                ;FIRST CHAR
2190  020770  162700  000060                  SUB     #60,R0                  ;0 TO 7
2191  020774  002766                          BLT     HDALGN                  ;TOO SMALL
2192  020776  022700  000067                  CMP     #67,R0                  ;MUST BE 7 OR LESS
2193  021002  002763                          BLT     HDALGN                  ;TOO BIG
2194  021004  000241                          CLC
2195  021006  006000                          ROR     R0
2196  021010  006000                          ROR     R0
2197  021012  006000                          ROR     R0
2198  021014  006000                          ROR     R0
2199  021016  010037  020470                  MOV     R0,DRIV0                ;ADDRESS OF DRIVE
2200  021022  112100                          MOVB    (R1)+,R0                ;NEXT INPUT CHAR
2201  021024  001412                          BEQ     5$                      ;ALL DONE IF C.R.
2202  021026  020027  000106                  CMP     R0,#'F                  ;IS IT F?
2203  021032  001347                          BNE     HDALGN                  ;NO, SO ERROR
2204  021034  105711                          TSTB    (R1)                    ;NEXT CHAR MUST BE C.R.
2205  021036  001345                          BNE     HDALGN                  ;ELSE, ERROR
2206  021040  042737  020000  020470          BIC     #BIT13,DRIV0            ;USE EVEN DRIVE IF RK-05F
2207  021046  005237  021364                  INC     FFLAG                   ;SHOW F TYPE DRIVE
2208  021052  013700  020470          5$:     MOV     DRIV0,R0                ;DRIVE ADDR TO R0
```

```
2209  021056  017737  160056  021366          MOV     @SWR,SWTCH              ;HOLD SWITCHES
2210  021064  042737  177775  021366          BIC     #177775,SWTCH           ;WANT SW1 ONLY
2211  021072  001005                          BNE     7$                      ;SW1 SET, SO LOW CYLINDER
2212  021074  005737  021364                  TST     FFLAG                   ;F DRIVE?
2213  021100  001402                          BEQ     7$                      ;NO
2214  021102  052700  020000                  BIS     #BIT13,R0               ;ODD DRIVE IF HIGH TRACK OF F
2215  021106  010077  160264          7$:     MOV     R0,@RKDA                ;ADDRESS DRIVE
2216  021112  012777  000017  160250          MOV     #17,@RKCS               ;WRITE PROTECT
2217  021120  105777  160244          8$:     TSTB    @RKCS
2218  021124  100375                          BPL     8$                      ;WAIT FOR DRIVE READY
2219  021126  012777  000001  160234          MOV     #1,@RKCS                ;RESET CONTROLLER
2220  021134  105777  160230          9$:     TSTB    @RKCS
2221  021140  100375                          BPL     9$                      ;WAIT FOR READY
2222  021142  005737  021364                  TST     FFLAG                   ;F DRIVE?
2223  021146  001410                          BEQ     13$                     ;NO
2224  021150  012701  000240                  MOV     #5.*40,R1               ;TRACK 5 OF HIGH
2225  021154  005737  021366                  TST     SWTCH                   ;SW1 SET?
2226  021160  001412                          BEQ     10$                     ;YES SO TEST TRACK 8 OF DRIVE HIGH
2227  021162  062701  010100                  ADD     #130.*40,R1             ;TRACK 130. IF SW1 SET
2228  021166  000407                          BR      10$
2229  021170  012701  004000          13$:    MOV     #64.*40,R1              ;CYLINDER 64 IF NOT F
2230  021174  005737  021366                  TST     SWTCH                   ;SW1 SET?
2231  021200  001002                          BNE     10$                     ;YES, SO CYLINDER 64
2232  021202  062701  002440                  ADD     #41.*40,R1              ;CYLINDER 105
2233  021206  005777  160156          10$:    TST     @RKCS                   ;ANY ERROR?
2234  021212  100006                          BPL     11$                     ;NO, CONTINUE
2235  021214  012777  000001  160146          MOV     #1,@RKCS                ;RESET
2236  021222  105777  160142          12$:    TSTB    @RKCS
2237  021226  100375                          BPL     12$                     ;WAIT FOR READY
2238  021230  017702  157704          11$:    MOV     @SWR,R2                 ;SWITCH REG TO R2
2239  021234  042702  177775                  BIC     #177775,R2              ;SW1 ONLY
2240  021240  020237  021366                  CMP     R2,SWTCH                ;ANY CHANGE SINCE LAST?
2241  021244  001302                          BNE     5$                      ;YES, GO SET-UP ADDR AGAIN
2242  021246  010077  160124          6$:     MOV     R0,@RKDA                ;ADDRESS THE DRIVE
2243  021252  012777  000017  160110          MOV     #17,@RKCS               ;WRITE  PROTECT THE DRIVE
2244  021260  105777  160104                  TSTB    @RKCS                   ;WAIT FOR CONTROL RDY
2245  021264  100375                          BPL     .-4
2246  021266  042700  000020          4$:     BIC     #20,R0                  ;CLEAR TRACK ADDR
2247  021272  032777  000001  157640          BIT     #1,@SWR                 ;SW0 SET?
2248  021300  001402                          BEQ     2$                      ;NO TEST TRACK 0
2249  021302  052700  000020                  BIS     #20,R0                  ;TEST TRACK 1
2250  021306  042700  017740          2$:     BIC     #17740,R0               ;CLEAR CYLINDER ADDR
2251  021312  050100                          BIS     R1,R0                   ;PUT CYLINDER ADDR IN ADDR
2252  021314  010077  160056                  MOV     R0,@RKDA                ;ADRES THE DRIVE
2253  021320  012777  177400  160044          MOV     #-400,@RKWC             ;READ 1 SECTOR
2254  021326  012777  007336  160040          MOV     #RDBUFF,@RKBA           ;INTO THIS BUFFER
2255  021334  012777  000005  160026          MOV     #5,@RKCS                ;READ, GO
2256
2257  021342  105777  160022          3$:     TSTB    @RKCS                   ;DONE?
2258  021346  100375                          BPL     3$                      ;NO
2259
2260  021350  032777  177774  157562          BIT     #177774,@SWR            ;EXIT OUT?
2261  021356  001713                          BEQ     10$                     ;NO, CONTINUE ON THIS DRIVE
2262  021360  000137  020752                  JMP     HDALGN                  ;YES, GET NEW DRIVE
2263
2264  021364  000000          FFLAG:  0
```



```
2265  021366  000000          SWTCH:  0
2266  021370  005015  051104  053111  EM10:   .ASCIZ  <15><12>/DRIVE?/
2267  021376  037505     000
2268
2269
2270
2271                                          .SBTTL  DISK POWER FAILURE TEST
2272
2273                                  ;(DISK)POWER FAILURE (DURING DISK WRITE) TEST
2274                                  ;THIS TEST CHECKS THAT THE INFORMATION WRITTEN ON THE DISK IS
2275                                  ;NOT DESTROYED WHEN THE DISK SENSES A LOSS OF POWER WHILE DOING A WRITE
2276                                  ;AND RETRACTS THE HEADS. UPON ENTRY THE PROGRAM FINDS OUT THE
2277                                  ;FIRST AVAILABLE DRIVE INDICATES IT (DRIVE XX) AND PROCEEDS TO TEST.
2278                                  ;CYLINDERS 0 TO 15 ARE WRITTEN WITH UNIQUE PATTERNS, THEN THE HEADS
2279                                  ;ARE POSITIONED ON CYLINDER 10 (DECIMAL) AND A MESSAGE (DROP
2280                                  ;POWER) IS GIVEN.  AFTER RECEIVING THIS MESSAGE THE USER SHOULD
2281                                  ;DROP POWER FROM THE DRIVE.  ON SENSING A LOSS OF POWER, THE
2282                                  ;PROGRAM ASKS THE USER TO PUT BACK THE POWER.  THE ERRORS (DPL)
2283                                  ;ARE CLEARED AND A WRITE-CHECK IS PERFORMED TO CHECK IF THE
2284                                  ;UNIQUE PATTERNS ON THE DISK (CYLINDERS 0-9 AND 11-15) ARE STILL
2285                                  ;THERE.  IF NOT A WRITE CHECK ERROR IS REPORTED.
2286          021402                          .EVEN
2287
2288  021402  005000          SECT.6: CLR     R0                      ;INITIALIZE DRIVE #
2289  021404  005037  020470          CLR     DRIV0
2290  021410  010077  157762  8$:     MOV     R0,@RKDA                ;IS IT PRESENT?
2291  021414  105777  157744          TSTB    @RKDS                   ;IF NOT SKIP
2292  021420  100406                  BMI     9$
2293  021422  005237  020470  10$:    INC     DRIV0
2294  021426  062700  020000          ADD     #20000,R0
2295  021432  001366                  BNE     8$
2296  021434  000762                  BR      SECT.6
2297  021436  104401  020324  9$:     TYPE    .EM1
2298  021442  013746  020470          MOV     DRIV0,-(SP)             ;GET DRIVE #
2299  021446  104402                  TYPOC
2300  021450  012777  000001  157712  1$:     MOV     #1,@RKCS                ;CONTROL RESET
2301  021456  105777  157706          TSTB    @RKCS
2302  021462  100375                  BPL     .-4
2303  021464  005005                  CLR     R5                      ;INITIALIZE PATTERN TO BE WRITTEN
2304                                                                  ;0 ON CYL 0, 1 ON CYL 1, ETC
2305  021466  013701  001376          MOV     RKDA,R1
2306  021472  013702  001372          MOV     RKWC,R2
2307  021476  013703  001374          MOV     RKBA,R3
2308  021502  013704  001370          MOV     RKCS,R4
2309  021506  010011                  MOV     R0,@R1
2310  021510  010537  022046          MOV     R5,BUFR                 ;FILL THE PATTER IN DATA BUFFER.
2311  021514  012713  022046          MOV     #BUFR,@R3               ;BUS ADRES
2312  021520  012712  164000          MOV     #-14000,@R2             ;WRITE 1 CYL (256X12X2 WORDS)
2313  021524  012714  004003          MOV     #4003,@R4               ;WRITE, GO, IBA SET
2314  021530  105714                  TSTB    @R4                     ;WAIT FOR CONTROL READY
2315  021532  100376                  BPL     .-2
2316                                                                  ;DONE
2317  021534  005205                  INC     R5                      ;WRITTEN ALL 15 CYLINDERS?
2318  021536  020527  000020          CMP     R5,#20                  ;IF NOT, GO BAK
2319  021542  001362                  BNE     1$
2320
```

```
2321  021544  010005                          MOV     R0,R5           ;DRIVE #
2322  021546  052705  000500                  BIS     #500,R5         ;CYL 10
2323  021552  012737  000012  022046          MOV     #12,BUFR        ;PATERN TO BE WRITTEN
2324  021560  010511                          MOV     R5,@R1          ;ADRES THE DISK
2325  021562  012712  164000                  MOV     #-14000,@R2     ;WORD COUNT= 1 CYLINDER
2326  021566  012713  022046                  MOV     #BUFR,@R3       ;BUS ADRES
2327  021572  012714  004003                  MOV     #4003,@R4       ;WRITE, GO, IBA
2328  021576  032777  000100  157560  2$:     BIT     #RWS,@RKDS      ;WAIT FOR THE HEADS TO SETTLE
2329  021604  001774                          BEQ     2$              ;ON CYL 10
2330  021606  104401  021614                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2331  021612  000406                          BR      64$             ;;GET OVER THE ASCIZ
2332                                  ;;65$:  .ASCIZ  /DROP POWER/
2333  021630                          64$:
2334
2335  021630  000407                          BR      5$
2336  021632  010511                  3$:     MOV     R5,@R1          ;ADRES THE DISK
2337  021634  012712  164000                  MOV     #-14000,@R2     ;WORD COUNT= 1 CYLINDER
2338  021640  012713  022046                  MOV     #BUFR,@R3       ;BUS ADRES
2339  021644  012714  004003                  MOV     #4003,@R4       ;WRITE, GO, IBA
2340  021650  105714                  5$:     TSTB    @R4             ;WAIT FOR CONTROL READY
2341  021652  100376                          BPL     .-2
2342  021654  005714                          TST     @R4             ;WAIT FOR DRIVE POWER TO GO DOWN,
2343                                                                  ;OTHERWISE, KEEP ON WRITING ON CYL 10
2344  021656  100365                          BPL     3$
2345                                                                  ;IF DRIVE POWER LOSS WAS SENSED,
2346                                                                  ;ASK TO PUT POWER ON.
2347  021660  104401  021666                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2348  021664  000406                          BR      66$             ;;GET OVER THE ASCIZ
2349                                  ;;67$:  .ASCIZ  <15><12>/POWER ON/
2350  021702                          66$:
2351  021702  105777  157456                  TSTB    @RKDS           ;WAIT FOR DRIVE READY
2352  021706  100375                          BPL     .-4
2353  021710  012714  000001                  MOV     #1,@R4          ;CONTROL RESET, CLEAR ERROR
2354  021714  105714                          TSTB    @R4
2355  021716  100376                          BPL     .-2
2356  021720  010077  157452                  MOV     R0,@RKDA
2357  021724  005005                          CLR     R5              ;INITIALIZE PATTERN
2358  021726  010537  022046          6$:     MOV     R5,BUFR
2359  021732  012713  022046                  MOV     #BUFR,@R3
2360  021736  012712  164000                  MOV     #-14000,@R2
2361  021742  012714  004007                  MOV     #4007,@R4       ;WRITE CHECK, GO, IBA
2362  021746  105714                          TSTB    @R4
2363  021750  100376                          BPL     .-2
2364
2365  021752  005714                          TST     @R4             ;ANY ERROR?
2366  021754  100023                          BPL     7$              ;NO
2367  021756  104401  021764                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2368  021762  000416                          BR      68$             ;;GET OVER THE ASCIZ
2369                                  ;;69$:  .ASCIZ  <15><12>/ERROR, ON PWR-UP,  RKDA=/
2370  022020                          68$:
2371  022020  011146                          MOV     @R1,-(SP)
2372  022022  104402                          TYPOC
2373  022024  005205                  7$:     INC     R5
2374  022026  020527  000012                  CMP     R5,#12          ;CYLINDER 10 ?
2375  022032  001774                          BEQ     7$              ;BR IF YES, INCREMENT AGAIN
2376  022034  020527  000020                  CMP     R5,#20
```



```
2377  022040  001332                          BNE     6$
2378
2379  022042  000137  021422                  JMP     10$
2380
2381  022046  000000                  BUFR:   .WORD   0
2382
2383
2384                                  .SBTTL  TYPE ROUTINE
2385
2386                                  ;!*************************************************************
2387                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2388                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2389                                  ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2390                                  ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2391                                  ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2392                                  ;*
2393                                  ;*CALL:
2394                                  ;*1) USING A TRAP INSTRUCTION
2395                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2396                                  ;*OR
2397                                  ;*      TYPE
2398                                  ;*      MESADR
2399                                  ;*
2400
2401  022050  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
2402  022054  100002                          BPL     1$              ;;BR IF YES
2403  022056  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
2404  022060  000407                          BR      3$              ;;LEAVE
2405  022062  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
2406  022064  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
2407  022070  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2408  022072  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
2409  022074  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
2410  022076  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
2411  022100  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
2412  022104  000002                          RTI                     ;;RETURN
2413  022106  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
2414  022112  001430                          BEQ     8$
2415  022114  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
2416  022120  001006                          BNE     5$
2417  022122  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
2418  022124  104401                          TYPE                    ;;TYPE A CR AND LF
2419  022126  001161                          $CRLF
2420  022130  105037  022264                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
2421  022134  000755                          BR      2$              ;;GET NEXT CHARACTER
2422  022136  004737  022220          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
2423  022142  123726  001156          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
2424  022146  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
2425  022150  013746  001154                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
2426                                                                  ;;AND THE NULL CHAR.
2427  022154  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2428  022160  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2429  022162  004737  022220                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
2430  022166  105337  022264                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2431  022172  000770                          BR      7$              ;;LOOP
2432
```

```
                                        ;HORIZONTAL TAB PROCESSOR
                                        
022174  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
022200  004737  022220          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
022204  132737  000007  022264          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
022212  001372                          BNE     9$              ;;TAB STOP
022214  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
022216  000724                          BR      2$              ;;GET NEXT CHARACTER
022220  105777  156724          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
022224  100375                          BPL     $TYPEC
022226  116677  000002  156716          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
022234  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
022242  001003                          BNE     1$              ;;BRANCH IF NO
022244  105037  022264                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
022250  000406                          BR      $TYPEX          ;;EXIT
022252  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
022260  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
022262  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
022264  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
022266  000207          $TYPEX: RTS     PC
                                        
                                        .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
                                        
022270  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
022274  116637  000001  022513          MOVB    1(SP),$0FILL    ;;LOAD ZERO FILL SWITCH
022302  112637  022515                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
022306  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
022312  000406                          BR      $TYPON
022314  112737  000001  022513  $TYPOC: MOVB    #1,$0FILL       ;;SET THE ZERO FILL SWITCH
022322  112737  000006  022515          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
022330  112737  000005  022512  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
022336  010346                          MOV     R3,-(SP)        ;;SAVE R3
022340  010446                          MOV     R4,-(SP)        ;;SAVE R4
```



```
022342  010546                          MOV     R5,-(SP)        ;;SAVE R5
022344  113704  022515                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
022350  005404                          NEG     R4
022352  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
022356  110437  022514                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
022362  113704  022513                  MOVB    $0FILL,R4       ;;GET THE ZERO FILL SWITCH
022366  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
022372  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
022374  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
022376  000404                          BR      3$              ;;GO DO MSB
022400  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
022402  006105                          ROL     R5
022404  006105                          ROL     R5
022406  010503                          MOV     R5,R3
022410  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
022412  105337  022514                  DECB    $OMODE          ;;TYPE THIS DIGIT?
022416  100016                          BPL     7$              ;;BR IF NO
022420  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
022424  001002                          BNE     4$              ;;TEST FOR 0
022426  005704                          TST     R4              ;;SUPPRESS THIS 0?
022430  001403                          BEQ     5$              ;;BR IF YES
022432  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
022434  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
022440  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
022444  110337  022510                  MOVB    R3,8$           ;;SAVE FOR TYPING
022450  104401  022510                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
022454  105337  022512          7$:     DECB    $OCNT           ;;COUNT BY 1
022460  003347                          BGT     2$              ;;BR IF MORE TO DO
022462  002402                          BLT     6$              ;;BR IF DONE
022464  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
022466  000744                          BR      2$              ;;GO DO THE LAST DIGIT
022470  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
022472  012604                          MOV     (SP)+,R4        ;;RESTORE R4
022474  012603                          MOV     (SP)+,R3        ;;RESTORE R3
022476  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
022504  012616                          MOV     (SP)+,(SP)
022506  000002                          RTI                     ;;RETURN
022510     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
022511     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
022512     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
022513     000          $0FILL: .BYTE   0               ;;ZERO FILL SWITCH
022514  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
                                        .SBTTL  TTY INPUT ROUTINE
                                        
                                        ;;*****************************************************************
                                        .ENABL  LSB
022516  000000          $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
022520  000000          $TKQIN: .WORD   0               ;;INPUT POINTER
022522  000000          $TKQOUT: .WORD  0               ;;OUTPUT POINTER
022524  000001          $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
        022525          $TKQEND=.
        022526                  .EVEN
                                        
                                        ;*TK INITIALIZE ROUTINE
                                        ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                        ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
```

```
                                        ;
                                        ;*CALL:
                                        ;*      JSR     PC,$TKINT
                                        ;*      RETURN
                                        ;
022526  005037  022516          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
022532  012737  022524  022520          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
022540  013737  022520  022522          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
022546  012737  022576  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
022554  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
022562  005777  156360                  TST     @$TKB           ;;CLEAR DONE FLAG
022566  012777  000100  156350          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
022574  000207                          RTS     PC              ;;RETURN TO CALLER
                                        ;*TK SERVICE ROUTINE
                                        ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                        ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                        ;*IT IN THE QUEUE.
                                        ;
022576  117746  156344          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
022602  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
022606  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
022612  001004                          BNE     2$              ;;BRANCH IF NO
022614  022737  000176  001140          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
022622  001500                          BEQ     6$              ;;GO TO SWR CHANGE
022624                          2$:
022624  022737  000001  022516          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
022632  001004                          BNE     3$              ;;BRANCH IF NO
022634  104401  023646                  TYPE    ,$BELL          ;;RING THE TTY BELL
022640  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
022642  000451                          BR      5$              ;;EXIT
022644  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
022650  001021                          BNE     32$             ;;BRANCH IF NO
022652  005077  156266                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
022656  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
022660  105777  156260          31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
022664  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
022666  117746  156254                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
022672  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
022676  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
022702  001366                          BNE     31$             ;;BRANCH IF NO
022704  012777  000100  156232          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
022712  000002                          RTI                     ;;RETURN
022714  005237  022516          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
022720  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
022724  002405                          BLT     4$              ;;BRANCH IF YES
022726  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
022732  003002                          BGT     4$              ;;BRANCH IF YES
022734  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
022740  112677  177554          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
022744  005237  022520                  INC     $TKQIN          ;;UPDATE THE POINTER
022750  023727  022520  022525          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
022756  001003                          BNE     5$              ;;BRANCH IF NO
022760  012737  022524  022520          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
022766  000002                  5$:     RTI                     ;;RETURN
```



```
                                        ;;**************************************************************
                                        ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                        ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                                        ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
                                        ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
022770  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
022776  001104                          BNE     15$             ;;EXIT IF NOT
023000  105777  156140                  TSTB    @$TKS           ;;IS A CHAR WAITING?
023004  100101                          BPL     15$             ;;IF NOT, EXIT
023006  117746  156134                  MOVB    @$TKB,-(SP)     ;;YES
023012  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
023016  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
023022  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                ;;AND EXIT
                                        ;;**************************************************************
                                        ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                        ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                        ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
023024  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
023032  001674                          BEQ     2$              ;;BRANCH IF YES
023034  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
023036  004737  022526                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
023042  005077  156076                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
023046  112737  000001  001135          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
023054  104401  023657                  TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G ("G)
023060  104401  023664          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
023064  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
023070  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
023072  104401  023675                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
023076  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
023100  005046                          CLR     -(SP)           ;;THE NEW SWR
023102  105777  156036          7$:     TSTB    @$TKS           ;;CHAR THERE?
023106  100375                          BPL     7$              ;;IF NOT TRY AGAIN
023110  117746  156032                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
023114  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
023120  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
023124  001005                          BNE     10$             ;;BRANCH IF NOT
023126  104401  023652                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U ("U)
023132  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
023136  000757                          BR      19$             ;;LET'S TRY IT AGAIN
023140  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
023144  001022                          BNE     16$             ;;BRANCH IF NO
023146  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
023152  001403                          BEQ     11$             ;;BRANCH IF YES
023154  016677  000002  155756          MOV     2(SP),@SWR      ;;SAVE NEW SWR
023162  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
023166  104401  001161          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
```

```
2657  023172  123727  001135  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
2658  023200  001003                          BNE     15$             ;;BRANCH IF NOT
2659  023202  012777  000100  155734          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
2660  023210  000002                  15$:    RTI                     ;;RETURN
2661  023212  004737  022220          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
2662  023216  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
2663  023222  002420                          BLT     18$             ;;BRANCH IF YES
2664  023224  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
2665  023230  003015                          BGT     18$             ;;BRANCH IF YES
2666  023232  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
2667  023236  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
2668  023242  001403                          BEQ     17$             ;;BRANCH IF YES
2669  023244  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
2670  023246  006316                          ASL     (SP)            ;;   CHAR OVER TO MAKE
2671  023250  006316                          ASL     (SP)            ;;   ROOM FOR NEW ONE.
2672  023252  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
2673  023256  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
2674  023262  000707                          BR      7$              ;;GET THE NEXT ONE
2675  023264  104401  001160          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
2676  023270  000720                          BR      20$             ;;SIMULATE CONTROL-U
2677                                  .DSABL  LSB
2678
2679
2680                                  ;;*****************************************************************
2681                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2682                                  ;*CALL:
2683                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
2684                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
2685                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
2686                                  ;
2687
2688  023272  011646          $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
2689  023274  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
2690  023302  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
2691  023306  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
2692  023310  012746  023316                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
2693  023314  000002                          RTI                     ;;POP NEW PC AND PS
2694  023316                          64$:
2695  023316  005737  022516          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
2696  023322  001775                          BEQ     1$
2697  023324  005337  022516                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
2698  023330  117766  177166  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
2699  023336  005237  022522                  INC     $TKQOUT         ;;UPDATE THE POINTER
2700  023342  023727  022522  022525          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
2701  023350  001003                          BNE     2$              ;;BRANCH IF NO
2702  023352  012737  022524  022522          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
2703  023360  000002                  2$:     RTI                     ;;RETURN
2704                                  ;;*****************************************************************
2705                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2706                                  ;*CALL:
2707                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
2708                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2709                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2710
2711  023362  010346          $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
2712  023364  005046                  CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
```



```
2713  023366  012703  023616          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
2714  023372  022703  023646          2$:     CMP     #$TTYIN+30,R3   ;;BUFFER FULL?
2715  023376  101456                          BLOS    4$              ;;BR IF YES
2716  023400  104407                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2717  023402  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
2718  023404  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
2719  023410  001022                          BNE     5$              ;;BR IF NO
2720  023412  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
2721  023414  001007                          BNE     6$              ;;BR IF NO
2722  023416  112737  000134  023614          MOVB    #'\,9$          ;;TYPE A BACK SLASH
2723  023424  104401  023614                  TYPE    ,9$
2724  023430  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
2725  023434  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
2726  023436  020327  023616                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
2727  023442  103434                          BLO     4$              ;;BR IF YES
2728  023444  111337  023614                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
2729  023450  104401  023614                  TYPE    ,9$             ;;GO TYPE
2730  023454  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
2731  023456  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
2732  023460  001406                          BEQ     7$              ;;BR IF NO
2733  023462  112737  000134  023614          MOVB    #'\,9$          ;;TYPE A BACK SLASH
2734  023470  104401  023614                  TYPE    ,9$
2735  023474  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
2736  023476  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
2737  023502  001003                          BNE     8$              ;;BR IF NO
2738  023504  104401  023652                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
2739  023510  000726                          BR      1$              ;;GO START OVER
2740  023512  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "R"?
2741  023516  001011                          BNE     3$              ;;BRANCH IF NO
2742  023520  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
2743  023522  104401  001161                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
2744  023526  104401  023616                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
2745  023532  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
2746  023534  104401  001160          4$:     TYPE    ,$QUES          ;;TYPE A '?'
2747  023540  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
2748  023542  111337  023614          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2749  023546  104401  023614                  TYPE    ,9$
2750  023552  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2751  023556  001305                          BNE     2$              ;;LOOP IF NOT RETURN
2752  023560  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2753  023564  104401  001162                  TYPE    ,$LF            ;;TYPE A LINE FEED
2754  023570  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
2755  023572  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2756  023574  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2757  023576  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2758  023604  012766  023616  000004          MOV     #$TTYIN,4(SP)
2759  023612  000002                          RTI                     ;;RETURN
2760  023614     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2761  023615     000                          .BYTE   0               ;;TERMINATOR
2762  023616  000030          $TTYIN: .BLKB   30              ;;RESERVE 30 BYTES FOR TTY INPUT
2763  023646  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
2764  023652  052536  005015     000  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
2765  023657     136  006507  000012  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
2766  023664  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
2767  023672  020075     000
2768  023675     040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
```

```
2769  023702  036440  000040
2770                          .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
2771
2772                          ;;*****************************************************************
2773                          ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
2774                          ;*CHANGE IT TO BINARY.
2775                          ;*CALL:
2776                          ;*      RDOCT                   ;;READ AN OCTAL NUMBER
2777                          ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
2778                          ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
2779
2780  023706  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
2781  023710  016666  000004  000002          MOV     4(SP),2(SP)     ;;INPUT NUMBER
2782  023716  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2783  023720  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2784  023722  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2785  023724  104410                  1$:     RDLIN                   ;;READ AN ASCIZ LINE
2786  023726  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
2787  023730  005001                  CLR     R1              ;;CLEAR DATA WORD
2788  023732  005002                  CLR     R2
2789  023734  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
2790  023736  001412                  BEQ     3$              ;;IF ZERO GET OUT
2791  023740  006301                  ASL     R1              ;;*2
2792  023742  006102                  ROL     R2
2793  023744  006301                  ASL     R1              ;;*4
2794  023746  006102                  ROL     R2
2795  023750  006301                  ASL     R1              ;;*8
2796  023752  006102                  ROL     R2
2797  023754  042716  177770                  BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
2798  023760  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
2799  023762  000764                  BR      2$              ;;LOOP
2800  023764  005726                  3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
2801  023766  010166  000012                  MOV     R1,12(SP)       ;;SAVE THE RESULT
2802  023772  010237  024006                  MOV     R2,$HIOCT
2803  023776  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
2804  024000  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
2805  024002  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
2806  024004  000002                  RTI                     ;;RETURN
2807  024006  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
2808                          .SBTTL  TRAP DECODER
2809
2810                          ;;*****************************************************************
2811                          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2812                          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2813                          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2814                          ;*GO TO THAT ROUTINE.
2815
2816  024010  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2817  024012  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
2818  024016  005740                  TST     -(R0)           ;;BACKUP BY 2
2819  024020  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2820  024022  006300                  ASL     R0              ;;POSITION FOR INDEXING
2821  024024  016000  024044          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2822  024030  000200                  RTS     R0              ;;GO TO ROUTINE
2823
2824
```



```
2825                          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
2826
2827  024032  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
2828  024034  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
2829  024042  000002                  RTI                     ;;RESTORE THE PSW
2830
2831                          .SBTTL  TRAP TABLE
2832
2833                          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2834                          ;*BY THE "TRAP" INSTRUCTION.
2835
2836                          ;       ROUTINE
2837                          ;       -------
2838  024044  024032          $TRPAD: .WORD   $TRAP2
2839  024046  022050                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
2840  024050  022314                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2841  024052  022270                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2842  024054  022330                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2843
2844  024056  023060                  $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
2845
2846  024060  022770                  $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
2847  024062  023272                  $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
2848  024064  023362                  $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
2849  024066  023706                  $RDOCT  ;;CALL=RDOCT    TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
2850                          .SBTTL  POWER DOWN AND UP ROUTINES
2851
2852                          ;;*****************************************************************
2853                          ;POWER DOWN ROUTINE
2854  024070  012737  024230  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2855  024076  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2856  024104  010046                  MOV     P0,-(SP)        ;;PUSH R0 ON STACK
2857  024106  010146                  MOV     P1,-(SP)        ;;PUSH R1 ON STACK
2858  024110  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2859  024112  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2860  024114  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2861  024116  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2862  024120  017746  155014          MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
2863  024124  010637  024234          MOV     SP,$SAVR6       ;;SAVE SP
2864  024130  012737  024142  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
2865  024136  000000                  HALT
2866  024140  000776                  BR      .-2             ;;HANG UP
2867
2868                          ;;*****************************************************************
2869                          ;POWER UP ROUTINE
2870  024142  012737  024230  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
2871  024150  013706  024234          MOV     $SAVR6,SP       ;;GET SP
2872  024154  005037  024234          CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
2873  024160  005237  024234          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
2874  024164  001375                  BNE     1$              ;;OF  WORD
2875  024166  012677  154746          MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
2876  024172  012605                  MOV     (SP)+,R5        ;;POP STACK INTO R5
2877  024174  012604                  MOV     (SP)+,R4        ;;POP STACK INTO R4
2878  024176  012603                  MOV     (SP)+,R3        ;;POP STACK INTO R3
2879  024200  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
2880  024202  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
```

```
  2881  024204  012600                      MOV     (SP)+,R0        ;;POP STACK INTO R0
  2882  024206  012737  024070  000024      MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
  2883  024214  012737  000340  000026      MOV     #340,@#PWRVEC+2 ;;PRIO:7
  2884  024222  104401                      TYPE                    ;REPORT THE POWER FAILURE
  2885  024224  024236              $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
  2886  024226  000002                      RTI
  2887  024230  000000              $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
  2888  024232  000776                      BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
  2889  024234  000000              $SAVR6: 0                       ;;PUT THE SP HERE
  2890  024236  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
  2891  024244  000122
  2892                                  .EVEN
  2893          000001                  .END
```



```
AUTSL2  002624      CYCLE   004154      EXECUT  005362      PRO1    003266      SECCNT  001344
BA      001406      CYCLE2  004200      EXIT    003346      PRO2    003140      SECOND  003432
BADINP  003254      CYCL2   004170      EXITA   003634      PRTTWO  014600      SECONE  006732
BADONE  010340      CYL     013524      EXITB   003642      PR0   = 000000      SECSYS  003030
BAD.IN  011752      CYLADR  005262      EXITX   003410      PR1   = 000040      SECTBL  001305
BAD.ON  013734      CYLAD2  005266      EXIT2   006056      PR2   = 000100      SECT.0  013744
BASE    001266      CYLBAS  001315      EXTFR2  003352      PR3   = 000140      SECT.1  011762
BASINC  005340      CYLCNT  001342      EXTR    001420      PR4   = 000200      SECT.2  010350
BEGCT   017276      CYLOFF  005272      FAIL    013276      PR5   = 000240      SECT.3  002622
BEGCT1  017304      CYLTBL  001274      FAILED  013260      PR6   = 000300      SECT.4  017154
BEGIN   002602      DA      001410      FFLAG   021364      PR7   = 000340      SECT.5  020510
BIT0  = 000001      DDISP = 177570      FIL.DN  003652      PS    = 177776      SECT.6  021402
BIT00 = 000001      DISPLA  001142      GD1     012506      PSW   = 177776      SEEKI   001402
BIT01 = 000002      DISPRE  000174      GO      003272      PWRVEC= 000024      SEEKO   001404
BIT02 = 000004      DN1DRV  020226      GOOT    012462      RBA     001414      SEKSET  011550
BIT03 = 000010      DO2     004230      GO1     003304      RBUFF   013654      SHFCNT  020320
BIT04 = 000020      DPL   = 010000      GO2     003316      RDBUFF  007336      SHF1    013570
BIT05 = 000040      DRACTV  001164      GO3     003324      RDCHK   006430      SHF2    013566
BIT06 = 000100      DRCNT1  001311      GTSWR = 104405      RDCHK0  016674      SHF3    013564
BIT07 = 000200      DRIVE   001314      HDALGN  020752      RDCHR = 104407      SIN   = 001000
BIT08 = 000400      DRIV0   020470      HDRFLG  001320      RDLIN = 104410      SINST   020054
BIT09 = 001000      DRIV1   020472      HT    = 000011      RDLINK  005612      SMTME   005574
BIT1  = 000002      DRIV2   020474      IDEX    001324      RDOCT = 104411      STACK = 001100
BIT10 = 002000      DRIV3   020476      ILEGAL  004122      RDTBL   001226      START   001440
BIT11 = 004000      DRIV4   020500      INITIL  004504      RD1     005636      STARTR  001434
BIT12 = 010000      DRIV5   020502      INITI2  004516      RD2     005650      STFLG   001325
BIT13 = 020000      DRIV6   020504      INIT.2  010446      RD3     005656      STKLMT= 177774
BIT14 = 040000      DRIV7   020506      INSYS2  003506      RD4     005720      STOP    013556
BIT15 = 100000      DRVCNT  020322      IO      012512      RD5     005726      STORE   004052
BIT2  = 000004      DRV0    001206      IOTVEC= 000020      RD6     005732      STRT1   001754
BIT3  = 000010      DSKADR  001334      KYTEMP  001330      RD7     006032      SUR     013550
BIT4  = 000020      DSKTMP  001352      LDFLG   004204      READCS  001356      SWR     001140
BIT5  = 000040      DSWR  = 177570      LDSEEK  011564      RESTRT  006212      SWREG   000176
BIT6  = 000100      ECNT    001321      LF    = 000012      RESVEC= 000010      SWTCH   021366
BIT7  = 000200      EMTVEC= 000030      LFLF  = 105212      RETFR2  007250      SW0   = 000001
BIT8  = 000400      EM1     020324      LOGA    001166      RETRN2  006426      SW00  = 000001
BIT9  = 001000      EM10    021370      MANSEL  010336      RETRN3  005354      SW01  = 000002
BPTVEC= 000014      EM2     020335      MASK    005214      RETRN4  004016      SW02  = 000004
BUFF    013652      EM3     020347      MASKER  004010      RETRY   013166      SW03  = 000010
BUFR    022046      EM4     020362      MODE    001312      RFCERR  013234      SW04  = 000020
BUSADR  001336      EM5     020375      MORE    013040      RFERR   013216      SW05  = 000040
CHECK1  005430      EM6     020413      MOUNT   004276      RKBA    001374      SW06  = 000100
CHKCNT  001350      EM7     020434      MSKTBL  001256      RKCS    001370      SW07  = 000200
CKSWR = 104406      EM8     020447      NEXT    012376      RKDA    001376      SW08  = 000400
CLRDPL  020212      EM9     020455      NG      002574      RKDS    001364      SW09  = 001000
CNTSIN  001322      ERRCHK  006062      NXT1    017460      RKER    001366      SW1   = 000002
COM     012300      ERRFLG  001360      NXT2    017554      RKWC    001372      SW10  = 002000
COMMON  012274      ERRRF   001424      OSPFLG  001326      RWC     001416      SW11  = 004000
COMND   001316      ERRRFC  001426      PASTBL  001246      RWS   = 000100      SW12  = 010000
CONTIN  011364      ERRVEC= 000004      PATTRN  001362      RWSRDY  011336      SW13  = 020000
CONTRL  001332      EPRWCH  001430      PIRQ  = 177772      R6    =%000006      SW14  = 040000
CR    = 000015      ERRWCS  001432      PIRQVE= 000240      R7    =%000007      SW15  = 100000
CRLF  = 000200      ERRWF   001422      PRONUM  001313      SEC     013536      SW2   = 000004
```

```
SW3    = 000010            TYPE   = 104401            $CMTAG  001100            $MNEW   023675            $TKQOU  022522
SW4    = 000020            TYPOC  = 104402            $CM3    = 000000            $MSWR   023664            $TKQSR  022524
SW5    = 000040            TYPON  = 104404            $CNTLG  023657            $NULL   001154            $TKS    001144
SW6    = 000100            TYPOS  = 104403            $CNTLU  023652            $OCNT   022512            $TKSRV  022576
SW7    = 000200            WATSK   017704            $CRLF   001161            $OMODE  022514            $TN     = 000001
SW8    = 000400            WC      001412            $ENDAD  001400            $PASS   001100            $TPB    001152
SW9    = 001000            WCERR   013576            $ERFLG  001103            $POWER  024236            $TPFLG  001157
SYSER   013132             WCERRR  013112            $ERMAX  001115            $PWRDN  024070            $TPS    001150
SYSERR  013646             WCHERR  013626            $ERRPC  001116            $PWRMG  024224            $TRAP   024010
S10     012154             WFERR   013116            $ERRTB  001434            $PWRUP  024142            $TRAP2  024032
S11     012132             WPS     = 000040            $ERTTL  001112            $QUES   001160            $TRP    = 000012
S12     012252             WRDCNT  001340            $FILLC  001156            $RDCHR  023272            $TRPAD  024044
S13     012122             WRITCS  001354            $FILLS  001155            $RDLIN  023362            $TSTNM  001102
TABLTY  002534             WRLINK  005064            $GDADR  001120            $RDOCT  023706            $TTYIN  023616
TBITVE= 000014             WRPRO   016344            $GDDAT  001124            $RDSZ   = 000030            $TYPE   022050
TBLFUL  004114             WRRDCK  016134            $GTSWR  023060            $SAVR6  024234            $TYPEC  022220
TIME    005460             WRTNBY  001317            $HD     = 000003            $SETUP= 000114            $TYPEX  022266
TIMR    001346             X       015472            $HIOCT  024006            $STUP   = 177777            $TYPOC  022314
TIMR2   001323             XX      015706            $ICNT   001104            $SVPC   = 000222            $TYPON  022330
TKVEC   = 000060           $AUTOB  001134            $ILLUP  024230            $SWR    = 160000            $TYPOS  022270
TPVEC   = 000064           $BDADR  001122            $INTAG  001135            $TKB    001146            $0FILL  022513
TRAPVE= 000034             $BDDAT  001126            $ITEMB  001114            $TKCNT  022516            .       = 024246
TRTVEC= 000014             $BELL   023646            $LF     001162            $TKINT  022526
TRYAGN  011346             $CHARC  022264            $LPADR  001106            $TKQEN= 022525
TSTSN1  006134             $CKSWR  022770            $LPERR  001110            $TKQIN  022520

. ABS.  024246     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DZRKID,DZRKID/LI:ME/NL:MC:MD:CND/SOL/NSQ_DZRKID.P11
 RUN-TIME:  42 25 1 SECONDS
 RUN-TIME RATIO: 347/70=4.9
 CORE USED:  19K  (37 PAGES)
```