IDENTIFICATION

PRODUCT CODE:	AC-9236F-MC
PRODUCT NAME:	CZRKIF0 RK11 UTILITY PACKAGE
DATE CREATED:	MARCH, 1978
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:

'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

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 0,1,3.....

****

EXAMPLE 1

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

   PACK GOOD.
            RK11 UTILITY PACKAGE

            NAME                      TYPE
   INDEX                               0
   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    RK05 CONTROL PANEL TEST

     PURPOSE:      TO INSURE ALL SWITCHES INDICATOR LAMPS, AND INTERLOCKS
                   ARE FUNCTIONAL IN THE RK05
     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                               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

          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                                    SEQ 0021

                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
MAINDEC-11-DZRKI-E      MACY11 30A(1052)  24-MAR-78  09:23
DZRKIF.P11    24-MAR-78 09:20           TABLE OF CONTENTS                                  SEQ 0001

    22          BASIC DEFINITIONS
   132          TRAP CATCHER
   141          STARTING ADDRESS(ES)
   146          ACT11 HOOKS
   156          COMMON TAGS
   296          ERROR POINTER TABLE
   324          INITIALIZE THE COMMON TAGS
   353          TYPE PROGRAM NAME
   358          GET VALUE FOR SOFTWARE SWITCH REGISTER
   440          COMPATIBILITY TEST
  1155          OSCILLATING SEEK ROUTINE
  1307          FORMATTER-SURFACE VERIFIER
  1571          RK05 CONTROL PANEL TEST
  1841          CONTROL PANEL TEST # 2
  2165          HEAD ALIGNMENT ROUTINE
  2288          DISK POWER FAILURE TEST
  2412          TYPE ROUTINE
  2482          BINARY TO OCTAL (ASCII) AND TYPE
  2559          TTY INPUT ROUTINE
  2799          READ AN OCTAL NUMBER FROM THE TTY
  2837          TRAP DECODER
  2860          TRAP TABLE
  2879          POWER DOWN AND UP ROUTINES
```

```
                        .TITLE  MAINDEC-11-DZRKI-E
                        ;*COPYRIGHT (C) 1974,1977
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY BOB COLLINS
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
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
000222                                  $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                          $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
```

```
                                MSKTBL: .BYTE   377             ;TABLE OF CYLINDER BASE FOR AUTO MODE
                                        .BYTE   177
                                        .BYTE   077
                                        .BYTE   037
                                        .BYTE   017
                                        .BYTE   007
                                        .BYTE   003
                                        .BYTE   001

                                BASE:   .BYTE   0               ;CYL 0  BASE CYLINDER ADDRESS
                                        .BYTE   50              ;CYL 40 BASE CYLINDER ADDRESS
                                        .BYTE   120             ;CYL 80 BASE CYLINDER ADDRESS
                                        .BYTE   170             ;CYL 120 BASE CYLINDER ADDRESS
                                        .BYTE   240             ;CYL 160 BASE CYLINDER ADDRESS
                                        .BYTE   303             ;CYL 195 BASE CYLINDER ADDRESS

                                CYLTBL: .BLKB   11              ;TABLE OF SELECTED BASES

                                SECTBL: .BYTE   0               ;SECTOR 0
                                        .BYTE   4               ;SECTOR 4
                                        .BYTE   7               ;SECTOR 7
                                        .BYTE   13              ;SECTOR 12

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

                                        .EVEN

                                KYTEMP: .WORD   0               ;TEMP. KEYBOARD BUFFER
                                CONTRL: .WORD   0               ;TEMP. CONTROL+STATUS WORD
                                DSKADR: .WORD   0               ;TEMP. DISK ADDRESS WORD
                                BUSADR: .WORD   0               ;TEMP. BUS ADDRESS WORD
                                WRDCNT: .WORD   0               ;TEMP. WORD COUNT
                                CYLCNT: .WORD   -6000           ;WORD COUNT OF 1 CYLINDER
                                SECCNT: .WORD   -400            ;WORD COUNT OF 1 SECTOR
                                TIMR:   .WORD   0               ;TIMER FOR OPERATIONS
                                CHKCNT: .WORD   0               ;NUMBER OF ERROR PRINTOUTS
                                DSKTMP: .WORD   0               ;SAVE OF CURRENT DISK #
                                WRITCS: .WORD   4003            ;IBA+WRITE+GO
                                READCS: .WORD   5               ;READ+GO
                                ERRFLG: .WORD   0               ;ERROR FLAG INHIBIT ADDRESS CHANGE
                                PATTRN: .WORD   0               ;DATA PATTERN
                                RKDS:   .WORD   177400
```



```
                                RKER:   .WORD   177402
                                RKCS:   .WORD   177404
                                RKWC:   .WORD   177406
                                RKBA:   .WORD   177410
                                RKDA:   .WORD   177412
                                $ENDAD: .WORD   0
                                SEEKI:  .WORD   0
                                SEEKO:  .WORD   0

                                LFLF=   105212
                                BA:     BUFF
                                DA:     .WORD   0
                                WC:     .WORD   0
                                RBA:    RBUFF
                                RWC:    .WORD   0
                                EXTR:   .WORD   0
                                ERRWF:  .WORD   0
                                ERRRF:  .WORD   0
                                ERRRFC: .WORD   0
                                ERRWCH: .WORD   0
                                ERRWCS: .WORD   0

                                ;BIT DEFINITIONS

                                DPL=BIT12
                                RWS=BIT8
                                WPS=BIT5
                                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          STARTR: CLRB    @#MODE
001440  000005                  START:  RESET                   ;CLEAR THE BUS
001442  012706  001100                  MOV     #STACK,SP
001446  012746  000000                  MOV     #0,-(SP)        ;SET UP STACK FOR PSW=0
001452  012746  001460                  MOV     #2$,-(SP)       ;RETURN FOR RTI
001456  000002                          RTI
001460  000240                  2$:     NOP
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001462  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001466  005026                  1$:     CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001470  022706  001140                  CMP     #SWR,R6 ;;DONE?
001474  001374                          BNE     .-6             ;;LOOP BACK IF NO
001476  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001502  012737  024230  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001510  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001516  012737  024310  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001524  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
001532  013746  000004          10$:    MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001536  012737  001572  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001544  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001552  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001560  022777  177777  177352          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001566  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
001570  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
001572  012716  001600          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001576  000002                          RTI
001600  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001606  012737  000174  001142          MOV     #DISPREG,DISPLAY
001614  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001620  004737  022746                  JSR     PC,$TKINT       ;INITIALIZE THE TTY INTERRUPT HANDLER
                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
001624  005227  177777                  INC     #-1             ;;FIRST TIME?
001630  001045                          BNE     67$             ;;BRANCH IF NO
001632  104401  001670                  TYPE    ,68$            ;;TYPE ASCIZ STRING
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
001636  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
001642  001006                          BNE     69$             ;;BRANCH IF YES
001644  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
001652  001005                          BNE     70$             ;;BRANCH IF NO
001654  104405                          GTSWR                   ;;GET SOFT-SWR SETTINGS
001656  000403                          BR      70$
001660  112737  000001  001134  69$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
001666                          70$:
001666  000426                          BR      67$             ;;GET OVER THE ASCIZ
                                ;;68$:  .ASCIZ  <CRLF>/RK11 UTILITY PACKAGE/<15><12>/MAINDEC-11-DZRKI-E/<CRLF>
001744                          67$:
001744  105737  001312                  TSTB    @#MODE
001750  100002                          BPL     1$
001752  000137  003452                  JMP     @#SECOND
```

```
                                        1$:     TSTB    STFLG                   ;PRINT ONLY THE FIRST TIME
                                                BEQ     10$
                                                JMP     TABLTY
                                        10$:    INCB    STFLG
                                        STRT1:  CLRB    OSPFLG
                                                TYPE    ,65$            ;;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    ,$QUES          ;'?'
                                                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                     ;***
```

```
003114  104407                          RDCHR                   ;READ A CHARACTER
003116  012637  001330                  MOV     (SP)+,@#KYTEMP  ;GET THE RESPONSE
003122  104401  001330                  TYPE    ,KYTEMP         ;ECHO
003126  022737  000131  001330          CMP     #131,@#KYTEMP   ;WAS IT A "Y" (FOR YES)?
003134  001411                          BEQ     PRO2            ;IF YES BRANCH TO PROCESSOR 2
003136  022737  000116  001330          CMP     #116,@#KYTEMP   ;WAS RESPONSE LEGAL (N FOR NO)?
003144  001460                          BEQ     PRO1            ;IF LEGAL BRANCH
003146  104401  003154                  TYPE    ,67$            ;;TYPE ASCIZ STRING
003152  000401                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /?/
                                66$:
003156
003156  000734                          BR      SECSYS          ;GO BACK ASK AGAIN
003160  152737  000377  001313  PRO2:   BISB    #377,@#PRONUM   ;SET FLAG TWO PROCESSORS
003166  000240                          NOP                     ;***
003170  104401  003176                  TYPE    ,65$            ;;TYPE ASCIZ STRING
003174  000406                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/DRIVE # =/
                                64$:
003212
003212  104407                          RDCHR                   ;READ A CHARACTER
003214  012637  001330                  MOV     (SP)+,@#KYTEMP  ;PICK UP THE RESPONSE
003220  104401  001330                  TYPE    ,KYTEMP         ;ECHO
003224  162737  000060  001330          SUB     #60,@#KYTEMP    ;MAKE IT A NUMBER
003232  100420                          BMI     BADINP          ;IF NOT A NUMBER, BRANCH
003234  022737  000010  001330          CMP     #10,@#KYTEMP    ;IS IT A LEGAL #?
003242  003414                          BLE     BADINP          ;IF NO BRANCH
003244  000337  001330                  SWAB    @#KYTEMP        ;GET THE DRIVE # TO THE HIGH BYTE
003250  052737  040000  001330          BIS     #BIT14,@#KYTEMP ;SET THE SECOND SYSTEM BIT
003256  113705  001311                  MOVB    @#DRCNT1,R5     ;GET THE DRIVE COUNT
003262  006105                          ROL     R5              ;MAKE IT AN INDEX
003264  013765  001330  001166          MOV     @#KYTEMP,LOGA(R5) ;STORE THE SYSTEM #2 WORD
003272  000407                          BR      GO              ;GO DO THE TEST
003274                          BADINP:
003274  104401  003302                  TYPE    ,65$            ;;TYPE ASCIZ STRING
003300  000401                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /?/
                                64$:
003304
003304  000725                          BR      PRO2            ;GO BACK ASK AGAIN
003306  105037  001313          PRO1:   CLRB    @#PRONUM        ;CLEAR THE FLAG ONE PROCESSOR

                                ;THIS IS THE ACTUAL PROGRAM

003312  012700  001166          GO:     MOV     #LOGA,R0        ;GET THE TABLE ADDRESS TO R0
003316  105037  001324                  CLRB    @#IDEX          ;CLRB THE INDEX
003322  000405                          BR      GO2             ;BRANCH AROUND INCREMENT ROUTINE

003324  062700  000002          GO1:    ADD     #2,R0           ;INDEX THRU THE TABLE
003330  022700  001206                  CMP     #DRV0,R0        ;DONE?
003334  001414                          BEQ     EXIT            ;IF YES GET OUT
003336  005710                  GO2:    TST     (R0)            ;IS THE DRIVE ACTIVE
003340  100001                          BPL     GO3             ;IF YES BRANCH
003342  000770                          BR      GO1             ;NO TRY THE NEXT ONE
003344  011037  001164          GO3:    MOV     (R0),@#DRACTV   ;PICK UP THE ACTIVE DRIVE WORD
003350  004737  004174                  JSR     PC,CYCLE        ;CALL CYCLE (PICK UP DRIVE #, CYL BASE, AND CALL MOUNT)
003354  004737  005104                  JSR     PC,WRLINK       ;CALL WRITE LINK TO LOAD REGISTERS FOR WRITE
003360  004737  005632                  JSR     PC,RDLINK       ;CALL READ LINK TO LOAD REGISTERS FOR READ
003364  000757                          BR      GO1             ;GO GET NEXT DRIVE
```



```
003366  012700  001166          EXIT:   MOV     #LOGA,R0
003372  022700  001206          EXTFR2: CMP     #DRV0,R0
003376  001414                          BEQ     EXITX
003400  032710  040000                  BIT     #BIT14,(R0)
003404  001011                          BNE     EXITX
003406  005720                          TST     (R0)+
003410  100770                          BMI     EXTFR2
003412  014001                          MOV     -(R0),R1
003414  004737  004250                  JSR     PC,DO2
003420  004737  005632                  JSR     PC,RDLINK
003424  005720                          TST     (R0)+
003426  000761                          BR      EXTFR2
003430                          EXITX:
003430  104401  003438                  TYPE    ,65$            ;;TYPE ASCIZ STRING
003434  000404                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/DONE!/
                                64$:
003446
003446  000137  001440                  JMP     @#START         ;RESTART

                                ;THIS IS PROCESSOR #2 CODE. THIS ROUTINE ASKS FOR AND UNPACKS THE CONTROL
                                ;WORDS FROM THE FIRST PROCESSOR.

003452                          SECOND:
003452  104401  003460                  TYPE    ,65$            ;;TYPE ASCIZ STRING
003456  000415                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/COMPATIBILITY-SYSTEM#2/
                                64$:
003512
003512  012704  000200                  MOV     #200,R4         ;SET UP MASK BIT IN R4
003516  012703  000001                  MOV     #1,R3           ;SET UP WORD COUNTER IN R3
003522  012702  001166                  MOV     #LOGA,R2        ;GET THE TABLE ADDRESS
003526                          INSYS2:
003526  104401  003534                  TYPE    ,65$            ;;TYPE ASCIZ STRING
003532  000405                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/WORD  /
                                64$:
003546
003546  000240                          NOP                     ;***
003550  010346                          MOV     R3,-(SP)        ;GET READY TO TYPE WORD COUNTER
003552  104403                          TYPOS
003554     006                          .BYTE   6
003555     000                          .BYTE   0
003556  104401  003564                  TYPE    ,67$            ;;TYPE ASCIZ STRING
003562  000401                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /=/
                                66$:
003566
003566  104411                          RDOCT
003570  012600                          MOV     (SP)+,R0        ;PICK UP THE OCTAL WORD
003572  110001                          MOVB    R0,R1           ;GET THE FIRST DRIVE TO R1
003574  000300                          SWAB    R0              ;GET THE SECOND DRIVE TO R0 LOW BYTE
003576  042700  177400                  BIC     #177400,R0      ;CLEAR THE UNUSED BITS
003602  042701  177400                  BIC     #177400,R1      ;CLEAR THE UNUSED BITS
003606  006000                          ROR     R0              ;ROTATE RIGHT R0
003610  103003                          BCC     2$              ;IF CARRY IS CLEAR BRANCH
003612  052700  000200                  BIS     #BIT7,R0        ;SET DOWN BIT IF CARRY SET
003616  000241                          CLC                     ;AND CLEAR THE CARRY BIT
003620  006001                  2$:     ROR     R1              ;NOW DO THE SAME FOR R1
003622  103002                          BCC     3$              ;IF NO ERROR, BRANCH
```

```
003624  052701  000200          BIS     #BIT7,R1        ;IF ERROR SET THE BIT
003630  110162  000001  3$:     MOVB    R1,1(R2)        ;SET DRIVE FIRST IN TABLE
003634  004737  004030          JSR     PC,MASKER       ;CALL THE MASK CONTROL SUBROUTINE
003640  110062  000001          MOVB    R0,1(R2)        ;SET DRIVE SECOND IN TABLE (NEXT WORD)
003644  004737  004030          JSR     PC,MASKER       ;CALL THE MASK CONTROL SUBROUTINE
003650  005203                  INC     R3              ;INCREMENT THE WORD COUNTER
003652  000725                  BR      INSYS2          ;GO BACK AND GET NEXT WORD
003654  050412          EXITA:  BIS     R4,(R2)         ;FILL THE TABLE (LAST WORD)
003656  006004                  ROR     R4              ;WITH ALL BITS SET
003660  103375                  BCC     EXITA           ;GO BACK IF NOT DONE
003662  042712  174000  EXITB:  BIC     #174000,(R2)    ;CLEAR DOWN AND SECOND SYSTEM BITS AT TABLE
003666  010200                  MOV     R2,R0           ;GET ADDRESS TO R0 (CURRENT TABLE)
003670  005722                  TST     (R2)+           ;ADD 2 TO THE POINTER
003672  022702  001206  FIL.DN: CMP     #DRV0,R2        ;TABLE FULL?
003676  001403                  BEQ     2$              ;IF YES BRANCH
003700  012722  100000          MOV     #BIT15,(R2)+    ;FILL THE TABLE
003704  000772                  BR      FIL.DN          ;GO BACK TRY AGAIN
003706  011001          2$:     MOV     (R0),R1         ;GET THE WORD TO R1
003710  110102                  MOVB    R1,R2
003712  004737  005234          JSR     PC,MASK
003716  004737  004250          JSR     PC,DO2          ;FORM DRIVE # FOR MOUNT
003722  004737  005104          JSR     PC,WRLINK       ;WRITE NEW INFO
003726  004737  005632          JSR     PC,RDLINK       ;READ SAMPLE (ALL DRIVES)
003732  104401  003740          TYPE    ,65$            ;;TYPE ASCIZ STRING
003736  000427                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  <15><12>/DONE SYSTEM 2 RESTART SYSTEM 1, TYPE WORD /
004016                  64$:
004016  011046                  MOV     (R0),-(SP)      ;GET WORD FOR SYSTEM 1 AND TYPE
004020  104403                  TYPOS
004022     006                  .BYTE   6
004023     001                  .BYTE   1
004024  000000          1$:     HALT                    ;HALT (FINISHED SYSTEM #2)
004026  000776                  BR      1$              ;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.

004030  005712          MASKER: TST     (R2)            ;IS THE DRIVE UP
004032  100401                  BMI     RETRN4          ;IF NO, BRANCH
004034  110412                  MOVB    R4,(R2)         ;MOVE THE MASK BIT IN THE TABLE
004036  006004          RETRN4: ROR     R4              ;ROTATE THE MASK
004040  103003                  BCC     1$              ;DONE?,IF NO,BRANCH
004042  012716  003662          MOV     #EXITB,(SP)     ;SET UP FOR RETURN
004046  000207                  RTS     PC
004050  032712  040000  1$:     BIT     #BIT14,(R2)     ;IS THIS SYSTEM # 2'S DRIVE?
004054  001403                  BEQ     2$              ;IF NO,BRANCH
004056  012716  003654          MOV     #EXITA,(SP)     ;SET UP FOR RETURN
004062  000207                  RTS     PC
004064  062702  000002  2$:     ADD     #2,R2           ;INDEX THRU LOGA TABLE
004070  000207                  RTS     PC              ;RETURN

                        ;ROUTINE TO BUILD THE ACTIVE LOGICAL UNIT TABLE

004072  022737  000010  001330  STORE:  CMP     #10,@#KYTEMP    ;IS INPUT A LEGAL NUMBER?
004100  000240                  NOP                     ;***
```



```
004102  003417                  BLE     ILEGAL          ;IF NOT BRANCH
004104  000337  001330          SWAB    @#KYTEMP        ;ALIGN DRIVE # FOR TABLE
004110  013721  001330          MOV     @#KYTEMP,(R1)+  ;PUT THE WORD IN THE TABLE
004114  005037  001330          CLR     @#KYTEMP        ;CLEAR THE TEMP WORD
004120  105237  001311          INCB    @#DRCNT1        ;INCREMENT THE COUNTER
004124  022701  001206          CMP     #DRV0,R1        ;IS THE TABLE FULL?
004130  001401                  BEQ     TBLFUL          ;IF YES BRANCH
004132  000207                  RTS     PC              ;IF NO RETURN
004134  012716  003050  TBLFUL: MOV     #SECSYS,(SP)    ;IF TABLE FULL SET UP FOR SYSTEM #2
004140  000207                  RTS     PC              ;RETURN
004142  012716  002644  ILEGAL: MOV     #AUTSL2,(SP)    ;IF ILLEGAL RESPONSE, GO BACK
004146  104401  004154          TYPE    ,65$            ;;TYPE ASCIZ STRING
004152  000407                  BR      64$             ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  /ILLEGAL INPUT/
004172                  64$:
004172  000207                  RTS     PC              ;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

004174  011001  CYCLE:  MOV     (R0),R1         ;GET THE LOGICAL UNIT ACTIVE WORD TO R1
004176  032701  040000          BIT     #BIT14,R1       ;IS IT ON SYSTEM #2
004202  001402                  BEQ     CYCL2           ;IF NO BRANCH
004204  000137  006752          JMP     @#SECONE        ;GO TO SYSTEM #2
004210  113703  001324  CYCL2:  MOVB    @#IDEX,R3       ;GET THE INDEX VALUE
004214  116302  001256          MOVB    MSKTBL(R3),R2   ;GET THE MASK TO R2
004220  004737  005234  CYCLE2: JSR     PC,MASK         ;CALL THE MASK SUBROUTINE
004224  012703  001166  LDFLG:  MOV     #LOGA,R3        ;GET TABLE ADDRESS TO R3
004230  020003          2$:     CMP     R0,R3           ;IS ACTIVE ADDRESS=TO FIRST ADDRESS
004232  001002                  BNE     3$              ;IF NO BRANCH
004234  050223                  BIS     R2,(R3)+        ;IF YES SET BITS TO SHOW WRITE
004236  000401                  BR      4$              ;BRANCH TO SEE IF DONE
004240  040223          3$:     BIC     R2,(R3)+        ;CLEAR BITS TO SHOW OVER-WRITE
004242  022703  001206  4$:     CMP     #DRV0,R3        ;DONE
004246  001370                  BNE     2$              ;IF NO GO BACK
004250  000301  DO2:    SWAB    R1              ;GET DRIVE # TO LOW BYTE
004252  000240                  NOP                     ;***
004254  110102                  MOVB    R1,R2           ;GET IT TO R2
004256  042702  000370          BIC     #370,R2         ;CLEAR THE UNUSED BITS
004262  110237  001314          MOVB    R2,@#DRIVE      ;GET THE DRIVE #
004266  006102                  ROL     R2              ;SHIFT THE DRIVE # TO
004270  006102                  ROL     R2              ;ALIGN IT FOR THE DRIVE ADDR.
004272  006102                  ROL     R2              ;KEEP IT MOVING!
004274  006102                  ROL     R2              ;A LITTLE MORE!
004276  006102                  ROL     R2              ;THERE IT IS
004300  110237  001353          MOVB    R2,@#DSKTMP+1
004304  110237  001335          MOVB    R2,@#DSKADR+1   ;GET IT TO DISK ADDR. TEMP.
004310  004737  004316          JSR     PC,MOUNT        ;CALL MOUNT
004314  000207                  RTS     PC              ;RETURN

004316  105237  001324  MOUNT:  INCB    @#IDEX          ;INCREMENT THE INDEX
004322  000240                  NOP                     ;***
004324  112737  000005  001321  MOVB    #5,@#ECNT       ;SET ERROR CNTR TO 5
004332  112737  000003  001322  MOVB    #3,@#CNTSIN     ;SET SIN CNTR TO 3
```

```
                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                        BR      64$             ;;GET OVER THE ASCIZ
                ;;65$:  .ASCIZ  <15><12>/MOUNT PACK ON DRIVE #/
                64$:
                        MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                        .BYTE   1               ;;TYPE 1 DIGIT(S)
                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                        BR      66$             ;;GET OVER THE ASCIZ
                ;;67$:  .ASCIZ  <15><12>/MAKE PACK WRITE ENABLE/
                66$:
                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                        BR      68$             ;;GET OVER THE ASCIZ
                ;;69$:  .ASCIZ  <15><12>/PRESS CONTINUE WHEN DRIVE RDY/
                68$:
                        HALT
                        JSR     PC,INITIL       ;CALL INITIALIZER
                        RTS     PC              ;RETURN

                ;THIS ROUTINE INITIALIZES A DRIVE AND INSURES THAT IT IS READY AND
                ;WRITE ENABLED, IT IS ENTERED FROM MOUNT OR FROM EXECUTE IF OPERATION FAILS

                INITIL: MOV     R0,-(SP)        ;SAVE R0
                        MOV     @#DSKADR,R0     ;GET THE DRIVE # TO R0
                        BIC     #1777,R0        ;CLEAR THE UNUSED BITS
                INITI2: CLR     @#TIMR          ;CLEAR THE TIMER
                        CLRB    @#TIMR2
                        NOP                     ;***
                        MOV     R0,@RKDA        ;GET DRIVE # TO 'DA' REGISTER
                        MOV     #1,@RKCS        ;ISSUE CONTROL RESET + GO
                        JSR     PC,SMTME
                1$:     TSTB    @RKCS           ;DID CONTROL READY SET
                        BMI     2$              ;IF YES BRANCH
                        INC     @#TIMR          ;IF NO INCREMENT THE TIMER
                        BNE     1$              ;IF TIMER NOT ZERO BRANCH
                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                        BR      64$             ;;GET OVER THE ASCIZ
                ;;65$:  .ASCIZ  <15><12>/CONTROL RESET TIME OUT/
                64$:
                2$:     MOV     R0,@RKDA        ;DRIVE NUMBER TO 'DA' REG.
                        TSTB    @RKDS           ;IS DRIVE READY
                        BMI     3$              ;IF YES BRANCH
                        TYPE    ,67$            ;;TYPE ASCIZ STRING
                        BR      66$             ;;GET OVER THE ASCIZ
                ;;67$:  .ASCIZ  <15><12>/DRIVE NOT READY/
                66$:
                        BR      INITI2          ;GO BACK TRY AGAIN
                3$:     BIT     #BIT5,@RKDS     ;IS DRIVE WRITE LOCKED?
                        BEQ     4$              ;IF NO, BRANCH
                        TYPE    ,69$            ;;TYPE ASCIZ STRING
                        BR      68$             ;;GET OVER THE ASCIZ
                ;;69$:  .ASCIZ  <15><12>/DRIVE WRITE PROTECTED/
                68$:
                        BR      INITI2          ;YES, GO BACK TRY AGAIN
                4$:     CLR     @#TIMR          ;CLEAR THE TIMER
```



```
                        MOV     R0,@RKDA        ;GET THE DRIVE # TO 'DA' REGISTER
                        MOV     #15,@RKCS       ;ISSUE DRIVE RESET + GO
                        JSR     PC,SMTME
                5$:     TSTB    @RKCS
                        BPL     5$
                        BIT     #100,@RKDS      ;READ/WRITE/SEEK READY BIT SET?
                        BNE     6$              ;IF YES, BRANCH
                        INC     @#TIMR          ;NO, INCREMENT THE TIMER
                        BNE     5$              ;GO BACK AND CHECK IF TIMER NOT 0
                        TSTB    @#TIMR2
                        BNE     7$
                        INCB    @#TIMR2
                        BR      5$
                7$:
                        TYPE    ,71$            ;;TYPE ASCIZ STRING
                        BR      70$             ;;GET OVER THE ASCIZ
                ;;71$:  .ASCIZ  <15><12>/DRIVE RESET TIMED OUT/
                70$:
                        BR      4$              ;GO BACK, TRY AGAIN
                6$:     MOV     (SP)+,R0        ;RESTORE R0
                        RTS     PC              ;RETURN TO CALLER

                ;THIS ROUTINE TAKES CARE OF ALL LINKAGES FOR THE
                ;EXECUTE ROUTINE.  IT FORMS THE ADDRESS AND SETS UP ALL THE
                ;REGISTERS FOR THE WRITE OPERATION

                WRLINK: CLRB    @#COMND         ;INDICATE WRITE OPERATION
                        NOP                     ;***
                        MOVB    @#DRIVE,R1      ;PICK UP THE DRIVE #
                        ROL     R1              ;MAKE IT A WORD INDEX
                1$:     MOV     DRV0(R1),@#PATTRN ;PICK UP THE DATA PATTERN
                        JSR     PC,CYLADR       ;CALL CYLINDER ADDRESS
                        BR      2$              ;RETURN HERE IF NOT LAST BASE
                        RTS     PC              ;RETURN HERE IF LAST BASE
                2$:     MOV     #PATTRN,@#BUSADR ;GET THE ADDRESS OF THE OUTPUT
                        MOV     @#CYLCNT,@#WRDCNT ;GET THE WORD COUNT
                        MOV     @#WRITCS,@#CONTRL ;GET THE CONTROL + STATUS WORD
                        JSR     PC,EXECUT       ;CALL EXECUTE
                        BIT     #BIT4,@#DSKADR  ;WAS THIS WRITE SURFACE "1"?
                        BNE     3$              ;IF YES BRANCH
                        BIS     #BIT4,@#DSKADR  ;SET SURFACE ONE BIT
                        COMB    @#PATTRN        ;MAKE IT SURFACE ONE DATA
                        BR      2$              ;RELOAD REGISTERS AND EXECUTE
                3$:     BIC     #BIT4,@#DSKADR  ;SET UP FOR SURFACE "0"
                        COMB    @#PATTRN        ;MAKE IT SURFACE 0 DATA
                        INC     R2              ;INC. THRU SELECTED CYL. OFFSET TABLE
                        JSR     PC,CYLOFF       ;GET THE CYLINDER VALUE
                        BR      2$              ;RETURN HERE IF MORE TO READ
                        RTS     PC              ;RETURN HERE IF FINISHED

                ;THIS ROUTINE EXPECTS TO FIND A MASK IN R2, AND FROM THIS MASK
                ;BUILDS A TABLE (AT CYLTBL) OF CYLINDER ADDRESS OFFSETS; THE TABLE
                ;IS TERMINATED BY A #377

                MASK:   MOV     R5,-(SP)        ;SAVE R5
                        BIC     #177400,R2      ;CLR THE UNUSED BITS OF THE MASK
```

```
                        000240                  NOP                     ;***
                        012703  000200          MOV     #200,R3         ;SET UP THE COMPARE MASK
                        005004                  CLR     R4              ;CLR THE INDEX COUNTER
                        012705  001274          MOV     #CYLTBL,R5      ;GET THE CYLINDER TABLE ADDRESS
                        030203          1$:     BIT     R2,R3           ;IS THE MASK BIT SELECTED IN BASE
                        001401                  BEQ     2$              ;IF NO BRANCH
                        110425                  MOVB    R4,(R5)+        ;MOVE THE CYLINDER BASE TO THE TABLE
                        105204          2$:     INCB    R4              ;INCREMENT THE BASE
                        006003                  ROR     R3              ;ROTATE THE COMPARE MASK
                        103372                  BCC     1$              ;IF NOT DONE GO BACK
                        112715  000377          MOVB    #377,(R5)       ;IF DONE LOAD FINISH FLAG
                        012605                  MOV     (SP)+,R5        ;RESTORE R5
                        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

                        001266  CYLADR: MOV     #BASE,R3        ;GET THE CYLINDER TABLE ADDRESS
                        001274  CYLAD2: MOV     #CYLTBL,R2      ;GET THE SELECTED CYL BASE ADDR.
                                CYLOFF: MOVB    (R2),R4         ;GET THE SELECTED CYL VALUE TO R4
                                        NOP                     ;***
                        000377          CMPB    #377,R4         ;IS IT THE TABLE TERMINATOR?
                                        BEQ     BASINC          ;IF YES BRANCH
                                        CLR     -(SP)           ;INSURE CLEAN WORD
                                        MOVB    (R3),(SP)       ;GET THE CYL ADDRESS ON THE STACK
                                        ADD     (SP)+,R4        ;AND IT TO THE SELECTED OFFSET
                                        ROL     R4              ;SHIFT THIS RESULT
                                        ROL     R4              ;TO ALIGN THE NEWLY FORMED
                                        ROL     R4              ;CYLINDER ADDRESS WITH BITS
                                        ROL     R4              ;5 THRU 12 OF RKDA AND
                                        ROL     R4              ;STORE THIS IN DSKADR
                        017777  001334  BIC     #017777,@#DSKADR;CLEAR ALL BUT DRIVE NUMBER
                        001334          BIS     R4,@#DSKADR     ;PUT IT IN DSKADR
                                        RTS     PC
                                BASINC: INC     R3              ;PICK UP ADDRESS OF NEXT BASE CYL.
                                        NOP                     ;***
                        001274          CMP     #CYLTBL,R3      ;ARE YOU FINISHED?
                                        BEQ     RETRN3          ;IF YES, BRANCH
                                        BR      CYLAD2          ;NO GO BACK
                                RETRN3: ADD     #2,(SP)         ;SET-UP FOR PC+2
                                        RTS     PC              ;RETURN

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

                        001346  EXECUT: CLR     @#TIMR          ;CLEAR THE TIMER
                                        NOP                     ;HALT HERE TO CHECK COMMAND ABOUT TO BE EXECUTED
                        001323          CLRB    @#TIMR2         ;CLEAR SECOND TIMER
                        001334  173754  MOV     @#DSKADR,@RKDA  ;LOAD THE DISK ADDRESS REGISTER
                        001336  173744  MOV     @#BUSADR,@RKBA  ;LOAD THE BUS ADDRESS REGISTER
                        001340  173734  MOV     @#WRDCNT,@RKWC  ;LOAD THE WORD COUNT REGISTER
                        001332  173724  MOV     @#CONTRL,@RKCS  ;LOAD THE CONTROL REGISTER
                                        JSR     PC,SMTME        ;KILL TIME FOR RK11-C
                                CHECK1: TSTB    @RKCS           ;IS CONTROL READY SET
                                        BPL     TIME            ;IF NO BRANCH
                                        JSR     PC,ERRCHK
```



```
                                        TST     @#ERRFLG        ;ERROR?
                                        BEQ     1$              ;IF NO BRANCH
                                        CLR     @#ERRFLG        ;CLEAR THE FLAG
                                        BR      EXECUT          ;TRY AGAIN
                                1$:     RTS     PC
                                TIME:   INC     @#TIMR
                                        NOP                     ;***
                                        BNE     CHECK1
                                        TSTB    @#TIMR2         ;SECOND TIMEOUT?
                                        BNE     1$              ;IF YES BRANCH
                                        INCB    @#TIMR2         ;INDICATE SECOND TIMEOUT
                                        BR      CHECK1          ;GO BACK
                                1$:     JSR     PC,INITIL
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/TIMED OUT ON OPERATION RETRY IN PROGRESS/
                                64$:
                                        BR      EXECUT
                        000500  001346  SMTME:  MOV     #500,@#TIMR
                                1$:     DEC     @#TIMR
                                        BNE     1$
                                        RTS     PC

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

                                RDLINK: MOV     R0,-(SP)        ;SAVE R0
                                        NOP                     ;***
                        001316          BISB    #377,@#COMND    ;INDICATE READ OPERATION
                        001166          MOV     #LOGA,R1        ;GET THE TABLE ADDRESS TO R1
                        001305          MOV     #SECTBL,R5      ;GET THE SECTOR TABLE ADDRESS
                                        BR      RD2             ;SKIP OVER THE INDEX, FIRST PASS
                                RD1:    ADD     #2,R1           ;ADD 2 TO THE ADDRESS
                        001206          CMP     #DRV0,R1        ;ARE YOU THRU THE ENTIRE TABLE
                                        BEQ     EXIT2           ;IF YES EXIT
                                RD2:    TST     (R1)            ;IS THE DRIVE ACTIVE
                                        BPL     RD3             ;IF YES BRANCH
                                        BR      RD1             ;IF NO GET NEXT WORD
                                RD3:    MOV     (R1),R0         ;GET THE ACTIVE WORD TO R0
                                        MOV     R0,R2           ;COPY THE WORD
                                        SWAB    R0              ;GET THE DRIVE # TO THE LOW BYTE
                        177770          BIC     #177770,R0      ;CLR ALL BUT THE DRIVE #
                        001317          MOVB    R0,@#WRTNBY     ;SAVE THE DRIVE #
                                        ROL     R0              ;MAKE IT AN INDEX
                        001206  001362  MOV     DRV0(R0),@#PATTRN ;PICK UP THE DATA PATTERN
                        005234          JSR     PC,MASK         ;CALL THE MASK SUBROUTINE
                        005302          JSR     PC,CYLADR       ;GO FORM A CYLINDER ADDRESS
                                        BR      RD4             ;RETURN HERE IF NOT LAST BASE ADDR.
                                        BR      RD1             ;IF LAST BASE ADDRESS, RETURN HERE
                        001352  001334  RD4:    BIS     @#DSKTMP,@#DSKADR ;SET THE DRIVE # BITS IN DISK ADDR.
                        001334          RD5:    BISB    (R5)+,@#DSKADR  ;SET THE SECTOR BITS IN DISK ADDR.
                        007356  001336  RD6:    MOV     #RDBUFF,@#BUSADR ;GET THE BUFFER ADDR.
                                        NOP                     ;***
                        001344  001340          MOV     @#SECCNT,@#WRDCNT ;GET THE WORD COUNT
```

```
 930  005770  013737  001356  001332          MOV     @#READCS,@#CONTRL ;GET THE READ CONTROL WORD
 931  005776  004737  005402                  JSR     PC,EXECUT         ;DO THE READ
 932  006002  004737  006450                  JSR     PC,RDCHK          ;CHECK THE DATA
 933  006006  042737  000017  001334          BIC     #17,@#DSKADR      ;CLR THE SECTOR BITS IN DISK ADDR.
 934  006014  022705  001311                  CMP     #DRCNT1,R5        ;WAS THIS THE LAST SECTOR?
 935  006020  001352                          BNE     RD5               ;IF NO GO BACK
 936  006022  012705  001305                  MOV     #SECTBL,R5        ;IF YES RESET SECTOR POINTER
 937  006026  032737  000020  001334          BIT     #BIT4,@#DSKADR    ;WAS IT SURFACE "1" THAT WAS READ?
 938  006034  001006                          BNE     RD7               ;IF YES, BRANCH
 939  006036  052737  000020  001334          BIS     #BIT4,@#DSKADR    ;NO, SET SURFACE "1" BIT
 940  006044  105137  001362                  COMB    @#PATTRN          ;MAKE HEAD "1" PATTERN
 941  006050  000736                          BR      RD5               ;GO BACK AND EXECUTE
 942  006052  042737  000020  001334  RD7:    BIC     #BIT4,@#DSKADR    ;CLEAR THE SURFACE BIT
 943  006060  105137  001362                  COMB    @#PATTRN          ;MAKE IT SURFACE 0 DATA
 944  006064  005202                          INC     R2                ;INCREMENT THE SELECTED CYL TABLE POINTER
 945  006066  004737  005312                  JSR     PC,CYLOFF         ;GO FORM NEXT ADDRESS
 946  006072  000722                          BR      RD4               ;IF HERE IT IS NOT THE LAST BASE ADDR
 947  006074  000670                          BR      RD1               ;IF HERE GET NEXT WORD-DRIVE FINISHED
 948
 949  006076  012600                  EXIT2:  MOV     (SP)+,R0          ;RESTORE R0
 950  006100  000207                          RTS     PC                ;RETURN TO MAIN LINE CODE
 951
 952                                  ;THE ERROR CHECK ROUTINE CHECKS IF AN ERROR OCCURRED
 953                                  ;ON WRITING OR READING.
 954
 955  006102  032777  140000  173260  ERRCHK: BIT     #140000,@RKCS     ;HARD ERROR OR ERROR SET?
 956  006110  000240                          NOP                       ;HALT HERE TO EXAMINE ERROR REG.,ECT.
 957  006112  001420                          BEQ     TSTSN1            ;IF NO, GO TEST 'SIN' BIT
 958  006114  012777  000001  173246          MOV     #1,@RKCS          ;IF YES, ISSUE CNTROL RESET + GO
 959  006122  004737  005614                  JSR     PC,SMTME
 960  006126  012777  177777  173224          MOV     #-1,@ERRFLG       ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)
 961  006134  105777  173230          1$:     TSTB    @RKCS             ;CNTROL READY BIT SET (FROM CNTROL RESET)
 962  006140  100375                          BPL     1$                ;IF NO WAIT.  (IF HUNG HERE RUN STATIC)
 963  006142  105337  001321                  DECB    @#ECNT            ;DECREMENT ERROR COUNTER
 964  006146  001002                          BNE     TSTSN1            ;HAVE ERROR BITS SET 5 TIMES?
 965  006150  000137  006232                  JMP     @#RESTRT          ;IF HERE 5 ERRORS HAVE OCCURRED
 966  006154  032777  001000  173202  TSTSN1: BIT     #1000,@RKDS       ;SEEK INCOMPLETE SET?
 967  006162  000240                          NOP                       ;***
 968  006164  001530                          BEQ     RETRN2            ;BRANCH IF NO
 969  006166  012777  000015  173174          MOV     #15,@RKCS         ;IF YES, ISSUE DRIVE RESET, GO
 970  006174  012777  177777  173156          MOV     #-1,@ERRFLG       ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)
 971  006202  004737  005614                  JSR     PC,SMTME
 972  006206  105777  173156          2$:     TSTB    @RKCS
 973  006212  100375                          BPL     2$
 974  006214  032777  000100  173142          BIT     #100,@RKDS        ;"R/W/S READY" BIT SET?
 975  006222  001771                          BEQ     2$                ;IF NO WAIT! (IF HUNG HERE RUN STATIC)
 976  006224  105337  001322                  DECB    @#CNTSIN          ;DECREMENT SEEK INCOMPLETE COUNTER
 977  006230  001106                          BNE     RETRN2            ;IF 3 'SIN' ERRORS FALL THROUGH
 978  006232  105737  001321          RESTRT: TSTB    @#ECNT
 979  006236  000240                          NOP                       ;***
 980  006240  001421                          BEQ     1$
 981  006242  104401  006250                  TYPE    ,65$              ;;TYPE ASCIZ STRING
 982  006246  000415                          BR      64$               ;;GET OVER THE ASCIZ
 983                                  ;;65$:  .ASCIZ  <15><12>/3 'SIN' ERRORS OCCURRED/
 984  006302                          64$:
 985  006302  000415                          BR      2$
```



```
 986  006304                          1$:
 987  006304  104401  006312                  TYPE    ,67$              ;;TYPE ASCIZ STRING
 988  006310  000412                          BR      66$               ;;GET OVER THE ASCIZ
 989                                  ;;67$:  .ASCIZ  <15><12>/5 ERRORS OCCURRED/
 990  006336                          66$:
 991  006336                          2$:
 992  006336  104401  006344                  TYPE    ,69$              ;;TYPE ASCIZ STRING
 993  006342  000422                          BR      68$               ;;GET OVER THE ASCIZ
 994                                  ;;69$:  .ASCIZ  / DRIVE DECLARED DOWN!! NOT TESTED !/
 995  006410                          68$:
 996  006410  105737  001316                  TSTB    @#COMND           ;TEST THE COMMAND
 997  006414  100006                          BPL     3$                ;IF WRITE, BRANCH
 998  006416  062706  000004                  ADD     #4,SP             ;POINT TO SAVE R0
 999  006422  012600                          MOV     (SP)+,R0          ;GET R0 BACK
1000  006424  052710  100000                  BIS     #BIT15,(R0)       ;SET THE DOWN BIT
1001  006430  000207                          RTS     PC                ;RETURN TO MAIN CODE
1002  006432  052710  100000          3$:     BIS     #BIT15,(R0)       ;SET THE DOWN BIT
1003  006436  012706  001100                  MOV     #STACK,SP         ;RESTORE THE STACK
1004  006442  000137  003324                  JMP     @#GO1
1005  006446  000207                  RETRN2: RTS     PC
1006
1007                                  ;THIS ROUTINE CHECKS A SECTORS WORTH OF DATA ON A READ OPERATION
1008                                  ;IT ALLOWS 5 ERROR PRINTOUTS PER SECTOR
1009
1010  006450  010446                  RDCHK:  MOV     R4,-(SP)          ;SAVE   R4
1011  006452  010546                          MOV     R5,-(SP)          ;SAVE R5 FOR RDLINK
1012  006454  105037  001320                  CLRB    @#HDRFLG          ;CLEAR THE PRINT HEADER FLAG
1013  006460  000240                          NOP                       ;***
1014  006462  012737  000005  001350          MOV     #5,@#CHKCNT       ;PUT ERROR COUNT IN CHECK COUNT
1015  006470  012704  007356                  MOV     #RDBUFF,R4        ;GET THE TABLE ADDRESS TO R4
1016  006474  013705  001362                  MOV     @#PATTRN,R5       ;GET THE EXPECTED DATA TO R5
1017  006500  020524                  1$:     CMP     R5,(R4)+          ;ARE THEY THE SAME
1018  006502  001515                          BEQ     3$                ;IF YES BRANCH
1019  006504  105737  001320                  TSTB    @#HDRFLG          ;IS THE HEADER FLAG CLEAR
1020  006510  001046                          BNE     2$                ;IF NO BRANCH
1021  006512  104401  006520                  TYPE    ,65$              ;;TYPE ASCIZ STRING
1022  006516  000420                          BR      64$               ;;GET OVER THE ASCIZ
1023                                  ;;65$:  .ASCIZ  <15><12>/ERROR! DATA WRITTEN BY DRIVE /
1024  006560                          64$:
1025  006560  152737  000377  001320          BISB    #377,@#HDRFLG     ;SET THE HEADER FLAG
1026  006566  113746  001317                  MOVB    @#WRTNBY,-(SP)    ;PICK UP THE DRIVE # THAT WROTE
1027  006572  104403                          TYPOS
1028  006574     001                          .BYTE   1
1029  006575     000                          .BYTE   0
1030  006576  104401  006604                  TYPE    ,67$              ;;TYPE ASCIZ STRING
1031  006602  000411                          BR      66$               ;;GET OVER THE ASCIZ
1032                                  ;;67$:  .ASCIZ  / CANNOT BE READ./
1033  006626                          66$:
1034  006626                          2$:
1035  006626  104401  006634                  TYPE    ,69$              ;;TYPE ASCIZ STRING
1036  006632  000405                          BR      68$               ;;GET OVER THE ASCIZ
1037                                  ;;69$:  .ASCIZ  <15><12>/  ADDR=/
1038  006646                          68$:
1039  006646  013746  001334                  MOV     @#DSKADR,-(SP)    ;PICK UP THE ADDRESS THAT FAILED
1040  006652  104403                          TYPOS
1041  006654     006                          .BYTE   6
```

```
1042  006655  001                     .BYTE   1
1043  006656  104401  006664          TYPE    ,71$            ;;TYPE ASCIZ STRING
1044  006662  000405                  BR      70$             ;;GET OVER THE ASCIZ
1045                          ;;71$:  .ASCIZ  /  EXPCTD=/
1046  006676                  70$:
1047  006676  010546                  MOV     R5,-(SP)        ;PICK UP THE EXPECTED DATA (GOOD)
1048  006700  104404                  TYPON
1049  006702  104401  006710          TYPE    ,73$            ;;TYPE ASCIZ STRING
1050  006706  000405                  BR      72$             ;;GET OVER THE ASCIZ
1051                          ;;73$:  .ASCIZ  /  RECVD=/
1052  006722                  72$:
1053  006722  016446  177776          MOV     -2(R4),-(SP)    ;PICK UP THE RECEIVED DATA (BAD)
1054  006726  104404                  TYPON
1055  006730  005337  001350          DEC     @#CHKCNT        ;DECREMENT THE CHECK COUNT
1056  006734  001403                  BEQ     4$              ;IF ZERO, BRANCH
1057  006736  022704  010356  3$:     CMP     #MANSEL,R4      ;DONE ALL CHECKS?
1058  006742  001256                  BNE     1$              ;IF NO, GO BACK
1059  006744  012605          4$:     MOV     (SP)+,R5        ;RESTORE R5
1060  006746  012604                  MOV     (SP)+,R4        ;RESTORE R4
1061  006750  000207                  RTS     PC              ;RETURN TO CALLER
1062
1063                          ;THIS ROUTINE BUILDS A TABLE OF PARAMETERS TO PASS TO THE SECOND SYSTEM
1064                          ;IT PACKS THE INFO FOR TWO DRIVES INTO ONE WORD
1065
1066  006752  005003          SECONE: CLR     R3              ;CLEAR THE WORD COUNTER
1067  006754  000240                  NOP                     ;***
1068  006756  012702  001256          MOV     #MSKTBL,R2
1069  006762  005042          6$:     CLR     -(R2)
1070  006764  022702  001246          CMP     #PASTBL,R2
1071  006770  001374                  BNE     6$
1072  006772  012700  001166          MOV     #LOGA,R0        ;GET THE ACTIVE TABLE ADDRESS
1073  006776  012001          1$:     MOV     (R0)+,R1        ;PICK UP THE WORD
1074  007000  000301                  SWAB    R1              ;GET THE DRIVE # TO THE LOW BYTE
1075  007002  042701  177400          BIC     #177400,R1      ;CLEAR THE UNWANTED BITS
1076  007006  106101                  ROLB    R1              ;ROTATE THE BYTE, WAS DOWN SET?
1077  007010  103002                  BCC     2$              ;IF NO BRANCH
1078  007012  052701  000001          BIS     #BIT0,R1        ;SHOW THE DRIVE AS DOWN IN THE TABLE
1079  007016  105701          2$:     TSTB    R1              ;IS THIS THE SYSTEM #2 DRIVE?
1080  007020  100404                  BMI     3$              ;BRANCH IF YES
1081  007022  142701  000360          BICB    #360,R1         ;CLEAR THE UNUSED BITS
1082  007026  110122                  MOVB    R1,(R2)+        ;GET THIS # TO THE PASS TABLE
1083  007030  000762                  BR      1$              ;GET THE NEXT WORD FROM ACTIVE TABLE
1084  007032  110112          3$:     MOVB    R1,(R2)         ;GET THE LAST DRIVE TO THE PASS TABLE
1085  007034  012702  001246          MOV     #PASTBL,R2      ;RESTORE THE TABLE POINTER
1086  007040  104401  007046          TYPE    ,65$            ;;TYPE ASCIZ STRING
1087  007044  000425                  BR      64$             ;;GET OVER THE ASCIZ
1088                          ;;65$:  .ASCIZ  <15><12>/LOAD AND START ADDRESS 210 ON SYSTEM #2/
1089  007120                  64$:
1090  007120  104401  007126          TYPE    ,67$            ;;TYPE ASCIZ STRING
1091  007124  000424                  BR      66$             ;;GET OVER THE ASCIZ
1092                          ;;67$:  .ASCIZ  <15><12>/AND TYPE THE BELOW WHEN ASKED FOR IT./
1093  007176                  66$:
1094  007176  005203          4$:     INC     R3              ;INCREMENT THE WORD COUNTER
1095  007200  104401  007206          TYPE    ,69$            ;;TYPE ASCIZ STRING
1096  007204  000405                  BR      68$             ;;GET OVER THE ASCIZ
1097                          ;;69$:  .ASCIZ  <15><12>/WORD  /
```



```
1098  007220                  68$:
1099  007220  010346                  MOV     R3,-(SP)        ;GET THE WORD COUNT ON THE STACK
1100  007222  104403                  TYPOS
1101  007224     006                  .BYTE   6
1102  007225     000                  .BYTE   0
1103  007226  104401  007234          TYPE    ,71$            ;;TYPE ASCIZ STRING
1104  007232  000401                  BR      70$             ;;GET OVER THE ASCIZ
1105                          ;;71$:  .ASCIZ  /=/
1106  007236                  70$:
1107  007236  012246                  MOV     (R2)+,-(SP)     ;GET THE FIRST TO THE STACK
1108  007240  104403                  TYPOS
1109  007242     006                  .BYTE   6
1110  007243     001                  .BYTE   1
1111  007244  032762  100000  177776  BIT     #BIT15,-2(R2)   ;WAS THIS THE TABLE TERMINATOR
1112  007252  001004                  BNE     5$              ;BRANCH IF YES
1113  007254  032762  000200  177776  BIT     #BIT7,-2(R2)    ;TERMINATOR?
1114  007262  001745                  BEQ     4$              ;IF NO BRANCH
1115  007264  005740          5$:     TST     -(R0)
1116  007266  000000                  HALT
1117
1118  007270                  RETFR2:
1119  007270  104401  007276          TYPE    ,65$            ;;TYPE ASCIZ STRING
1120  007274  000404                  BR      64$             ;;GET OVER THE ASCIZ
1121                          ;;65$:  .ASCIZ  <15><12>/WORD=/
1122  007306                  64$:
1123  007306  104411                  RDOCT
1124  007310  012602                  MOV     (SP)+,R2        ;GET THE WORD FROM SYSTEM 2 TO TABLE
1125  007312  042702  177400          BIC     #177400,R2
1126  007316  012704  001166          MOV     #LOGA,R4        ;SET POINTER LOOK FOR FIRST "UP" DRIVE
1127  007322  005724          1$:     TST     (R4)+           ;DRIVE UP?
1128  007324  100776                  BMI     1$              ;IF NO BRANCH
1129  007326  010437  001100          MOV     R4,@#$PASS
1130  007332  01440                   MOV     -(R4),R1
1131  007334  000240                  NOP                     ;***
1132  007336  004737  004224          JSR     PC,LDFLG        ;CALL D02+
1133  007342  004737  005632          JSR     PC,RDLINK       ;CALL READ CHECK
1134  007346  013700  001100          MOV     @#$PASS,R0
1135  007352  000137  003372          JMP     @#EXTFR2        ;GO TO END OF TEST
1136
1137  007356  000400          RDBUFF: .BLKW   400
1138
1139  010356  000240          MANSEL: NOP                     ;TABLE TERMINATOR
1140
1141
1142
1143
1144
1145  010360                  BADONE:
1146  010360  104401  010368          TYPE    ,65$            ;;TYPE ASCIZ STRING
1147  010364  000401                  BR      64$             ;;GET OVER THE ASCIZ
1148                          ;;65$:  .ASCIZ  /?/
1149  010370                  64$:
1150
1151
1152                                  .SBTTL  OSCILLATING SEEK ROUTINE
1153
```

```
1154  010370                          SECT.2:
1155  010370  104401  010376                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1156  010374  000416                          BR      64$             ;;GET OVER THE ASCIZ
1157                                  ;;65$:  .ASCIZ  <15><12>/OSCILLATING SEEK PACKAGE/
1158  010432                          64$:
1159
1160  010432  012700  020614                  MOV     #DRIV0,R0       ;FIND OUT WHICH DRIVES ARE
1161  010436  005001                          CLR     R1              ;PRESENT AND PUT THE
1162  010440  010177  170732          1$:     MOV     R1,@RKDA        ;DRIVE #'S IN A TABLE STARTING
1163  010444  105777  170714                  TSTB    @RKDS           ;AT 'DRIV0'. BITS 15-13 CONTAINS
1164  010450  100001                          BPL     2$              ;THE DRIVE #
1165  010452  010120                          MOV     R1,(R0)+
1166  010454  062701  020000          2$:     ADD     #20000,R1
1167  010460  001367                          BNE     1$
1168  010462  012710  177777                  MOV     #-1,(R0)        ;SET THE TERMINATOR TO THE TABLE
1169
1170  010466  013702  001376          INIT.2: MOV     RKDA,R2
1171  010472  012777  000001  170670          MOV     #1,@RKCS        ;ISSUE CONTROL RESET + GO !
1172  010500  004737  005614                  JSR     PC,SMTME
1173  010504  105777  170660          1$:     TSTB    @RKCS           ;DID CONTROL READY SET?
1174  010510  100375                          BPL     1$              ;IF NO WAIT! (IF HUNG RUN STATIC)
1175  010512  012700  020614                  MOV     #DRIV0,R0
1176  010516  012077  170654          3$:     MOV     (R0)+,@RKDA
1177  010522  012777  000015  170640          MOV     #15,@RKCS       ;ISSUE DRIVE RESET + GO!
1178  010530  004737  005614                  JSR     PC,SMTME
1179  010534  105777  170630          2$:     TSTB    @RKCS
1180  010540  100375                          BPL     2$
1181  010542  022710  177777                  CMP     #-1,(R0)
1182  010546  001363                          BNE     3$
1183  010550  104401  010556                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1184  010554  000432                          BR      64$             ;;GET OVER THE ASCIZ
1185                                  ;;65$:  .ASCIZ  <15><12>/SET SW0 TO SW7 TO SELECT DRIVES FOR TEST AND CONT/
1186  010642                          64$:
1187  010642  104401  010650                  TYPE    ,67$            ;;TYPE ASCIZ STRING
1188  010646  000426                          BR      66$             ;;GET OVER THE ASCIZ
1189                                  ;;67$:  .ASCIZ  <15><12>/RESET SW0 TO SW7 TO TEST ALL AVAIL DRIVES/
1190  010724                          66$:
1191  010724  022737  000176  001140  7$:     CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER IN USE ?
1192  010732  001002                          BNE     9$              ;BR IF NOT
1193  010734  104405                          GTSWR                   ;REQUEST NEW CONTENTS FOR SWITCH REG
1194  010736  000401                          BR      8$              ;CONTINUE
1195  010740  000000          9$:     HALT                            ;WAIT FOR OPERATOR TO ENTER NEW SWR VALUE
1196  010742  117704  170172          8$:     MOVB    @SWR,R4         ;SW0 TO SW7 TO R4
1197  010746  001413                          BEQ     4$              ;NONE SET, SO TEST ALL
1198  010750  012700  020614                  MOV     #DRIV0,R0       ;TABLE TO STORE DRIVE ADDRS
1199  010754  005001                          CLR     R1              ;ADDR OF DRIVE
1200  010756  006004          6$:     ROR     R4                      ;NEXT SWITCH TO CARRY
1201  010760  103001                          BCC     5$              ;SWITCH NOT SET
1202  010762  010120                          MOV     R1,(R0)+        ;SWITCH SET, SO MOVE ADDR TO TABLE
1203  010764  062701  020000          5$:     ADD     #20000,R1       ;ADDR OF NEXT DRIVE
1204  010770  001372                          BNE     6$              ;ALL DONE WHEN ZERO
1205  010772  012720  177777                  MOV     #-1,(R0)+       ;TABLE TERMINATOR
1206  010776  105737  001326          4$:     TSTB    OSPFLG          ;TYPED ONCE?
1207  011002  001165                          BNE     RWSRDY          ;IF YES, DONT RETYPE
1208  011004  104401  011012                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1209  011010  000432                          BR      68$             ;;GET OVER THE ASCIZ
```



```
1210                                  ;;69$:  .ASCIZ  <15><12>/TOGGLE THE "FIRST CYLINDER ADDRESS" (OUTER LIMIT)/
1211  011076                          68$:
1212  011076  104401  011104                  TYPE    ,71$            ;;TYPE ASCIZ STRING
1213  011102  000441                          BR      70$             ;;GET OVER THE ASCIZ
1214                                  ;;71$:  .ASCIZ  <15><12>/INTO THE LOW BYTE (BIT0-7) OF THE SWITCH REGISTER AND THE "LAST
1215  011206                          70$:
1216  011206  104401  011214                  TYPE    ,73$            ;;TYPE ASCIZ STRING
1217  011212  000430                          BR      72$             ;;GET OVER THE ASCIZ
1218                                  ;;73$:  .ASCIZ  <15><12>/CYLINDER ADDRESS" (INNER LIMIT) INTO THE HIGH/
1219  011274                          72$:
1220  011274  104401  011302                  TYPE    ,75$            ;;TYPE ASCIZ STRING
1221  011300  000424                          BR      74$             ;;GET OVER THE ASCIZ
1222                                  ;;75$:  .ASCIZ  <15><12>/       BYTE (BIT8-15), THEN PRESS CONTINUE./
1223  011352                          74$:
1224  011352  105237  001326                  INCB    OSPFLG
1225  011356  032777  000100  170000  RWSRDY: BIT     #100,@RKDS      ;"READ/WRITE/SEEK READY" BIT SET?
1226  011364  001774                          BEQ     RWSRDY          ;IF NO WAIT! (IF HUNG RUN STATIC)
1227  011366  022737  000176  001140  TRYAGN: CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER  IN USE ?
1228  011374  001002                          BNE     1$              ;BR IF NOT
1229  011376  104405                          GTSWR                   ;GET NEW CONTENTS
1230  011400  000401                          BR      CONTIN          ;CONTINUE
1231  011402  000000          1$:     HALT
1232  011404  012777  000001  167756  CONTIN: MOV     #1,@RKCS        ;CONTROL RESET
1233  011412  105777  167752                  TSTB    @RKCS
1234  011416  100375                          BPL     .-4
1235  011420  013705  001140                  MOV     SWR,R5          ;GET THE ADDRESS OF THE SWITCH REG. TO R5
1236  011424  112501          1$:     MOVB    (R5)+,R1                ;GET A BYTE TO R1
1237  011426  042701  177400                  BIC     #177400,R1      ;CLEAR THE UNUSED BITS
1238  011432  022701  000312                  CMP     #312,R1         ;IS ADDRESS LEGAL?
1239  011436  100034                          BPL     2$              ;BRANCH IF YES
1240  011440  104401  011446                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1241  011444  000430                          BR      64$             ;;GET OVER THE ASCIZ
1242                                  ;;65$:  .ASCIZ  <15><12>/INVALID ADDRESS IN SWITCH REGISTER TRY AGAIN/
1243  011526                          64$:
1244  011526  000717                          BR      TRYAGN          ;GO BACK FOR NEW PARAMETERS
1245  011530  006101          2$:     ROL     R1                      ;ROTATING THIS REGISTER
1246  011532  006101                  ROL     R1                      ;MAKES THE BYTE FROM THE
1247  011534  006101                  ROL     R1                      ;SWITCH REGISTER LINE UP
1248  011536  006101                  ROL     R1                      ;WITH THE CYLINDER BITS OF
1249  011540  006101                  ROL     R1                      ;THE RKDA REGISTER.
1250  011542  042701  160037                  BIC     #160037,R1      ;CLEAR THE UNUSED BITS
1251  011546  032705  000001                  BIT     #1,R5           ;IS THIS THE LOW BYTE?
1252  011552  001003                          BNE     3$              ;BRANCH IF YES
1253  011554  010137  001402                  MOV     R1,@#SEEKI      ;STORE THE INNER LIMIT OF THE SEEK
1254  011560  000403                          BR      SEKSET          ;START SEEKS
1255  011562  010137  001404          3$:     MOV     R1,@#SEEKO      ;STORE THE OUTER LIMIT OF THE SEEK
1256  011566  000716                          BR      1$              ;GO BACK AND GET HIGH BYTE.
1257
1258  011570  012703  000050          SEKSET: MOV     #50,R3          ;GET THE NUMBER OF SEEK CYCLES TO R3
1259  011574  013704  001404                  MOV     @#SEEKO,R4      ;ADD THE OUTER LIMIT CYLINDER ADDRESS
1260  011600  013705  001402                  MOV     @#SEEKI,R5      ;ADD THE INNER LIMIT CYLINDER ADDRESS
1261
1262  011604  012700  020614          LDSEEK: MOV     #DRIV0,R0       ;INITIALIZE POINTER
1263  011610  010477  167582          5$:     MOV     R4,@RKDA        ;GET INNER LIMIT CYL ADRES
1264  011614  052077  167556                  BIS     (R0)+,@RKDA     ;SET DRIVE ADRES
1265  011620  012777  000011  167542          MOV     #11,@RKCS       ;ISSUE SEEK+GO! (FOR INNER LIMIT)
```

```
1266  011626  004737  005614              JSR     PC,SMTME
1267  011632  105777  167532        3$:   TSTB    @RKCS
1268  011636  100375              BPL     3$
1269
1270  011640  022710  177777              CMP     #-1,(R0)        ;ALL DRIVES DONE?
1271  011644  001361              BNE     5$              ;NO
1272  011646  012700  020614              MOV     #DRIV0,R0
1273  011652  012077  167520        6$:   MOV     (R0)+,@RKDA     ;ADRES THE DRIVE
1274  011656  032777  000100  167500  7$:   BIT     #RWS,@RKDS      ;SEEK DONE?
1275  011664  001774              BEQ     7$              ;NO, WAIT
1276  011666  022710  177777              CMP     #-1,(R0)        ;ALL DRIVES DONE?
1277  011672  001367              BNE     6$              ;NO
1278
1279  011674  012700  020614              MOV     #DRIV0,R0
1280  011700  010577  167472        8$:   MOV     R5,@RKDA        ;SET OUTER CYL ADRES
1281  011704  052077  167466              BIS     (R0)+,@RKDA     ;SET DRIVE ADRES
1282  011710  012777  000011  167452      MOV     #11,@RKCS       ;ISSUE SEEK+GO! (FOR OUTER LIMIT)
1283  011716  004737  005614              JSR     PC,SMTME
1284  011722  105777  167442        9$:   TSTB    @RKCS
1285  011726  100375              BPL     9$
1286
1287  011730  022710  177777              CMP     #-1,(R0)        ;ALL DONE?
1288  011734  001361              BNE     8$              ;NO
1289
1290  011736  012700  020614              MOV     #DRIV0,R0
1291  011742  012077  167430        10$:  MOV     (R0)+,@RKDA     ;SET DRIVE ADRES
1292  011746  032777  000100  167410  11$:  BIT     #RWS,@RKDS              ;SEEK DONE?
1293  011754  001774              BEQ     11$
1294  011756  022710  177777              CMP     #-1,(R0)        ;ALL DONE?
1295  011762  001367              BNE     10$
1296  011764  005303              DEC     R3              ;DONE 50 SEEK CYCLES (100 SEEKS)
1297  011766  001306              BNE     LDSEEK          ;IF NO BRANCH (KEEP CYCLING)
1298  011770  000605              BR      CONTIN          ;CHECK SWR FOR CHANGE AND CONTINUE
1299
1300
1301
1302
1303
1304                                    .SBTTL  FORMATTER-SURFACE VERIFIER
1305
1306  011772                    BAD.IN:
1307  011772  104401  012000              TYPE    ,65$            ;;TYPE ASCIZ STRING
1308  011776  000401              BR      64$             ;;GET OVER THE ASCIZ
1309                            ;;65$:  .ASCIZ  /?/
1310  012002                    64$:
1311  012002                    SECT.1:
1312  012002  104401  012010              TYPE    ,65$            ;;TYPE ASCIZ STRING
1313  012006  000441              BR      64$             ;;GET OVER THE ASCIZ
1314                            ;;65$:  .ASCIZ  <15><12>/FORMATTER-SURFACE VERIFIER,SET SW REG FOR DRV #'S. PRESS CONT./
1315  012112                    64$:
1316  012112  022737  000176  001140        CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER IN USE ?
1317  012120  001002              BNE     1$              ;BR IF NOT
1318  012122  104405              GTSWR                   ;GET SWITCH REGISTER VALUE
1319  012124  000401              BR      2$              ;CONTINUE
1320  012126  000000        1$:   HALT                    ;WAIT FOR 'CONTINUE'
1321  012130  012737  000001  020444  2$:   MOV     #1,SHFCNT       ;SET SHIFT COUNT
```



```
1322  012136  005037  020446        CLR     DRVCNT          ;CLEAR DRIVE COUNT
1323  012142  033777  020444  166770  S13:  BIT     SHFCNT,@SWR     ;IS THIS SW SET?
1324  012150  001011              BNE     S10             ;YES FORMAT THIS DRIVE
1325  012152  006337  020444        S11:  ASL     SHFCNT
1326  012156  005237  020446              INC     DRVCNT
1327  012162  022737  000010  020446      CMP     #10,DRVCNT      ;ALL DONE?
1328  012170  001561              BEQ     GD1
1329  012172  000763              BR      S13
1330  012174  013700  020446        S10:  MOV     DRVCNT,R0
1331  012200  104401  001161              TYPE    ,$CRLF
1332  012204  104401  020450              TYPE    ,EM1            ;TYPE 'DRIVE'
1333  012210  010046              MOV     R0,-(SP)        ;TYPE DRIVE #
1334  012212  104402              TYPOC
1335  012214  000300              SWAB    R0
1336  012216  006100              ROL     R0
1337  012220  006100              ROL     R0
1338  012222  006100              ROL     R0
1339  012224  006100              ROL     R0
1340  012226  006100              ROL     R0
1341  012230  010037  001352              MOV     R0,@#DSKTMP
1342  012234  012737  000000  001422      MOV     #0,ERRWF
1343  012242  012737  000000  001424      MOV     #0,ERRRF        ;CLEAR OUT ERROR COUNTS.
1344  012250  012737  000000  001426      MOV     #0,ERRRFC
1345  012256  012737  000000  001430      MOV     #0,ERRWCH
1346  012264  012737  000000  001432      MOV     #0,ERRWCS
1347  012272  012737  177750  001416  S12:  MOV     #-24.,RWC       ;SET WORD COUNT FOR READ FORMAT.
1348  012300  012737  164000  001412      MOV     #-6144.,WC      ;SET UP WORD COUNT FOR WRITES.
1349  012306  012737  000000  001420      MOV     #0,EXTR         ;CLEAR EXTRA BIT FOR 12 SECTOR PACK.
1350  012314  012701  177772        COMMON: MOV     #-6.,R1         ;SET UP LOOP COUNT FOR THE CLEANER.
1351  012320  012777  014500  167050  COM:  MOV     #14500,@RKDA    ;SET UP FOR A SEEK TO 202.
1352  012326  053777  001352  167042      BIS     @#DSKTMP,@RKDA
1353  012334  105777  167030        1$:   TSTB    @RKCS           ;IS THE CONTROLER READY?
1354  012340  100375              BPL     1$              ;NO SO WAIT.
1355  012342  012777  000011  167020      MOV     #11,@RKCS       ;DO THE SEEK.
1356  012350  032777  000100  167006  2$:   BIT     #BIT6,@RKDS     ;IS THE SEEK DONE?
1357  012356  001774              BEQ     2$              ;NO SO WAIT.
1358  012360  105777  167004        3$:   TSTB    @RKCS           ;IS CONTROLER READY?
1359  012364  100375              BPL     3$              ;NO
1360  012366  012777  000015  166774      MOV     #15,@RKCS       ;DO A DRIVE RESET.
1361  012374  032777  000100  166762  4$:   BIT     #BIT6,@RKDS     ;IS DRIVE RESET DONE.
1362  012402  001774              BEQ     4$              ;NO
1363  012404  005201              INC     R1              ;COUNT THE CLEANER LOOP.
1364  012406  001344              BNE     COM             ;MORE TO GO.
1365  012410  012737  000000  001410      MOV     #0,DA           ;START OUT AT CYL. 0.
1366  012416  012777  177777  166762  NEXT:  MOV     #177777,@BA     ;PUT ALL ONE'S IN BUFFER.
1367  012424  004137  012540              JSR     R1,IO           ;GO DO THE DISK THING.
1368  012430  012777  000000  166750      MOV     #0,@BA          ;PUT ALL ZERO'S IN BUFFER.
1369  012436  004137  012540              JSR     R1,IO           ;GO DO IT AGIAN.
1370  012442  012777  125252  166736      MOV     #125252,@BA     ;PUT A ALT. PATTERN IN BUFFER.
1371  012450  004137  012540              JSR     R1,IO           ;ONCE MORE.
1372  012454  005177  166726              COM     @BA             ;COMPLEMENT THE LAST PATTERN.
1373  012460  004137  012540              JSR     R1,IO           ;AND AGAIN.
1374  012464  062737  000040  001410      ADD     #40,DA          ;INCREMENT TO THE NEXT CYL.
1375  012472  022737  014540  001410      CMP     #14540,DA       ;ARE WE DONE WITH THIS ONE?
1376  012500  001346              BNE     NEXT            ;NO SO DO THE NEXT CYL.
1377  012502                    GOOT:
```

```
1378  012502  104401  012510          TYPE    ,65$            ;;TYPE ASCIZ STRING
1379  012506  000411                  BR      64$             ;;GET OVER THE ASCIZ
1380                                  ;;65$:  .ASCIZ  <CR><LF>/PACK GOOD    /
1381  012532                          64$:
1382  012532  000607                  BR      S11
1383  012534  000137  001440          GD1:    JMP     @#START         ;RESTART
1384                                  ;
1385                                  ;
1386                                  ;DISK I/O SUBROUTINE.
1387                                  ;
1388                                  ;
1389                                  ;SET UP FOR A WRITE/FORMAT.
1390  012540  013777  001412  166624  IO:     MOV     WC,@RKWC        ;SET UP THE WORD COUNT REG.
1391  012546  013777  001410  166622          MOV     DA,@RKDA        ;SET UP THE DISK ADDRESS.
1392  012554  053777  001352  166614          BIS     @#DSKTMP,@RKDA  ;SET THE UNIT NUMBER UP.
1393  012562  013777  001406  166604          MOV     BA,@RKBA        ;SET UP THE BUSS ADDRESS.
1394  012570  012777  000000  166572          MOV     #0,@RKCS        ;CLEAR THE CONTROL REG. FOR SET UP.
1395  012576  052777  006000  166564          BIS     #BIT10+BIT11,@RKCS ;SET FORMAT&INHIBT INC. BITS.
1396  012604  052777  000002  166556          BIS     #BIT1,@RKCS     ;SET UP WRITE FUN.
1397  012612  053777  001420  166550          BIS     EXTR,@RKCS      ;SET UP 12OR16 SECTOR PACK.
1398  012620  052777  000001  166542          BIS     #BIT0,@RKCS     ;GO DO THE WRITE FORMAT.
1399  012626  105777  166536          1$:     TSTB    @RKCS           ;IS WRITE FORMAT DONE?
1400  012632  100375                  BPL     1$              ;NO SO WAIT.
1401  012634  005777  166530          TST     @RKCS           ;WAS THERE A ERROR?
1402  012640  100541                  BMI     WFERR           ;YES GO SERVICE IT.
1403  012642  012737  000000  001422  MOV     #0,ERRWF        ;CLEAR OUT THE ERROR COUNTER.
1404                                  ;
1405                                  ;SET UP FOR A READ/FORMAT
1406                                  ;
1407  012650  013777  001416  166514          MOV     RWC,@RKWC       ;SET UP WORD COUNT REG.
1408  012656  013777  001410  166512          MOV     DA,@RKDA        ;SET UP DISK ADDRESS.
1409  012664  053777  001352  166504          BIS     @#DSKTMP,@RKDA  ;SET THE UNIT NUMBER .
1410  012672  013777  001414  166474          MOV     RBA,@RKBA       ;SET UP THE BUSS ADDRESS.
1411  012700  012777  000000  166462          MOV     #0,@RKCS        ;CLEAR THE CONTROL REG.
1412  012706  052777  002000  166454          BIS     #BIT10,@RKCS    ;SET THE FORMAT BIT.
1413  012714  052777  000004  166446          BIS     #BIT2,@RKCS     ;SET UP READ FUN.
1414  012722  053777  001420  166440          BIS     EXTR,@RKCS      ;SET UP 12 OR 16 SECTOR PACK.
1415  012730  052777  000001  166432          BIS     #BIT0,@RKCS     ;GO DO THE READ FORMAT.
1416  012736  105777  166426          2$:     TSTB    @RKCS           ;IS THE READ FORMAT DONE?
1417  012742  100375                  BPL     2$              ;NO SO WAIT.
1418  012744  032777  040000  166416          BIT     #BIT14,@RKCS    ;WAS TRERE A ERROR?
1419  012752  001134                  BNE     RFERR           ;YES GO SERVICE IT.
1420  012754  012737  000000  001424          MOV     #0,ERRRF        ;CLEAR OUT THE ERROR COUNT.
1421                                  ;
1422                                  ;SET UP FOR A WRITE CHECK.
1423                                  ;
1424  012762  013777  001412  166402          MOV     WC,@RKWC        ;SET UP WORD COUNT REG.
1425  012770  013777  001410  166400          MOV     DA,@RKDA        ;SET UP DISK ADDRESS.
1426  012776  053777  001352  166372          BIS     @#DSKTMP,@RKDA  ;SET UP THE UNIT NUMBER
1427  013004  013777  001406  166362          MOV     BA,@RKBA        ;SET UP BUSS ADDRESS.
1428  013012  012777  000000  166350          MOV     #0,@RKCS        ;CLEAR THE CONTROL REG.
1429  013020  052777  004400  166342          BIS     #BIT11+BIT8,@RKCS ;SET INHIBIT INCR.&STOP ON SOFT ERROR BITS.
1430  013026  053777  001420  166334          BIS     EXTR,@RKCS      ;SET 12 OR 16 SECTOR PACK.
1431  013034  052777  000006  166326          BIS     #BIT1+BIT2,@RKCS ;SET UP WRITE CHECK FUN.
1432  013042  052777  000001  166320          BIS     #BIT0,@RKCS     ;GO DO THE WRITE CHECK.
1433                                  ;
```



```
1434                                  ;CHECK HEADERS READ BY THE READ/FORMAT.
1435                                  ;
1436  013050  013703  001416          MOV     RWC,R3          ;PUT NUMBER OF WORDS TO
1437  013054  005403                  NEG     R3              ;CHECK IN REG 3.
1438  013056  063703  001414          ADD     RBA,R3          ;SET REG 3 TO THE LAST WORD TO BE CHECKED.
1439  013062  013702  001414          MOV     RBA,R2          ;SET REG 2 TO STARTING ADD. OF BUFF.
1440  013066  023722  001410  MORE:   CMP     DA,(2)+         ;CHECK THAT HEADER IS RIGHT.
1441  013072  001073                  BNE     RFCERR          ;THIS HEADER WAS WRONG GO SERVICE IT.
1442  013074  020302                  CMP     R3,R2           ;ARE WE DONE?
1443  013076  001373                  BNE     MORE            ;NO GO CHECK THE NEXT ONE.
1444  013100  012737  000000  001426  MOV     #0,ERRRFC       ;CLEAR OUT THE ERROR COUNT.
1445                                  ;
1446                                  ;LETS CHECK ON THE WRITE CHECK WE STARTED.
1447                                  ;
1448  013106  105777  166256          1$:     TSTB    @RKCS
1449  013112  100375                  BPL     1$              ;THE CONTROLER IS STILL BUSY.
1450  013114  005777  166250          TST     @RKCS           ;WAS THERE A ERROR?
1451  013120  100407                  BMI     WCERRR          ;YES GO SERVICE IT.
1452  013122  012737  000000  001430  MOV     #0,ERRWCH       ;CLEAR OUT THE
1453  013130  012737  000000  001432  MOV     #0,ERRWCS       ;ERROR COUNTERS.
1454  013136  000201                  RTS     R1              ;RETURN TO THE MAIN LINE.
1455  013140  000137  013624  WCERRR: JMP     WCERR
1456                                  ;
1457                                  ;ERRORS FOR WRITE FORMAT.
1458                                  ;
1459  013144  005237  001422  WFERR:  INC     ERRWF           ;ADD ONE TO THE ERROR COUNT.
1460  013150  022737  000004  001422  CMP     #4,ERRWF        ;HAS IT HAPPEND 4 TIMES ON THIS CYL.
1461  013156  001016                  BNE     RETRY           ;NO.
1462  013160                          SYSER:
1463  013160  104401  013166          TYPE    ,65$            ;;TYPE ASCIZ STRING
1464  013164  000410                  BR      64$             ;;GET OVER THE ASCIZ
1465                                  ;;65$:  .ASCIZ  <15><12>/SYSTEM ERROR/
1466  013206                          64$:
1467  013206  000000                  HALT                    ;LET THE TECH. BREATH.
1468  013210  000137  001440          JMP     START           ;RESTART THE TEST.
1469  013214  012777  000000  166146  RETRY:  MOV     #0,@RKCS        ;CLEAR OUT THE CONTROL REG.
1470  013222  012777  000015  166140          MOV     #15,@RKCS       ;DO A DRIVE RESET.
1471  013230  032777  000100  166126  1$:     BIT     #BIT6,@RKDS     ;IS IT DONE.
1472  013236  001774                  BEQ     1$              ;NO SO WAIT.
1473  013240  000137  012540          JMP     IO              ;TRY AGAIN.
1474                                  ;
1475                                  ;ERRORS FOR READ/FORMAT.
1476                                  ;
1477  013244  005237  001424  RFERR:  INC     ERRRF           ;ADDONE TO ERROR COUNT.
1478  013250  022737  000004  001424  CMP     #4,ERRRF        ;HAS IT HAPPEND 4 TIMES ON THIS CYL?
1479  013256  001356                  BNE     RETRY           ;NO DO IT AGAIN.
1480  013260  000737                  BR      SYSER           ;YES SO TELL HIM SO.
1481                                  ;
1482                                  ;READ/FORMAT ERRORS FOUND BY SOFTWARE CHECKS.
1483                                  ;
1484  013262  005237  001426  RFCERR: INC     ERRRFC          ;ADD ONE TO ERROR COUNT.
1485  013266  105777  166076  1$:     TSTB    @RKCS           ;WAIT FOR THE WRITE CHECK.
1486  013272  100375                  BPL     1$
1487  013274  022737  000004  001426  CMP     #4,ERRRFC       ;IS IT 4?
1488  013302  001401                  BEQ     FAILED          ;PUT OUT FAILED MESSAGE.
1489  013304  000743                  BR      RETRY           ;NO SO GO TRY IT AGAIN.
```

```
 1490  013306  042777  000037  166062  FAILED: BIC     #37,@RKDA       ;PUT WHICH SECTORS HEADER
 1491  013314  042702  177740                  BIC     #177740,R2
 1492  013320  060277  166052                  ADD     R2,@RKDA        ;FAILED IN RKDA FOR THE MESSAGE.
 1493  013324                          FAIL:
 1494  013324  104401  013332                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 1495  013330  000417                          BR      64$             ;;GET OVER THE ASCIZ
 1496                                  ;;65$:  .ASCIZ  <15><12>/PACK FAILED AT (IN OCTAL) /
 1497  013370                          64$:
 1498  013370  017701  166002                  MOV     @RKDA,R1        ;GENERAT THE CYL,SECTOR,SURFACE
 1499  013374  010102                          MOV     R1,R2   ;MESSAGE FROM RKDA
 1500  013376  042702  177770                  BIC     #177770,R2
 1501  013402  062702  000260                  ADD     #260,R2
 1502  013406  110237  013565                  MOVB    R2,SEC+1
 1503  013412  004337  013612                  JSR     R3,SHF3
 1504  013416  042702  177776                  BIC     #177776,R2
 1505  013422  062702  000260                  ADD     #260,R2
 1506  013426  110237  013564                  MOVB    R2,SEC
 1507  013432  004337  013616                  JSR     R3,SHF1
 1508  013436  042702  177776                  BIC     #177776,R2
 1509  013442  062702  000260                  ADD     #260,R2
 1510  013446  110237  013576                  MOVB    R2,SUR
 1511  013452  004337  013616                  JSR     R3,SHF1
 1512  013456  042702  177770                  BIC     #177770,R2
 1513  013462  062702  000260                  ADD     #260,R2
 1514  013466  110237  013554                  MOVB    R2,CYL+2
 1515  013472  004337  013612                  JSR     R3,SHF3
 1516  013476  042702  177770                  BIC     #177770,R2
 1517  013502  062702  000260                  ADD     #260,R2
 1518  013506  110237  013553                  MOVB    R2,CYL+1
 1519  013512  004337  013612                  JSR     R3,SHF3
 1520  013516  042702  177774                  BIC     #177774,R2
 1521  013522  062702  000260                  ADD     #260,R2
 1522  013526  110237  013552                  MOVB    R2,CYL
 1523  013532  104401  013542                  TYPE    ,1$             ;TYPE OUT THE GENERATED MESSAGE
 1524  013536  000137  013604                  JMP     STOP            ;BYPASS THE INLINE MESSAGE
 1525  013542  005015  054503  027114  1$:     .ASCII  <15><12>/CYL.  /
 1526  013550  020040
 1527  013552  030060  020060          CYL:    .ASCII  /000 /
 1528  013556  042523  027103  020040          .ASCII  /SEC.  /
 1529  013564  030060     040          SEC:    .ASCII  /00 /
 1530  013567     123  051125  027106          .ASCII  /SURF.  /
 1531  013574  020040
 1532  013576  020060  005015     000  SUR:    .ASCIZ  /0 /<15><12>
 1533          013604                          .EVEN
 1534  013604  000000                  STOP:   HALT                    ;LET OPER DO HIS THING.
 1535  013606  000137  001440                  JMP     START           ;RESTART THE TEST.
 1536                                  ;
 1537                                  ;SHIFT SUBROUTINES.
 1538                                  ;
 1539  013612  006201                  SHF3:   ASR     R1              ;HERE FOR A SHIFT OF 3.
 1540  013614  006201                  SHF2:   ASR     R1              ;HERE FOR A SHIFT OF 2.
 1541  013616  006201                  SHF1:   ASR     R1              ;HERE FOR A SHIFT OF 1.
 1542  013620  010102                          MOV     R1,R2           ;PUT RESULTES IN THE WORKING REG.
 1543  013622  000203                          RTS     R3
 1544                                  ;
 1545                                  ;ERRORS FOR WRITE CHECK.
```



```
 1546                                  ;
 1547  013624  032777  040000  165536  WCERR:  BIT     #BIT14,@RKCS    ;WAS IT A HARD ERROR.
 1548  013632  001010                          BNE     WCHERR          ;YES GO PROSSES IT.
 1549  013634  005237  001432                  INC     ERRWCS          ;ADD 1 TO THE SOFT ERROR COUNT.
 1550  013640  022737  000004  001432          CMP     #4,ERRWCS       ;HAS THERE BEEN 4 OF THEM?
 1551  013646  001626                          BEQ     FAIL            ;YES PUT OUT FAILED MESSAGE.
 1552  013650  000137  012540                  JMP     IO              ;NO SO TRY AGAIN.
 1553  013654  005237  001430          WCHERR: INC     ERRWCH          ;ADD 1 TO THE HARD ERROR COUNT.
 1554  013660  022737  000004  001430          CMP     #4,ERRWCH       ;HAS THERE BEEN 4 OF THEM?
 1555  013666  001402                          BEQ     SYSERR          ;YES PUT OUT SYSTEM ERROR MESSAGE.
 1556  013670  000137  013214                  JMP     RETRY           ;NO SO TRY AGAIN.
 1557  013674  000137  013160          SYSERR: JMP     SYSER
 1558                                  ;
 1559                                  ;BUFFERS.
 1560                                  ;
 1561  013700  000000                  BUFF:   .WORD   0
 1562  013702  000030                  RBUFF:  .BLKW   30
 1563
 1564
 1565
 1566
 1567
 1568                                          .SBTTL  RK05 CONTROL PANEL TEST
 1569
 1570  013762                          BAD.ON:
 1571  013762  104401  013770                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 1572  013766  000401                          BR      64$             ;;GET OVER THE ASCIZ
 1573                                  ;;65$:  .ASCIZ  /?/
 1574  013772                          64$:
 1575  013772                          SECT.0:
 1576  013772  104401  014000                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 1577  013776  000424                          BR      64$             ;;GET OVER THE ASCIZ
 1578                                  ;;65$:  .ASCIZ  <15><12>/RK05 CONTROL PANEL TEST, WHICH DRIVE?/
 1579  014050                          64$:
 1580  014050  104407                          RDCHR
 1581  014052  011600                          MOV     (SP),R0
 1582  014054  104403                          TYPOS
 1583  014056     001                          .BYTE   1
 1584  014057     000                          .BYTE   0
 1585  014060  162700  000060                  SUB     #60,R0
 1586  014064  100736                          BMI     BAD.ON
 1587  014066  022700  000010                  CMP     #10,R0
 1588  014072  003733                          BLE     BAD.ON
 1589  014074  110037  001314                  MOVB    R0,@#DRIVE
 1590  014100  000300                          SWAB    R0
 1591  014102  006100                          ROL     R0
 1592  014104  006100                          ROL     R0
 1593  014106  006100                          ROL     R0
 1594  014110  006100                          ROL     R0
 1595  014112  006100                          ROL     R0
 1596  014114  010037  001352                  MOV     R0,@#DSKTMP
 1597  014120  104401  014128                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 1598  014124  000414                          BR      66$             ;;GET OVER THE ASCIZ
 1599                                  ;;67$:  .ASCIZ  <15><12>/MOUNT PACK ON DRIVE#/
 1600  014156                          66$:
 1601  014156  013746  001314                  MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
```

```
1602  014162  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
1603  014164     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
1604  014165     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
1605  014166  104401  014174                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1606  014172  000425                          BR      68$             ;;GET OVER THE ASCIZ
1607                                  ;;69$:  .ASCIZ  <15><12>/PLACE DRIVE IN RUN ;SHOULD SEE THE RUN./
1608  014246                          68$:
1609  014246  104401  014254                  TYPE    ,71$            ;;TYPE ASCIZ STRING
1610  014252  000423                          BR      70$             ;;GET OVER THE ASCIZ
1611                                  ;;71$:  .ASCIZ  <15><12>/POWER, AND ON CYLINDER LAMPS LIGHT./
1612  014322                          70$:
1613  014322  104401  014330                  TYPE    ,73$            ;;TYPE ASCIZ STRING
1614  014326  000425                          BR      72$             ;;GET OVER THE ASCIZ
1615                                  ;;73$:  .ASCIZ  <15><12>/MAKE DRIVE WRITE ENABLE PRESS CONTINUE/
1616  014402                          72$:
1617  014402  000000                          HALT
1618  014404  004737  016260                  JSR     PC,WRRDCK       ;GO WRITE 0'S, RD 0'S, CHECK
1619  014410  104401  014416                  TYPE    ,75$            ;;TYPE ASCIZ STRING
1620  014414  000430                          BR      74$             ;;GET OVER THE ASCIZ
1621                                  ;;75$:  .ASCIZ  <15><12><15><12>/WRITE PROTECT THE DRIVE THEN PRESS CONTINUE/
1622  014476                          74$:
1623  014476  000000                          HALT
1624  014500  004737  016470                  JSR     PC,WRPRO        ;GO TRY OVERWRITE
1625  014504  104401  014512                  TYPE    ,77$            ;;TYPE ASCIZ STRING
1626  014510  000426                          BR      76$             ;;GET OVER THE ASCIZ
1627                                  ;;77$:  .ASCIZ  <15><12><15><12>/CLEAR WRITE PROTECT THEN PRESS CONTINUE/
1628  014566                          76$:
1629  014566  000000                          HALT
1630  014570  004737  016260                  JSR     PC,WRRDCK       ;GO WRITE 0'S, RD 0'S, CHECK
1631  014574  013777  001352  164574          MOV     @#DSKTMP,@RKDA  ;SET UP DRIVE#
1632  014602  012777  000017  164560          MOV     #17,@RKCS       ;FUNCTION WRITE PROTECT
1633  014610  004737  005614                  JSR     PC,SMTME        ;GO WAIST TIME FOR RK11-C
1634  014614  105777  164550          1$:     TSTB    @RKCS           ;IS CONTROL READY SET
1635  014620  100375                          BPL     1$              ;IF NO, BRANCH
1636  014622  004737  016470                  JSR     PC,WRPRO        ;GO TRY OVERWRITE OF IERO'S
1637  014626                          PRTTWO:
1638  014626  104401  014634                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1639  014632  000431                          BR      64$             ;;GET OVER THE ASCIZ
1640                                  ;;65$:  .ASCIZ  <15><12><15><12>/CAUTION! TRY TO OPEN THE DOOR, DO NOT FORCE:/
1641  014716                          64$:
1642  014716  104401  014724                  TYPE    ,67$            ;;TYPE ASCIZ STRING
1643  014722  000414                          BR      66$             ;;GET OVER THE ASCIZ
1644                                  ;;67$:  .ASCIZ  <15><12>/DOOR SHOULD NOT OPEN!/
1645  014754                          66$:
1646  014754  104401  014762                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1647  014760  000420                          BR      68$             ;;GET OVER THE ASCIZ
1648                                  ;;69$:  .ASCIZ  <15><12>/PRESS CONTINUE WHEN FINISHED/
1649  015022                          68$:
1650  015022  000000                          HALT
1651  015024  104401  015032                  TYPE    ,71$            ;;TYPE ASCIZ STRING
1652  015030  000426                          BR      70$             ;;GET OVER THE ASCIZ
1653                                  ;;71$:  .ASCIZ  <15><12><15><12>/PUT DRIVE IN LOAD, WAIT FOR LOAD LIGHT/
1654  015106                          70$:
1655  015106  104401  015114                  TYPE    ,73$            ;;TYPE ASCIZ STRING
1656  015112  000420                          BR      72$             ;;GET OVER THE ASCIZ
1657                                  ;;73$:  .ASCIZ  <15><12>/PRESS CONTINUE WHEN FINISHED/
```



```
1658  015154                          72$:
1659  015154  000000                          HALT
1660  015156  013777  001352  164212          MOV     @#DSKTMP,@RKDA  ;SET UP DISK ADDRESS
1661  015164  012777  000015  164176          MOV     #15,@RKCS       ;ISSUE A DRIVE RESET
1662  015172  004737  005614                  JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1663  015176  105777  164166          1$:     TSTB    @RKCS           ;CONTROL READY SET?
1664  015202  100375                          BPL     1$              ;IF NO, BRANCH
1665  015204  032777  100200  164154          BIT     #100200,@RKER   ;DRE SET
1666  015212  001057                          BNE     2$              ;IF YES BRANCH
1667  015214  104401  015222                  TYPE    ,75$            ;;TYPE ASCIZ STRING
1668  015220  000427                          BR      74$             ;;GET OVER THE ASCIZ
1669                                  ;;75$:  .ASCIZ  <15><12>/DRE=BIT15 OF RKER DID NOT SET IF AN RK11-C/
1670  015300                          74$:
1671  015300  104401  015306                  TYPE    ,77$            ;;TYPE ASCIZ STRING
1672  015304  000422                          BR      76$             ;;GET OVER THE ASCIZ
1673                                  ;;77$:  .ASCIZ  <15><12>/OR BIT07 DID NOT SET IF AN RK11-D/
1674  015352                          76$:
1675  015352  032777  140000  164010  2$:     BIT     #140000,@RKCS   ;DID HARD ERROR ON ERROR SET
1676  015360  001015                          BNE     3$              ;BRANCH IF YES
1677  015362  104401  015370                  TYPE    ,79$            ;;TYPE ASCIZ STRING
1678  015366  000412                          BR      78$             ;;GET OVER THE ASCIZ
1679                                  ;;79$:  .ASCIZ  <15><12>/ERROR DID NOT SET/
1680  015414                          78$:
1681  015414  012777  000001  163746  3$:     MOV     #1,@RKCS        ;ISSUE A CONTROL RESET
1682  015422  004737  005614                  JSR     PC,SMTME        ;WAIST TIME
1683  015426  105777  163736          4$:     TSTB    @RKCS           ;CONTROL READY SET
1684  015432  100375                          BPL     4$              ;IF NO, BRANCH
1685  015434  032777  100000  163724          BIT     #BIT15,@RKER    ;'DRE' CLEAR
1686  015442  001425                          BEQ     5$              ;IF YES BRANCH
1687  015444  104401  015452                  TYPE    ,81$            ;;TYPE ASCIZ STRING
1688  015450  000422                          BR      80$             ;;GET OVER THE ASCIZ
1689                                  ;;81$:  .ASCIZ  <15><12>/CONTROL RESET DID NOT CLEAR 'DRE'/
1690  015516                          80$:
1691  015516  032777  140000  163644  5$:     BIT     #140000,@RKCS   ;ERROR BITS CLEAR
1692  015524  001431                          BEQ     X               ;IF YES BRANCH
1693  015526  104401  015534                  TYPE    ,83$            ;;TYPE ASCIZ STRING
1694  015532  000426                          BR      82$             ;;GET OVER THE ASCIZ
1695                                  ;;83$:  .ASCIZ  <15><12>/CONTROL RESET DID NOT CLEAR 'ERROR', RKCS/
1696  015610                          82$:
1697  015610                          X:
1698  015610  104401  015616                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1699  015614  000422                          BR      64$             ;;GET OVER THE ASCIZ
1700                                  ;;65$:  .ASCIZ  <15><12><15><12>/OPEN THE DOOR, PUT DRIVE IN RUN/
1701  015662                          64$:
1702  015662  104401  015670                  TYPE    ,67$            ;;TYPE ASCIZ STRING
1703  015666  000425                          BR      66$             ;;GET OVER THE ASCIZ
1704                                  ;;67$:  .ASCIZ  <15><12>/CAUTION! IF RUN LIGHT ON ERROR! DEPRESS/
1705  015742                          66$:
1706  015742  104401  015750                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1707  015746  000426                          BR      68$             ;;GET OVER THE ASCIZ
1708                                  ;;69$:  .ASCIZ  <15><12>/LOAD IMMEDIATELY, CONTINUE WHEN FINISHED/
1709  016024                          68$:
1710  016024  000000                  XX:     HALT
1711  016026  104401  016034                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1712  016032  000422                          BR      64$             ;;GET OVER THE ASCIZ
1713                                  ;;65$:  .ASCIZ  <15><12><15><12>/REMOVE THE PACK, CLOSE THE DOOR/
```

```
1714  016100                          64$:
1715  016100  104401  016106                  TYPE    ,67$            ;;TYPE ASCIZ STRING
1716  016104  000423                  BR      66$             ;;GET OVER THE ASCIZ
1717                          ;;67$:  .ASCIZ  <15><12>/PUT DRIVE IN RUN, DRIVE SHOULD NOT/
1718  016154                          66$:
1719  016154  104401  016162                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1720  016160  000423                  BR      68$             ;;GET OVER THE ASCIZ
1721                          ;;69$:  .ASCIZ  <15><12>/RUN...INTERCLOCKS HAVE BEEN CHECKED/
1722  016230                          68$:
1723  016230  104401  016236                  TYPE    ,71$            ;;TYPE ASCIZ STRING
1724  016234  000407                  BR      70$             ;;GET OVER THE ASCIZ
1725                          ;;71$:  .ASCIZ  <15><12>/DONE!      /
1726  016254                          70$:
1727  016254  000137  001440                  JMP     @#START
1728  016260  005037  001362          WRRDCK: CLR     @#PATTRN        ;MAKE A PATTERN OF ZERO'S
1729  016264  013777  001352  163104          MOV     @#DSKTMP,@RKDA  ;SET UP DRIVE ADDRESS
1730  016272  012777  000001  163070          MOV     #1,@RKCS        ;ISSUE A CONTROL RESET
1731  016300  004737  005614                  JSR     PC,SMTME
1732  016304  105777  163060          5$:     TSTB    @RKCS           ;CONTROL READY SET
1733  016310  100375                  BPL     5$              ;IF NO BRANCH
1734  016312  013777  001352  163056          MOV     @#DSKTMP,@RKDA  ;SET UP DRIVE ADDRESS
1735  016320  012777  001362  163046          MOV     #PATTRN,@RKBA   ;GET BUSS ADDRESS
1736  016326  013777  001344  163036          MOV     @#SECCNT,@RKWC  ;WORD COUNT 1 SECTOR
1737  016334  013777  001354  163026          MOV     @#WRITCS,@RKCS  ;IBA + WRITE + GO
1738  016342  004737  005614                  JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1739  016346  105777  163016          1$:     TSTB    @RKCS           ;CONTROL READY SET
1740  016352  100375                  BPL     1$              ;IF NO, BRANCH
1741  016354  013777  001352  163014          MOV     @#DSKTMP,@RKDA  ;SET UP RK REGISTERS
1742  016362  012777  007356  163004          MOV     #RDBUFF,@RKBA   ;TO READ ONE SECTOR
1743  016370  013777  001344  162774          MOV     @#SECCNT,@RKWC  ;TO THE READ BUFFER
1744  016376  013777  001356  162764          MOV     @#READCS,@RKCS
1745  016404  004737  005614                  JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1746  016410  105777  162754          2$:     TSTB    @RKCS           ;CONTROL READY SET
1747  016414  100375                  BPL     2$              ;IF NO, BRANCH
1748  016416  012704  007356                  MOV     #RDBUFF,R4      ;GET BUFFER TO R4
1749  016422  005005                  CLR     R5              ;SET UP TO COMPARE
1750  016424  020524                  3$:     CMP     R5,(R4)+        ;FOR ZERO'S
1751  016426  001414                  BEQ     4$              ;IF OK, BRANCH
1752  016430  104401  016436                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1753  016434  000410                  BR      64$             ;;GET OVER THE ASCIZ
1754                          ;;65$:  .ASCIZ  <15><12>/WRITE FAILED/
1755  016456                          64$:
1756  016456  000700                  BR      WRRDCK          ;GO BACK TRY AGAIN
1757  016460  022704  010356          4$:     CMP     #MANSEL,R4      ;DONE ALL CHECKS
1758  016464  001357                  BNE     3$              ;IF NO, BRANCH
1759  016466  000207                  RTS     PC              ;IF YES, RETURN
1760  016470  032777  000040  162666  WRPRO:  BIT     #BIT5,@RKDS     ;BIT 5 ON
1761  016476  001021                  BNE     1$              ;IF YES, BRANCH
1762  016500  104401  016506                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1763  016504  000416                  BR      64$             ;;GET OVER THE ASCIZ
1764                          ;;65$:  .ASCIZ  <15><12>/WPS=BIT5 OF RKDS NOT SET/
1765  016542                          64$:
1766  016542  012737  177777  001362  1$:     MOV     #177777,@#PATTRN ;GO LOAD ALL
1767  016550  013777  001352  162620          MOV     @#DSKTMP,@RKDA  ;RK REGISTERS
1768  016556  012777  001362  162610          MOV     #PATTRN,@RKBA   ;TO WRITE
1769  016564  013777  001344  162600          MOV     @#SECCNT,@RKWC  ;ALL ONES (WITH WRITE LOCK)
```



```
1770  016572  013777  001354  162570          MOV     @#WRITCS,@RKCS  ;OVER THE ZERO'S
1771  016600  004737  005614                  JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1772  016604  105777  162560          2$:     TSTB    @RKCS           ;CONTROL READY SET?
1773  016610  100375                  BPL     2$              ;IF NO, BRANCH
1774  016612  032777  020000  162548          BIT     #BIT13,@RKER    ;WLO BIT SET
1775  016620  001032                  BNE     3$              ;IF YES BRANCH
1776  016622  104401  016630                  TYPE    ,67$            ;;TYPE ASCIZ STRING
1777  016626  000427                  BR      66$             ;;GET OVER THE ASCIZ
1778                          ;;67$:  .ASCIZ  <15><12>/EXPECTED WLO=BIT13 OF RKER BUT DID NOT SET/
1779  016706                          66$:
1780  016706  012777  000001  162454  3$:     MOV     #1,@RKCS        ;DO A CONTROL RESET
1781  016714  004737  005614                  JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1782  016720  105777  162444          4$:     TSTB    @RKCS           ;CONTROL READY SET?
1783  016724  100375                  BPL     4$              ;IF NO BRANCH
1784  016726  032777  020000  162432          BIT     #BIT13,@RKER    ;WLO BIT CLEAR
1785  016734  001431                  BEQ     RDCHK0          ;IF YES, BRANCH
1786  016736  104401  016744                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1787  016742  000426                  BR      68$             ;;GET OVER THE ASCIZ
1788                          ;;69$:  .ASCIZ  <15><12>/CONTROL RESET DID NOT CLEAR 'WLO' OF RKER/
1789  017020                          68$:
1790  017020  013777  001352  162350  RDCHK0: MOV     @#DSKTMP,@RKDA  ;SET UP RK REGISTERS
1791  017026  012777  007356  162340          MOV     #RDBUFF,@RKBA   ;TO READ SECTOR 0,
1792  017034  013777  001344  162330          MOV     @#SECCNT,@RKWC  ;CYLINDER 0, HEAD 0
1793  017042  013777  001356  162320          MOV     @#READCS,@RKCS  ;TO ENSURE NO WRITE TOOK PLACE
1794  017050  004737  005614                  JSR     PC,SMTME        ;KILL TIME
1795  017054  105777  162310          3$:     TSTB    @RKCS
1796  017060  100375                  BPL     3$
1797  017062  012703  000005                  MOV     #5,R3
1798  017066  012704  007356                  MOV     #RDBUFF,R4      ;CHECK TO INSURE NO WRITE
1799  017072  005005                  CLR     R5              ;TOOK PLACE
1800  017074  020524                  1$:     CMP     R5,(R4)+        ;WITH WRITE LOCK
1801  017076  001474                  BEQ     2$
1802  017100  005303                  DEC     R3              ;DEC THE ERROR COUNT
1803  017102  001475                  BEQ     4$              ;IF ZERO BRANCH
1804  017104  104401  017112                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1805  017110  000422                  BR      64$             ;;GET OVER THE ASCIZ
1806                          ;;65$:  .ASCIZ  <15><12>/WRITE OCCURRED WITH WRITE PROTECT/
1807  017156                          64$:
1808  017156  005744                  TST     -(R4)
1809  017160  104401  017166                  TYPE    ,67$            ;;TYPE ASCIZ STRING
1810  017164  000410                  BR      66$             ;;GET OVER THE ASCIZ
1811                          ;;67$:  .ASCIZ  <15><12>/BUFFER ADDR=/
1812  017206                          66$:
1813  017206  010446                  MOV     R4,-(SP)
1814  017210  104403                  TYPOS
1815  017212     006                  .BYTE   6
1816  017213     001                  .BYTE   1
1817  017214  104401  017222                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1818  017220  000406                  BR      68$             ;;GET OVER THE ASCIZ
1819                          ;;69$:  .ASCIZ  /    EXPCTD=/
1820  017236                          68$:
1821  017236  010546                  MOV     R5,-(SP)
1822  017240  104404                  TYPON
1823  017242  104401  017250                  TYPE    ,71$            ;;TYPE ASCIZ STRING
1824  017246  000406                  BR      70$             ;;GET OVER THE ASCIZ
1825                          ;;71$:  .ASCIZ  /    RECVD=/
```

```
1826  017264                        70$:
1827  017264  012446                      MOV     (R4)+,-(SP)
1828  017266  104404                      TYPON
1829  017270  022704  010356        2$:   CMP     #MANSEL,R4      ;FINISHED ALL CHECKS
1830  017274  001277                      BNE     1$              ;IF NO, BRANCH
1831  017276  000207                4$:   RTS     PC              ;RETURN
1832
1833
1834
1835
```



```
1836                                ;THE FOLLOWING REVISION WAS MADE BY JIM KAPADIA
1837
1838                                        .SBTTL  CONTROL PANEL TEST # 2
1839
1840                                ;THIS IS THE ENTRY POINT INTO CONTROL PANEL TEST #2.  ALL
1841                                ;THE DRIVES THAT ARE PRESENT AND IN 'RDY' CONDITION ARE
1842                                ;REPORTED (ON LINE).
1843
1844  017300  000240                SECT.4: NOP
1845  017302  012777  000001  162060        MOV     #1,@RKCS
1846  017310  105777  162054        3$:   TSTB    @RKCS
1847  017314  100375                       BPL     3$
1848  017316  012700  020614               MOV     #DRIV0,R0
1849  017322  005001                       CLR     R1
1850  017324  005002                       CLR     R2
1851
1852  017326  010210                1$:   MOV     R2,(R0)         ;SET UP ADDRESS TABLE
1853  017330  010277  162042               MOV     R2,@RKDA        ;ADDRESS THE DRIVE
1854  017334  105777  162024               TSTB    @RKDS           ;IS IT PRESENT?
1855  017340  100021                       BPL     2$              ;NO
1856
1857  017342  104401  020450               TYPE    ,EM1            ;TYPE 'DRIVE'
1858  017346  010146                       MOV     R1,-(SP)        ;TYPE OUT DRIVE #
1859  017350  104402                       TYPOC
1860  017352  104401  020461               TYPE    ,EM2            ;TYPE 'ON LINE'
1861  017356  052710  000300               BIS     #BIT6+BIT7,(R0) ;SET BITS INDICATING THIS
1862                                                                ;DRIVE PRESENT
1863
1864  017362  012777  000015  162000        MOV     #15,@RKCS       ;ISSUE A DRIVE RESET
1865  017370  004737  005614               JSR     PC,SMTME        ;ALLOW SOME TIME
1866  017374  032777  000100  161762  4$:   BIT     #RWS,@RKDS      ;WAIT FOR RWS RDY
1867  017402  001774                       BEQ     4$
1868
1869  017404  005720                2$:   TST     (R0)+
1870  017406  005201                       INC     R1
1871  017410  062702  020000               ADD     #20000,R2       ;NXT DRIVE
1872  017414  001344                       BNE     1$              ;ALL DONE?
1873
1874  017416  104401  001161               TYPE    ,$CRLF
1875
1876                                ;THIS CODE CHECKS THE CONDITION OF 'DRY' BIT IN RKDS FOR EVERY
1877                                ;DRIVE.  IF 'DRY' IS SET DRIVE IS SAID TO BE 'ON LINE', OTHERWISE IT
1878                                ;IS OFFLINE.  IF THE 'DRY' BIT HAS CHANGED FROM LAST TIME, THEN
1879                                ;IT IS REPORTED.  IF THERE IS NO CHANGE NOTHING IS REPORTED.
1880
1881  017422  012700  020614        BEGCT:  MOV     #DRIV0,R0       ;INITIALIZE POINTERS
1882  017426  005001                       CLR     R1
1883
1884  017430  011077  161742        BEGCT1: MOV     (R0),@RKDA      ;ADDRESS A DRIVE
1885  017434  042777  017777  161734       BIC     #17777,@RKDA    ;MASK OUT NON DR# BITS
1886  017442  105777  161716               TSTB    @RKDS           ;IS THIS DRIVE ON LINE?
1887  017446  100044                       BPL     1$              ;NO
1888                                                                ;YES
1889  017450  105710                       TSTB    (R0)            ;WAS IT 'ON LINE' LAST TIME?
1890  017452  100454                       BMI     NXT1            ;YES, NO MESSAGE TO REPORT
1891  017454  052710  000200               BIS     #BIT7,(R0)      ;IT CHANGED FROM OFF LINE TO ON
```

```
1892  017460  104401  020450          TYPE    ,EM1            ;LINE, REPORT MESSAGE
1893  017464  010146                  MOV     R1,-(SP)
1894  017466  104402                  TYPOC
1895  017470  104401  020461          TYPE    ,EM2            ;TYPE 'ON LINE'
1896  017474  032777  000040  161662  BIT     #WPS,@RKDS      ;WRITE ENABLED?
1897  017502  001417                  BEQ     2$              ;YES, OK
1898  017504  104401  017512          TYPE    ,65$            ;;TYPE ASCIZ STRING
1899  017510  000414                  BR      64$             ;;GET OVER THE ASCIZ
1900                          ;;65$:  .ASCIZ  <15><12>/EROR,NOT WRT ENABLED/
1901  017542                  64$:
1902  017542  012777  000017  161620 2$:     MOV     #17,@RKCS       ;WRITE PROT THE DISK
1903  017550  105777  161614          3$:     TSTB    @RKCS
1904  017554  100375                  BPL     3$
1905  017556  000412                  BR      NXT1
1906
1907  017560  105710                  1$:     TSTB    (R0)            ;WAS THIS DRIVE OFF LINE LAST
1908  017562  100010                  BPL     NXT1            ;TIME?  BRNCH IF YES
1909  017564  104401  020450          TYPE    ,EM1            ;IF NOT, REPORT THE CHANGE
1910  017570  010146                  MOV     R1,-(SP)        ;TYPE DRIVE #
1911  017572  104402                  TYPOC
1912  017574  104401  020473          TYPE    ,EM3            ;TYPE 'OFF LINE'
1913  017600  042710  000200          BIC     #BIT7,(R0)      ;CLEAR BIT TO INDICATE THIS
1914                                                          ;DRIVE 'OFF LINE'
1915
1916                          ;THIS CODE CHECKS 'WPS' BIT FOR EVERY DRIVE THAT IS IN 'DRY'
1917                          ;CONDITION (ON LINE).  IT REPORTS ANY CHANGE IN THE CONDITION OF
1918                          ;THE 'WPS' BIT.  IF THERE WAS NO CHANGE FROM LAST TIME NOTHING
1919                          ;IS REPORTED.  AT THE TIME OF ENTRY R0 POINTS TO DRIVE FLAG.
1920
1921  017604  105777  161554  NXT1:   TSTB    @RKDS           ;IS THIS DRIVE PRESENT?
1922                                                          ;RKDA CONTAINS THE DRV #
1923  017610  100033                  BPL     NXT2            ;NO, SKIP CHECKING
1924
1925  017612  032777  000040  161544  BIT     #WPS,@RKDS      ;WPS BIT SET?
1926  017620  001014                  BNE     1$              ;YES
1927                                                          ;WPS BIT CLEAR
1928  017622  032710  000004          BIT     #BIT2,(R0)      ;WAS IT CLR LAST TIME ALSO?
1929  017626  001424                  BEQ     NXT2            ;YES, NOTHING TO REPORT.
1930                                                          ;WPS CHANGED FROM 'SET'
1931  017630  104401  020450          TYPE    ,EM1            ;TO 'CLR', REPORT IT
1932  017634  010146                  MOV     R1,-(SP)        ;TYPE DRIVE #
1933  017636  104402                  TYPOC
1934  017640  042710  000004          BIC     #BIT2,(R0)      ;INDICATE THAT 'WPS' IS CLEAR
1935  017644  104401  020521          TYPE    ,EM5            ;TYPE 'WPS CLEAR'
1936  017650  000413                  BR      NXT2
1937                                                          ;WPS BIT IS SET
1938  017652  032710  000004          1$:     BIT     #BIT2,(R0)      ;WAS IT SET LAST TIME ALSO?
1939  017656  001010                  BNE     NXT2            ;YES, NOTHING TO REPORT.
1940                                                          ;WPS CHANGED, FROM 'CLR' TO
1941                                                          ;'SET', REPORT THIS CHANGE
1942  017660  104401  020450          TYPE    ,EM1            ;TYPE 'DRIVE'
1943  017664  010146                  MOV     R1,-(SP)        ;TYPE DRIVE #
1944  017666  104402                  TYPOC
1945  017670  104401  020506          TYPE    ,EM4            ;TYPE 'WPS SET'
1946  017674  052710  000004          BIS     #BIT2,(R0)      ;SET FLAG BIT INDICATING WPS SET
1947
```



```
1948                          ;THIS CODE PERFORMS A SEEK FUNCTION ON A DRIVE AND CHECKS IF
1949                          ;THE 'DPL' BIT SET AS A RESULT. (IF THE POWER WAS CUT OFF
1950                          ;FROM THE DRIVE).  NOTE THAT ONLY THOSE DRIVES ARE
1951                          ;CHECKED WHICH WERE FOUND TO BE PRESENT AT BEGINNING (WHEN
1952                          ;THIS TEST WAS ENTERED).  SEEK IS DONE TO CYLINDER 1.
1953                          ;AT THE TIME OF ENTRY R0 POINTS TO THE DRIVE FLAG.
1954
1955  017700  032710  000100  NXT2:   BIT     #BIT6,(R0)      ;WAS THIS DRIVE PRESENT AT BEGNG
1956  017704  001403                  BEQ     4$              ;NO
1957  017706  105777  16 452          TSTB    @RKDS           ;IS IT PRESENT NOW?
1958  017712  100402                  BMI     3$              ;YES
1959  017714  000137  020352  4$:     JMP     DN1DRV          ;IF NOT SKIP THIS CHECK
1960
1961  017720  052777  000040  161450 3$:     BIS     #40,@RKDA       ;RKDA ALREADY HAS THE DRV #
1962                                                          ;SET CYL 1 ADDRESS
1963  017726  012777  000011  161434          MOV     #11,@RKCS       ;SEEK, GO
1964
1965  017734  105777  161430          1$:     TSTB    @RKCS           ;WAIT FOR CONTROL RDY?
1966  017740  100375                  BPL     1$              ;SOMETHING WRONG IF CNTRL RDY
1967                                                          ;DOES NOT COME BACK
1968  017742  032777  010000  161414          BIT     #DPL,@RKDS      ;DPL BIT SET?
1969  017750  001414                  BEQ     2$              ;NO
1970                                                          ;YES, DPL SET
1971  017752  032710  000001          BIT     #BIT0,(R0)      ;WAS 'DPL' SET LAST TIME ALSO?
1972  017756  001167                  BNE     CLRDPL          ;YES, NOTHING TO REPORT.
1973                                                          ;DPL CHANGED, GOT SET THIS
1974  017760  104401  020450          TYPE    ,EM1            ;TIME, REPORT IT
1975  017764  010146                  MOV     R1,-(SP)
1976  017766  104402                  TYPOC
1977  017770  104401  020537          TYPE    ,EM6            ;TYPE 'POWER LO'
1978  017774  052710  000001          BIS     #BIT0,(R0)      ;SET FLAG BIT INDICATING THAT
1979                                                          :DPL SET THIS TIME
1980  020000  000556                  BR      CLRDPL
1981                                                          ;'DPL' BIT IS CLEAR
1982  020002  032710  000001  2$:     BIT     #BIT0,(R0)      ;WAS 'DPL' CLEAR LAST TIME ALSO?
1983  020006  001410                  BEQ     WATSK           ;YES, NOTHING TO REPORT
1984                                                          ;REPORT THAT 'DPL' BIT CHANGED,
1985  020010  104401  020450          TYPE    ,EM1            ;FROM SET TO CLEAR
1986  020014  010146                  MOV     R1,-(SP)
1987  020016  104402                  TYPOC
1988  020020  104401  020560          TYPE    ,EM7            ;TYPE 'POWER UP'
1989  020024  042710  000001          BIC     #BIT0,(R0)      ;SET FLAG BIT INDICATING THAT DPL
1990                                                          ;IS CLEAR THIS TIME
1991
1992                          ;THIS CODE WAITS FOR THE SEEK (DONE ABOVE) TO FINISH.  WAITING
1993                          ;TIME IS APPROX. 50 MS (FOR THE WORST CASE).  IF R/W/S RDY
1994                          ;DOES NOT SET WITHIN 50 MS, THEN IT IS ASSUMED THAT A 'SIN'
1995                          ;IS POSSIBLE AND THE PROGRAM WAITS FOR 1450 MS MORE, SO THAT
1996                          ;THE 'SIN' CAN SET.  IF 'SIN' DOES NOT SET WITHIN THIS
1997                          ;TIME AN ERROR IS REPORTED:
1998                          ;       SIN DIDN'T OCCUR
1999                          ;IF R/W/S RDY SETS WITHIN 50 MS THE PROGRAM PROCEEDS TO
2000                          ;CHECK THE NEXT DRIVE.
2001
2002  020030  012705  164220  WATSK:  MOV     #-6000.,R5      ;SET COUNT TO WAIT FOR
2003                                                          ;50 MS
```

```
2004  020034  032777  000100  161322  1$:     BIT     #RWS,@RKDS      ;R/W/S. RDY SET?
2005  020042  001042                          BNE     3$              ;YES
2006  020044  005205                          INC     R5              ;WAIT
2007  020046  001372                          BNE     1$
2008                                                                  ;50 MS OVER. R/W/S RDY
2009                                                                  ;DIDN'T SET. WAIT FOR
2010                                                                  ;SIN TO SET.
2011  020050  005004                          CLR     R4
2012  020052  012705  177777                  MOV     #177777,R5      ;SET UP COUNT
2013  020056  032777  001000  161300  2$:     BIT     #SIN,@RKDS      ;SIN SET?
2014  020064  001045                          BNE     SINST           ;YES
2015  020066  005305                          DEC     R5              ;WAIT
2016  020070  001372                          BNE     2$
2017  020072  005704                          TST     R4
2018  020074  001002                          BNE     4$
2019  020076  005204                          INC     R4
2020  020100  000766                          BR      2$
2021                                                                  ;1500 MS ELAPSED, BUT SIN
2022                                                                  ;DIDN'T SET. ERROR!
2023  020102                                4$:
2024  020102  104401  020110                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2025  020106  000415                          BR      64$             ;;GET OVER THE ASCIZ
2026                                        ;;65$:  .ASCIZ  <15><12>/SIN DIDN'T OCCUR, DRVE/
2027  020142                                64$:
2028  020142  010146                          MOV     R1,-(SP)                ;TYPE DRIVE #
2029  020144  104402                          TYPOC
2030  020146  000501                          BR      DN1DRV
2031  020150  032710  000002          3$:     BIT     #BIT1,(R0)      ;DID R/W/S RDY SET LAST TIME?
2032  020154  001476                          BEQ     DN1DRV          ;YES
2033  020156  042710  000002                  BIC     #BIT1,(R0)      ;CLR FLAG INDICATING THAT SEEK IS OK
2034  020162  104401  020450                  TYPE    ,EM1            ;REPORT THAT SEEK IS OK
2035  020166  010146                          MOV     R1,-(SP)
2036  020170  104402                          TYPOC
2037  020172  104401  020601                  TYPE    ,EM9
2038  020176  000465                          BR      DN1DRV
2039
2040                                        ;IF SIN SET, DO DRIVE RESET AND CLEAR IT
2041
2042  020200  032710  000002          SINST:  BIT     #BIT1,(R0)      ;DID 'SIN' SET LAST TIME ALSO?
2043  020204  001010                          BNE     4$              ;YES, NOTHING TO REPORT
2044  020206  052710  000002                  BIS     #BIT1,(R0)      ;SET FLAG INDICATING THAT
2045  020212  104401  020450                  TYPE    ,EM1            ;'SIN' SET, AND REPORT THE CHANGE
2046  020216  010146                          MOV     R1,-(SP)
2047  020220  104402                          TYPOC
2048  020222  104401  020573                  TYPE    ,EM8            ;TYPE 'SIN'
2049
2050  020226  017705  161144          4$:     MOV     @RKDA,R5        ;SAVE RKDA
2051  020232  012777  000001  161130          MOV     #1,@RKCS        ;DO CONTROL RESET
2052  020240  105777  161124          1$:     TSTB    @RKCS           ;WAIT FOR CONTROL RDY
2053  020244  100375                          BPL     1$
2054  020246  010577  161124                  MOV     R5,@RKDA
2055  020252  012777  000015  161110          MOV     #15,@RKCS       ;DO DRIVE RESET. RKDA
2056                                                                  ;ALREADY HAS THE DRVE #
2057  020260  105777  161104          2$:     TSTB    @RKCS           ;WAIT FOR CNTRL RDY
2058  020264  100375                          BPL     2$
2059  020266  005005                          CLR     R5
```



```
2060  020270  032777  000100  161066  3$:     BIT     #RWS,@RKDS      ;R/W/S SET?
2061  020276  001025                          BNE     DN1DRV          ;YES
2062  020300  005205                          INC     R5
2063  020302  001372                          BNE     3$              ;WAIT FOR R/W/S RDY
2064                                                                  ;REPORT ERROR. R/W/S RDY CLR
2065  020304  104401  020312                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2066  020310  000411                          BR      64$             ;;GET OVER THE ASCIZ
2067                                        ;;65$:  .ASCIZ  <15><12>/RWS RDY NOT SET/
2068  020334                                64$:
2069
2070  020334  000406                          BR      DN1DRV
2071
2072                                        ;IF DPL SET CLEAR THE ERROR BY DOING CONTROL RESET.
2073
2074  020336  012777  000001  161024  CLRDPL: MOV     #1,@RKCS        ;CONTROL RESET
2075  020344  105777  161020          1$:     TSTB    @RKCS           ;WAIT FOR CNTRL RDY
2076  020350  100375                          BPL     1$
2077                                        ;AT THIS STAGE THE DRIVE (# IN RKDA) HAS BEEN CHECKED
2078                                        ;FOR DRY, WPS, DPL, & SIN. THE POINTERS ARE INCREMENTED
2079                                        ;AND THE SAME CHECKS WILL BE DONE ON THE NEXT
2080                                        ;DRIVE & THEN THE NEXT ONE & SO ON. NOTE THAT
2081                                        ;THIS SUB-PROGRAM KEEPS ON CYCLING THROUGH
2082                                        ;ALL THE DRIVES. AT THE TIME OF ENTRY (HERE)
2083                                        ;R0 POINTS TO THE FLAG FOR THE DRIVE THAT WAS
2084                                        ;JUST CHECKED. BEFORE GOING ON TO THE NEXT
2085                                        ;DRIVE THE HEADS ARE BROUGHT BACK TO CYLINDER
2086                                        ;0 (FOR THE NEXT CYCLE).
2087
2088  020352  011077  161020          DN1DRV: MOV     (R0),@RKDA      ;GET DRIVE #
2089  020356  105777  161002                  TSTB    @RKDS           ;DRIVE PRESENT?
2090  020362  100017                          BPL     3$              ;NO
2091  020364  042777  017777  161004          BIC     #17777,@RKDA    ;CYL ADRES = 0
2092  020372  012777  000011  160770          MOV     #11,@RKCS       ;GO, SEEK
2093
2094  020400  105777  160764          1$:     TSTB    @RKCS           ;WAIT FOR CNTRL RDY
2095  020404  100375                          BPL     1$
2096
2097  020406  004737  005614                  JSR     PC,SMTME
2098  020412  032777  000100  160744  4$:     BIT     #RWS,@RKDS
2099  020420  001774                          BEQ     4$
2100
2101  020422  005720                  3$:     TST     (R0)+           ;INCREMENT POINTERS TO
2102  020424  005201                          INC     R1              ;NEXT DRIVE
2103  020426  020127  000010                  CMP     R1,#10          ;ALL DONE, THIS CYCLE?
2104  020432  001402                          BEQ     2$              ;YES
2105  020434  000137  017430                  JMP     BEGCT1          ;GO DO NEXT DRIVE
2106  020440  000137  017422          2$:     JMP     BEGCT           ;RESTART THE CYCLE OVER
2107                                                                  ;AGAIN
2108
2109
2110
2111  020444  000000                  SMFCNT: .WORD   0
2112  020446  000000                  DRVCNT: .WORD   0
2113                                        ;MESSAGES
2114  020450  005015  051104  053111  EM1:    .ASCIZ  <15><12>/DRIVE /
2115  020456  020105     000
```

```
2116  020461  040     047440  020116  EM2:    .ASCIZ  /  ON LINE/
2117  020466  044514  042516  000
2118  020473  040     047440  043106  EM3:    .ASCIZ  /  OFF LINE/
2119  020500  046040  047111  000105
2120  020506  020040  051127  020124  EM4:    .ASCIZ  /  WRT PROT/
2121  020514  051120  052117  000
2122  020521  040     053440  052122  EM5:    .ASCIZ  /  WRT ENABLED/
2123  020526  042440  040516  046102
2124  020534  042105  000
2125  020537  040     042040  044522  EM6:    .ASCIZ  /  DRIVE POWER LO/
2126  020544  042526  050040  053517
2127  020552  051105  046040  000117
2128  020560  020040  047520  042527  EM7:    .ASCIZ  /  POWER UP/
2129  020566  020122  050125  000
2130  020573  040     051440  047111  EM8:    .ASCIZ  /  SIN/
2131  020600  000
2132  020601  040     051440  042505  EM9:    .ASCIZ  /  SEEK OK/
2133  020606  020113  045517  000
2134
2135                                  ;DRIVE FLAGS FOR CONTROL PANEL TEST #2
2136                                  ;BITS 15,14,13 GIVE THE DRIVE NO (EX: 0,1,2,---)
2137                                  ;BIT 7 IS SET WHEN 'DRY' BIT IS SET FOR THE DRIVE (ON LINE)
2138                                  ;BIT 7 IS CLEAR WHEN 'DRY' IS CLEAR (WHEN DRIVE IS IN LOAD/OFF LINE.
2139                                  ;DRIVE POWER IS CUT OFF)
2140                                  ;BIT 6 IS SET IF A DRIVE IS FOUND TO BE PRESENT (DRY) AT
2141                                  ;THE BEGINING. UNLIKE BIT 7 THIS BIT DOES NOT GET SET OR
2142                                  ;CLEARED AS THE DRIVE CONDITIONS CHANGE. IT JUST INDICATES
2143                                  ;THAT THE DRIVE IS AVAILABLE FOR CHECKING.
2144                                  ;BIT 0 IS SET IF 'DPL' BIT GETS SET, IE: DRIVE POWER OFF
2145                                  ;BIT 0 IS CLEARED WHEN DRIVE POWER IS ON.
2146                                  ;BIT 1 IS SET WHEN SEEK INCOMPLETE 'SIN' OCCURS.
2147                                  ;BIT 1 IS CLEAR WHEN SEEK IS OK.
2148                                  ;BIT 2 IS SET WHEN WRT PROT IS SET FROM CONSOLE.
2149                                  ;BIT 2 IS CLEARED WHEN DRIVE IS WRITE ENABLED.
2150
2151          020614                          .EVEN
2152  020614  000000                  DRIV0:  .WORD   0               ;DRIVE FLAGS
2153  020616  000000                  DRIV1:  .WORD   0
2154  020620  000000                  DRIV2:  .WORD   0
2155  020622  000000                  DRIV3:  .WORD   0
2156  020624  000000                  DRIV4:  .WORD   0
2157  020626  000000                  DRIV5:  .WORD   0
2158  020630  000000                  DRIV6:  .WORD   0
2159  020632  000000                  DRIV7:  .WORD   0
2160
2161
```



```
2162                                          .SBTTL  HEAD ALIGNMENT ROUTINE
2163
2164                                  ;HEAD ALIGNMENT ROUTINE
2165                                  ;THIS MAINTAINANCE ROUTINE IS HELPFUL IN HEAD ALIGNMENT. UPON ENTRY
2166                                  ;THE QUESTION - DRIVE? - IS ASKED , THE USER SHOULD REPLY WITH THE
2167                                  ;DRIVE NUMBER THAT IS TO BE ALIGNED. IF THE DRIVE IS AN RK-05F
2168                                  ;THE LETTER 'F' IS ADDED AS A SUFFIX. FOR SELECTING SURFACE 0
2169                                  ;PUT SW0=0, FOR SELECTING SURFACE 1 PUT SW0=1. SET SW1 =1 TO SELECT
2170                                  ;CYLINDER 64.  SET SW1=0 TO SELECT CYLINDER 105.
2171                                  ;IF THE DRIVE IS AN RK-05F, CYLINDER 64 BECOMES CYLINDER 130
2172                                  ;OF THE EVEN DRIVE, AND CYLINDER 105 BECOMES CYLINDER 5 OF THE ODD DRIVE
2173                                  ;THE HEADS ARE PLACED ON THE SELECTED CYLINDER AND DATA IS READ
2174                                  ;CONTINUOSLY FROM THE CYLINDER (SECTOR 0)
2175                                  ; THE UPPER OR LOWER HEAD AND CYLINDER CAN BE SELECTED
2176                                  ;DYNAMICALLY, IE. THE PROGRAM DOES NOT HAVE TO BE STOPPED TO SELECT THE
2177                                  ;UPPER OR LOWER HEAD OR CYLINDER.
2178                                  ;IN ORDER TO SELECT ANOTHER DRIVE, PUT ANY SWITCH FROM SW2 TO SW15 UP AND
2179                                  ;THE PROGRAM WILL AGAIN ASK THE QUESTION (DRIVE?).
2180
2181  020634  000240                  SECT.5: NOP
2182  020636  104401  020644                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2183  020642  000426                          BR      64$             ;;GET OVER THE ASCIZ
2184                                  ;;65$:  .ASCIZ  <15><12>/SET SW0=0 FOR SURFACE 0, SW0=1 FOR SUR 1./
2185  020720                          64$:
2186  020720  104401  020726                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2187  020724  000432                          BR      66$             ;;GET OVER THE ASCIZ
2188                                  ;;67$:  .ASCIZ  <15><12>/SET SW1=0 FOR CYLINDER 105, SW1=1 FOR CYLINDER 64/
2189  021012                          66$:
2190  021012  104401  021020                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2191  021016  000427                          BR      68$             ;;GET OVER THE ASCIZ
2192                                  ;;69$:  .ASCIZ  <15><12>/PUT ANY SW FROM 2-15 HI TO SELECT NEW DRVE/
2193  021076                          68$:
2194  021076  104401  021532          HDALGN: TYPE    ,EM10                   ;ASK FOR DRIVE #
2195  021102  005037  021526                  CLR     FFLAG                   ;FLAG FOR RK-05F
2196  021106  104410                          RDLIN                           ;GET OPR INPUT
2197  021110  012601                          MOV     (SP)+,R1                ;ADDR OF COMMAND STRING
2198  021112  112100                          MOVB    (R1)+,R0                ;FIRST CHAR
2199  021114  162700  000060                  SUB     #60,R0                  ;0 TO 7
2200  021120  002766                          BLT     HDALGN                  ;TOO SMALL
2201  021122  022700  000067                  CMP     #67,R0                  ;MUST BE 7 OR LESS
2202  021126  002763                          BLT     HDALGN                  ;TOO BIG
2203  021130  000241                          CLC
2204  021132  006000                          ROR     R0
2205  021134  006000                          ROR     R0
2206  021136  006000                          ROR     R0
2207  021140  006000                          ROR     R0
2208  021142  010037  020614                  MOV     R0,DRIV0                ;ADDRESS OF DRIVE
2209  021146  112100                          MOVB    (R1)+,R0                ;NEXT INPUT CHAR
2210  021150  001412                          BEQ     5$                      ;ALL DONE IF C.R.
2211  021152  020027  000106                  CMP     R0,#'F                  ;IS IT F?
2212  021156  001347                          BNE     HDALGN                  ;NO, SO ERROR
2213  021160  105711                          TSTB    (R1)                    ;NEXT CHAR MUST BE C.R.
2214  021162  001345                          BNE     HDALGN                  ;ELSE, ERROR
2215  021164  042737  020000  020614          BIC     #BIT13,DRIV0            ;USE EVEN DRIVE IF RK-05F
2216  021172  005237  021526                  INC     FFLAG                   ;SHOW F TYPE DRIVE
2217  021176  013700  020614          5$:     MOV     DRIV0,R0                ;DRIVE ADDR TO R0
```

```
2218  021202  022737  000176  001140          CMP     #SWREG,SWR          ;SOFTWARE SWITCH REGISTER IN USE?
2219  021210  001002                          BNE     15$                 ;BR IF NOT
2220  021212  104405                          GTSWR                       ;REQUEST NEW CONTENTS FOR SWITCH REG
2221  021214  000401                          BR      16$                 ;CONTINUE
2222  021216  000000                  15$:    HALT                        ;WAIT FOR OPERATOR TO ENTER NEW SWR VALUE
2223  021220  017737  157714  021530  16$:    MOV     @SWR,SWTCH          ;HOLD SWITCHES
2224  021226  042737  177775  021530          BIC     #177775,SWTCH       ;WANT SW1 ONLY
2225  021234  001005                          BNE     7$                  ;SW1 SET, SO LOW CYLINDER
2226  021236  005737  021526                  TST     FFLAG               ;F DRIVE?
2227  021242  001402                          BEQ     7$                  ;NO
2228  021244  052700  020000                  BIS     #BIT13,R0           ;ODD DRIVE IF HIGH TRACK OF F
2229  021250  010077  160122          7$:     MOV     R0,@RKDA            ;ADDRESS DRIVE
2230  021254  012777  000017  160106          MOV     #17,@RKCS           ;WRITE PROTECT
2231  021262  105777  160102          8$:     TSTB    @RKCS
2232  021266  100375                          BPL     8$                  ;WAIT FOR DRIVE READY
2233  021270  012777  000001  160072          MOV     #1,@RKCS            ;RESET CONTROLLER
2234  021276  105777  160066          9$:     TSTB    @RKCS
2235  021302  100375                          BPL     9$                  ;WAIT FOR READY
2236  021304  005737  021526                  TST     FFLAG               ;F DRIVE?
2237  021310  001410                          BEQ     13$                 ;NO
2238  021312  012701  000240                  MOV     #5.*40,R1           ;TRACK 5 OF HIGH
2239  021316  005737  021530                  TST     SWTCH               ;SW1 SET?
2240  021322  001412                          BEQ     10$                 ;YES SO TEST TRACK B OF DRIVE HIGH
2241  021324  062701  010100                  ADD     #130.*40,R1         ;TRACK 130. IF SW1 SET
2242  021330  000407                          BR      10$
2243  021332  012701  004000          13$:    MOV     #64.*40,R1          ;CYLINDER 64 IF NOT F
2244  021336  005737  021530                  TST     SWTCH               ;SW1 SET?
2245  021342  001002                          BNE     10$                 ;YES, SO CYLINDER 64
2246  021344  062701  002440                  ADD     #41.*40,R1          ;CYLINDER 105
2247  021350  005777  160014          10$:    TST     @RKCS               ;ANY ERROR?
2248  021354  100006                          BPL     11$                 ;NO, CONTINUE
2249  021356  012777  000001  160004          MOV     #1,@RKCS            ;RESET
2250  021364  105777  160000          12$:    TSTB    @RKCS
2251  021370  100375                          BPL     12$                 ;WAIT FOR READY
2252  021372  017702  157542          11$:    MOV     @SWR,R2             ;SWITCH REG TO R2
2253  021376  042702  177775                  BIC     #177775,R2          ;SW1 ONLY
2254  021402  020237  021530                  CMP     R2,SWTCH            ;ANY CHANGE SINCE LAST?
2255  021406  001273                          BNE     5$                  ;YES, GO SET-UP ADDR AGAIN
2256  021410  010077  157762          6$:     MOV     R0,@RKDA            ;ADDRESS THE DRIVE
2257  021414  012777  000017  157746          MOV     #17,@RKCS           ;WRITE  PROTECT THE DRIVE
2258  021422  105777  157742                  TSTB    @RKCS               ;WAIT FOR CONTROL RDY
2259  021426  100375                          BPL     .-4
2260  021430  042700  000020          4$:     BIC     #20,R0              .CLEAR TRACK ADDR
2261  021434  032777  000001  157476          BIT     #1,@SWR             ;SW0 SET?
2262  021442  001402                          BEQ     2$                  ;NO TEST TRACK 0
2263  021444  052700  000020                  BIS     #20,R0              ;TEST TRACK 1
2264  021450  042700  017740          2$:     BIC     #17740,R0           ;CLEAR CYLINDER ADDR
2265  021454  050100                          BIS     R1,R0               ;PUT CYLINDER ADDR IN ADDR
2266  021456  010077  157714                  MOV     R0,@RKDA            ;ADRES THE DRIVE
2267  021462  012777  177400  157702          MOV     #-400,@RKWC         ;READ 1 SECTOR
2268  021470  012777  007356  157676          MOV     #RDBUFF,@RKBA       ;INTO THIS BUFFER
2269  021476  012777  000005  157664          MOV     #5,@RKCS            ;READ, GO
2270
2271  021504  105777  157660          3$:     TSTB    @RKCS               ;DONE?
2272  021510  100375                          BPL     3$                  ;NO
2273
```



```
2274  021512  032777  177774  157420          BIT     #177774,@SWR        ;EXIT OUT?
2275  021520  001713                          BEQ     10$                 ;NO, CONTINUE ON THIS DRIVE
2276  021522  000137  021076                  JMP     HDALGN              ;YES, GET NEW DRIVE
2277
2278  021526  000000                  FFLAG:  0
2279  021530  000000                  SWTCH:  0
2280  021532  005015  051104  053111  EM10:   .ASCIZ  <15><12>/DRIVE?/
2281  021540  037505     000
2282
2283
2284
2285                                          .SBTTL  DISK POWER FAILURE TEST
2286
2287                                  ;(DISK)POWER FAILURE (DURING DISK WRITE) TEST
2288                                  ;THIS TEST CHECKS THAT THE INFORMATION WRITTEN ON THE DISK IS
2289                                  ;NOT DESTROYED WHEN THE DISK SENSES A LOSS OF POWER WHILE DOING A WRITE
2290                                  ;AND RETRACTS THE HEADS. UPON ENTRY THE PROGRAM FINDS OUT THE
2291                                  ;FIRST AVAILABLE DRIVE INDICATES IT (DRIVE XX) AND PROCEEDS TO TEST.
2292                                  ;CYLINDERS 0 TO 15 ARE WRITTEN WITH UNIQUE PATTERNS, THEN THE HEADS
2293                                  ;ARE POSITIONED ON CYLINDER 10 (DECIMAL) AND A MESSAGE (DROP
2294                                  ;POWER) IS GIVEN.  AFTER RECEIVING THIS MESSAGE THE USER SHOULD
2295                                  ;DROP POWER FROM THE DRIVE.  ON SENSING A LOSS OF POWER, THE
2296                                  ;PROGRAM ASKS THE USER TO PUT BACK THE POWER.  THE ERRORS (DPL)
2297                                  ;ARE CLEARED AND A WRITE-CHECK IS PERFORMED TO CHECK IF THE
2298                                  ;UNIQUE PATTERNS ON THE DISK (CYLINDERS 0-9 AND 11-15) ARE STILL
2299                                  ;THERE.  IF NOT A WRITE CHECK ERROR IS REPORTED.
2300          021544                          .EVEN
2301
2302  021544  005000          SECT.6: CLR     R0                  ;INITIALIZE DRIVE #
2303  021546  005037  020614          CLR     DRIV0
2304  021552  010077  157620  8$:     MOV     R0,@RKDA            ;IS IT PRESENT?
2305  021556  105777  157602          TSTB    @RKDS               ;IF NOT SKIP
2306  021562  100406                  BMI     12$
2307  021564  005237  020614  10$:     INC     DRIV0
2308  021570  062700  020000          ADD     #20000,R0
2309  021574  001366                  BNE     8$
2310  021576  000762                  BR      SECT.6
2311  021600  013746  020614  12$:     MOV     DRIV0,-(SP)         ;GET DRIVE #
2312  021604  032777  000040  157552          BIT     #40,@RKDS
2313  021612  001406                  BEQ     9$
2314  021614  104401  022234                  TYPE    ,EM11
2315  021620  104402                  TYPOC
2316  021622  104401  022244                  TYPE    ,EM12
2317  021626  000756                  BR      10$
2318  021630  104401  020450  9$:     TYPE    ,EM1
2319  021634  013746  020614          MOV     DRIV0,-(SP)
2320  021640  104402                  TYPOC
2321  021642  012777  000001  157520          MOV     #1,@RKCS            ;CONTROL RESET
2322  021650  105777  157514          TSTB    @RKCS
2323  021654  100375                  BPL     .-4
2324  021656  005005                  CLR     R5                  ;INITIALIZE PATTERN TO BE WRITTEN
2325                                                              ;0 ON CYL 0, 1 ON CYL 1, ETC
2326  021660  013701  001376                  MOV     RKDA,R1
2327  021664  013702  001372                  MOV     RKWC,R2
2328  021670  013703  001374                  MOV     RKBA,R3
2329  021674  013704  001370                  MOV     RKCS,R4
```

```
2330  021700  010011                          MOV     R0,@R1
2331  021702  010537  022232          1$:     MOV     R5,BUFR         ;FILL THE PATTER IN DATA BUFFER.
2332  021706  012713  022232                  MOV     #BUFR,@R3       ;BUS ADRES
2333  021712  012712  164000                  MOV     #-14000,@R2     ;WRITE 1 CYL (256X12X2 WORDS)
2334  021716  012714  004003                  MOV     #4003,@R4       ;WRITE, GO, IBA SET
2335  021722  105714                          TSTB    @R4             ;WAIT FOR CONTROL READY
2336  021724  100376                          BPL     .-2
2337                                                                  ;DONE
2338  021726  005205                          INC     R5              ;WRITTEN ALL 15 CYLINDERS?
2339  021730  020527  000020                  CMP     R5,#20          ;IF NOT, GO BAK
2340  021734  001362                          BNE     1$
2341
2342  021736  010005                          MOV     R0,R5           ;DRIVE #
2343  021740  052705  000500                  BIS     #500,R5         ;CYL 10
2344  021744  012737  000012  022232          MOV     #12,BUFR        ;PATERN TO BE WRITTEN
2345  021752  010511                          MOV     R5,@R1          ;ADRES THE DISK
2346  021754  012712  164000                  MOV     #-14000,@R2     ;WORD COUNT= 1 CYLINDER
2347  021760  012713  022232                  MOV     #BUFR,@R3       ;BUS ADRES
2348  021754  012714  004003                  MOV     #4003,@R4       ;WRITE, GO, IBA
2349  021770  032777  000100  157366  2$:     BIT     #RWS,@RKDS      ;WAIT FOR THE HEADS TO SETTLE
2350  021776  001774                          BEQ     2$              ;ON CYL 10
2351  022000  104401  022006                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2352  022004  000406                          BR      64$             ;;GET OVER THE ASCIZ
2353                                  ;;65$:  .ASCIZ  /DROP POWER/
2354  022022                          64$:
2355
2356  022022  000407                          BR      5$
2357  022024  010511                  3$:     MOV     R5,@R1          ;ADRES THE DISK
2358  022026  012712  164000                  MOV     #-14000,@R2     ;WORD COUNT= 1 CYLINDER
2359  022032  012713  022232                  MOV     #BUFR,@R3       ;BUS ADRES
2360  022036  012714  004003                  MOV     #4003,@R4       ;WRITE, GO, IBA
2361  022042  105714                  5$:     TSTB    @R4             ;WAIT FOR CONTROL READY
2362  022044  100376                          BPL     .-2
2363  022046  005714                          TST     @R4             ;WAIT FOR DRIVE POWER TO GO DOWN,
2364                                                                  ;OTHERWISE, KEEP ON WRITING ON CYL 10
2365  022050  100365                          BPL     3$
2366                                                                  ;IF DRIVE POWER LOSS WAS SENSED,
2367                                                                  ;ASK TO PUT POWER ON.
2368  022052  104401  022060                  TYPE    ,67$            ;;TYPE ASCIZ STRING
2369  022056  000406                          BR      66$             ;;GET OVER THE ASCIZ
2370                                  ;;67$:  .ASCIZ  <15><12>/POWER ON/
2371  022074                          66$:
2372  022074  105777  157264                  TSTB    @RKDS           ;WAIT FOR DRIVE READY
2373  022100  100375                          BPL     .-4
2374  022102  012714  000001                  MOV     #1,@R4          ;CONTRO. RESET, CLEAR ERROR
2375  022106  105714                          TSTB    @R4
2376  022110  100376                          BPL     .-2
2377  022112  010077  157260                  MOV     R0,@RKDA
2378  022116  005005                          CLR     R5              ;INITIALIZE PATTERN
2379  022120  010537  022232          6$:     MOV     R5,BUFR
2380  022124  012713  022232                  MOV     #BUFR,@R3
2381  022130  012712  164000                  MOV     #-14000,@R2
2382  022134  012714  004007                  MOV     #4007,@R4       ;WRITE CHECK, GO, IBA
2383  022140  105714                          TSTB    @R4
2384  022142  100376                          BPL     .-2
2385
```



```
2386  022144  005714                          TST     @R4             ;ANY ERROR?
2387  022146  100023                          BPL     7$              ;NO
2388  022150  104401  022156                  TYPE    ,69$            ;;TYPE ASCIZ STRING
2389  022154  000416                          BR      68$             ;;GET OVER THE ASCIZ
2390                                  ;;69$:  .ASCIZ  <15><12>/ERROR, ON PWR-UP.  RKDA=/
2391  022212                          68$:
2392  022212  011146                          MOV     @R1,-(SP)
2393  022214  104402                          TYPOC
2394  022216  005205                  7$:     INC     R5
2395  022220  020527  000020                  CMP     R5,#20
2396  022224  001335                          BNE     6$
2397
2398  022226  000137  021564                  JMP     10$
2399
2400  022232  000000                  BUFR:   .WORD   0
2401  022234  051104  053111  020105  EM11:   .ASCIZ  /DRIVE  /
2402  022242  000040
2403  022244  051511  053440  044522  EM12:   .ASCIZ  /IS WRITE PROTECTED /
2404  022252  042524  050040  047522
2405  022260  042524  052103  042105
2406  022266  000040
2407
2408
2409                                  .SBTTL  TYPE ROUTINE
2410
2411                                  ;;*****************************************************************
2412                                  ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
2413                                  ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
2414                                  ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
2415                                  ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
2416                                  ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
2417                                  ;*
2418                                  ;*CALL:
2419                                  ;*1) USING A TRAP INSTRUCTION
2420                                  ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
2421                                  ;*OR
2422                                  ;*      TYPE
2423                                  ;*      MESADR
2424                                  ;*
2425
2426  022270  105737  001157          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
2427  022274  100002                          BPL     1$              ;;BR IF YES
2428  022276  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
2429  022300  000407                          BR      3$              ;;LEAVE
2430  022302  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
2431  022304  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
2432  022310  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
2433  022312  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
2434  022314  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
2435  022316  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
2436  022320  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
2437  022324  000002                          RTI                     ;;RETURN
2438  022326  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
2439  022332  001430                          BEQ     8$
2440  022334  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
2441  022340  001006                          BNE     5$
```

```
2442  022342  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
2443  022344  104401                        TYPE                    ;;TYPE A CR AND LF
2444  022346  001161                        $CRLF
2445  022350  105037  022504                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
2446  022354  000755                        BR      2$              ;;GET NEXT CHARACTER
2447  022356  004737  022440        5$:     JSR     PC,$TYPEC       ;;GO TYPE A FILLER CHARACTER
2448  022362  123726  001156        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
2449  022366  001350                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
2450  022370  013746  001154                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
2451                                                                ;;AND THE NULL CHAR.
2452  022374  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
2453  022400  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
2454  022402  004737  022440                JSR     PC,$TYPEC       ;;GO TYPE A NULL
2455  022406  105337  022504                DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
2456  022412  000770                        BR      7$              ;;LOOP
2457
2458                                        ;HORIZONTAL TAB PROCESSOR
2459
2460  022414  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
2461  022420  004737  022440        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
2462  022424  132737  000007  022504        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
2463  022432  001372                        BNE     9$              ;;TAB STOP
2464  022434  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
2465  022436  000724                        BR      2$              ;;GET NEXT CHARACTER
2466  022440  105777  156504        $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
2467  022444  100375                        BPL     $TYPEC
2468  022446  116677  000002  156476        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2469  022454  122766  000015  000002        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
2470  022462  001003                        BNE     1$              ;;BRANCH IF NO
2471  022464  105037  022504                CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
2472  022470  000406                        BR      $TYPEX          ;;EXIT
2473  022472  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
2474  022500  001402                        BEQ     $TYPEX          ;;BRANCH IF YES
2475  022502  105227                        INCB    (PC)+           ;;COUNT THE CHARACTER
2476  022504  000000        $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
2477  022506  000207        $TYPEX: RTS     PC
2478
2479                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
2480
2481                                        ;;**************************************************************
2482                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2483                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2484                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2485                                        ;*CALL:
2486                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2487                                        ;*      TYPOS                   ;;CALL FOR TYPEOUT
2488                                        ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2489                                        ;*      .BYTE   M               ;;M=1 OR 0
2490                                        ;*                                      ;;1=TYPE LEADING ZEROS
2491                                        ;*                                      ;;0=SUPPRESS LEADING ZEROS
2492                                        ;*
2493                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2494                                        ;*$TYPOS OR $TYPOC
2495                                        ;*CALL:
2496                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2497                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
```



```
2498                                        ;*
2499                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2500                                        ;*CALL:
2501                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2502                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
2503
2504  022510  017646  000000        $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
2505  022514  116637  000001  022733        MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
2506  022522  112637  022735                MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
2507  022526  062716  000002                ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
2508  022532  000406                        BR      $TYPON
2509  022534  112737  000001  022733  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2510  022542  112737  000006  022735        MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2511  022550  112737  000005  022732  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2512  022556  010346                        MOV     R3,-(SP)        ;;SAVE R3
2513  022560  010446                        MOV     R4,-(SP)        ;;SAVE R4
2514  022562  010546                        MOV     R5,-(SP)        ;;SAVE R5
2515  022564  113704  022735                MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2516  022570  005404                        NEG     R4
2517  022572  062704  000006                ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2518  022576  110437  022734                MOVB    R4,$OMODE       ;;SAVE IT FOR USE
2519  022602  113704  022733                MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2520  022606  016605  000012                MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2521  022612  005003                        CLR     R3              ;;CLEAR THE OUTPUT WORD
2522  022614  006105                1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2523  022616  000404                        BR      3$              ;;GO DO MSB
2524  022620  006105                2$:     ROL     R5              ;;FORM THIS DIGIT
2525  022622  006105                        ROL     R5
2526  022624  006105                        ROL     R5
2527  022626  010503                        MOV     R5,R3
2528  022630  006103                3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2529  022632  105337  022734                DECB    $OMODE          ;;TYPE THIS DIGIT?
2530  022636  100016                        BPL     7$              ;;BR IF NO
2531  022640  042703  177770                BIC     #177770,R3      ;;GET RID OF JUNK
2532  022644  001002                        BNE     4$              ;;TEST FOR 0
2533  022646  005704                        TST     R4              ;;SUPPRESS THIS 0?
2534  022650  001403                        BEQ     5$              ;;BR IF YES
2535  022652  005204                4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2536  022654  052703  000060                BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2537  022660  052703  000040        5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2538  022664  110337  022730                MOVB    R3,8$           ;;SAVE FOR TYPING
2539  022670  104401  022730                TYPE    ,8$             ;;GO TYPE THIS DIGIT
2540  022674  105337  022732        7$:     DECB    $OCNT           ;;COUNT BY 1
2541  022700  003347                        BGT     2$              ;;BR IF MORE TO DO
2542  022702  002402                        BLT     6$              ;;BR IF DONE
2543  022704  005204                        INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2544  022706  000744                        BR      2$              ;;GO DO THE LAST DIGIT
2545  022710  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
2546  022712  012604                        MOV     (SP)+,R4        ;;RESTORE R4
2547  022714  012603                        MOV     (SP)+,R3        ;;RESTORE R3
2548  022716  016666  000002  000004        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2549  022724  012616                        MOV     (SP)+,(SP)
2550  022726  000002                        RTI                     ;;RETURN
2551  022730     000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2552  022731     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2553  022732     000        $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
```

```
2554  022733  000                 $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2555  022734  000000              $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2556                              .SBTTL  TTY INPUT ROUTINE
2557
2558                              ;;*****************************************************************
2559                              .ENABL  LSB
2560  022736  000000              $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
2561  022740  000000              $TKQIN: .WORD   0               ;;INPUT POINTER
2562  022742  000000              $TKQOUT: .WORD  0               ;;OUTPUT POINTER
2563  022744  000001              $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
2564          022745              $TKQEND=.
2565          022746              .EVEN
2566
2567                              ;*TK INITIALIZE ROUTINE
2568                              ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
2569                              ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
2570                              ;
2571                              ;*CALL:
2572                              ;*      JSR     PC,$TKINT
2573                              ;*      RETURN
2574                              ;
2575  022746  005037  022736      $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
2576  022752  012737  022744  022740      MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
2577  022760  013737  022740  022742      MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
2578  022766  012737  023016  000060      MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
2579  022774  012737  000200  000062      MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
2580  023002  005777  156140              TST     @$TKB           ;;CLEAR DONE FLAG
2581  023006  012777  000100  156130      MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
2582  023014  000207                      RTS     PC              ;;RETURN TO CALLER
2583
2584                              ;*TK SERVICE ROUTINE
2585                              ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
2586                              ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
2587                              ;*IT IN THE QUEUE.
2588                              ;
2589  023016  117746  156124      $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
2590  023022  042716  177600              BIC     #^C177,(SP)     ;;STRIP THE JUNK
2591  023026  021627  000007      1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
2592  023032  001004                      BNE     2$              ;;BRANCH IF NO
2593  023034  022737  000176  001140      CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
2594  023042  001500                      BEQ     6$              ;;GO TO SWR CHANGE
2595
2596  023044                      2$:
2597  023044  022737  000001  022736      CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
2598  023052  001004                      BNE     3$              ;;BRANCH IF NO
2599  023054  104401  024066              TYPE    ,$BELL          ;;RING THE TTY BELL
2600  023060  005726                      TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
2601  023062  000451                      BR      5$              ;;EXIT
2602  023064  021627  000023      3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
2603  023070  001021                      BNE     32$             ;;BRANCH IF NO
2604  023072  005077  156046              CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
2605  023076  005726                      TST     (SP)+           ;;CLEAN CHAR OFF STACK
2606  023100  105777  156040      31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
2607  023104  100375                      BPL     31$             ;;LOOP UNTIL ITS THERE
2608  023106  117746  156034              MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
2609  023112  042716  177600              BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
```



```
2610  023116  022627  000021              CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
2611  023122  001366                      BNE     31$             ;;BRANCH IF NO
2612  023124  012777  000100  156012      MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
2613  023132  000002                      RTI                     ;;RETURN
2614  023134  005237  022736      32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
2615  023140  021627  000140              CMP     (SP),#140       ;;IS IT UPPER CASE?
2616  023144  002405                      BLT     4$              ;;BRANCH IF YES
2617  023146  021627  000175              CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
2618  023152  003002                      BGT     4$              ;;BRANCH IF YES
2619  023154  042716  000040              BIC     #40,(SP)        ;;MAKE IT UPPER CASE
2620  023160  112677  177554      4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
2621  023164  005237  022740              INC     $TKQIN          ;;UPDATE THE POINTER
2622  023170  023727  022740  022745      CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
2623  023176  001003                      BNE     5$              ;;BRANCH IF NO
2624  023200  012737  022744  022740      MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
2625  023206  000002              5$:     RTI                     ;;RETURN
2626
2627                              ;;*****************************************************************
2628                              ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
2629                              ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER. AND WILL
2630                              ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
2631                              ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
2632  023210  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
2633  023216  001104                      BNE     15$             ;;EXIT IF NOT
2634  023220  105777  155720              TSTB    @$TKS           ;;IS A CHAR WAITING?
2635  023224  100101                      BPL     15$             ;;IF NOT, EXIT
2636  023226  117746  155714              MOVB    @$TKB,-(SP)     ;;YES
2637  023232  042716  177600              BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
2638  023236  021627  000007              CMP     (SP),#7         ;;IS IT A CONTROL-G?
2639  023242  001300                      BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
2640                                                              ;;AND EXIT
2641
2642                              ;;*****************************************************************
2643                              ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
2644                              ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL. AS A RESULT OF A
2645                              ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
2646  023244  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;ARE WE RUNNING IN AUTO-MODE?
2647  023252  001674                      BEQ     2$              ;BRANCH IF YES
2648  023254  005726                      TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
2649  023256  004737  022746              JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
2650  023262  005077  155656              CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
2651  023266  112737  000001  001135      MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
2652
2653  023274  104401  024077              TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
2654  023300  104401  024104      $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
2655  023304  013746  000176              MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
2656  023310  104402                      TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2657  023312  104401  024115              TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
2658  023316  005046              19$:    CLR     -(SP)           ;;CLEAR COUNTER
2659  023320  005046                      CLR     -(SP)           ;;THE NEW SWR
2660  023322  105777  155616      7$:     TSTB    @$TKS           ;;CHAR THERE?
2661  023326  100375                      BPL     7$              ;;IF NOT TRY AGAIN
2662
2663  023330  117746  155612              MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
2664  023334  042716  177600              BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
2665
```

```
023340 021627 000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
023344 001005                         BNE     10$             ;;BRANCH IF NOT
023346 104401 024072                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
023352 062706 000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
023356 000757                         BR      19$             ;;LET'S TRY IT AGAIN

023360 021627 000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
023364 001022                         BNE     16$             ;;BRANCH IF NO
023366 005766 000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
023372 001403                         BEQ     11$             ;;BRANCH IF YES
023374 016677 000002 155536           MOV     2(SP),@SWR      ;;SAVE NEW SWR
023402 062706 000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
023406 104401 001161          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
023412 123727 001135 000001           CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
023420 001003                         BNE     15$             ;;BRANCH IF NOT
023422 012777 000100 155514           MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
023430 000002                 15$:    RTI                     ;;RETURN
023432 004737 022440          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
023436 021627 000060                  CMP     (SP),#60        ;;CHAR < 0?
023442 002420                         BLT     18$             ;;BRANCH IF YES
023444 021627 000067                  CMP     (SP),#67        ;;CHAR > 7?
023450 003015                         BGT     18$             ;;BRANCH IF YES
023452 042726 000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
023456 005766 000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
023462 001403                         BEQ     17$             ;;BRANCH IF YES
023464 006316                         ASL     (SP)            ;;NO, SHIFT PRESENT
023466 006316                         ASL     (SP)            ;;   CHAR OVER TO MAKE
023470 006316                         ASL     (SP)            ;;   ROOM FOR NEW ONE.
023472 005266 000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
023476 056616 177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
023502 000707                         BR      7$              ;;GET THE NEXT ONE
023504 104401 001160          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
023510 000720                         BR      20$             ;;SIMULATE CONTROL-U
                              .DSABL  LSB



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

023512 011646                 $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
023514 016666 000004 000002           MOV     4(SP),2(SP)     ;;THE PS
023522 005066 000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
023526 005046                         CLR     -(SP)           ;;PUT NEW PS ON STACK
023530 012746 023536                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
023534 000002                         RTI                     ;;POP NEW PC AND PS
023536                        64$:
023536 005737 022736          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
023542 001775                         BEQ     1$
```



```
023544 005337 022736                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
023550 117766 177166 000004           MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
023556 005237 022742                  INC     $TKQOUT         ;;UPDATE THE POINTER
023562 023727 022742 022745           CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
023570 001003                         BNE     2$              ;;BRANCH IF NO
023572 012737 022744 022742           MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
023600 000002                 2$:     RTI                     ;;RETURN
                              ;;*************************************************************
                              ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                              ;*CALL:
                              ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                              ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                              ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

023602 010346                 $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
023604 005046                         CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
023606 012703 024036          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
023612 022703 024066          2$:     CMP     #$TTYIN+30,R3   ;;BUFFER FULL?
023616 101456                         BLOS    4$              ;;BR IF YES
023620 104407                         RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
023622 112613                         MOVB    (SP)+,(R3)      ;;GET CHARACTER
023624 122713 000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
023630 001022                         BNE     5$              ;;BR IF NO
023632 005716                         TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
023634 001007                         BNE     6$              ;;BR IF NO
023636 112737 000134 024034           MOVB    #'\,9$          ;;TYPE A BACK SLASH
023644 104401 024034                  TYPE    ,9$
023650 012716 177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
023654 005303                 6$:     DEC     R3              ;;BACKUP BY ONE
023656 020327 024036                  CMP     R3,#$TTYIN      ;STACK EMPTY?
023662 103434                         BLO     4$              ;BR IF YES
023664 111337 024034                  MOVB    (R3),9$         ;SETUP TO TYPEOUT THE DELETED CHAR.
023670 104401 024034                  TYPE    ,9$             ;;GO TYPE
023674 000746                         BR      2$              ;;GO READ ANOTHER CHAR.
023676 005716                 5$:     TST     (SP)            ;;RUBOUT KEY SET?
023700 001406                         BEQ     7$              ;;BR IF NO
023702 112737 000134 024034           MOVB    #'\,9$          ;;TYPE A BACK SLASH
023710 104401 024034                  TYPE    ,9$
023714 005016                         CLR     (SP)            ;;CLEAR THE RUBOUT KEY
023716 122713 000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
023722 001003                         BNE     8$              ;;BR IF NO
023724 104401 024072                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
023730 000726                         BR      1$              ;;GO START OVER
023732 122713 000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
023736 001011                         BNE     3$              ;;BRANCH IF NO
023740 105013                         CLRB    (R3)            ;;CLEAR THE CHARACTER
023742 104401 001161                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
023746 104401 024036                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
023752 000717                         BR      2$              ;;GO PICKUP ANOTHER CHACTER
023754 104401 001160          4$:     TYPE    ,$QUES          ;;TYPE A '?'
023760 000712                         BR      1$              ;;CLEAR THE BUFFER AND LOOP
023762 111337 024034          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
023766 104401 024034                  TYPE    ,9$
023772 122723 000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
023776 001305                         BNE     2$              ;;LOOP IF NOT RETURN
024000 105063 177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
```

```
2778  024004  104401  001162          TYPE    ,$LF            ;;TYPE A LINE FEED
2779  024010  005726                  TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
2780  024012  012603                  MOV     (SP)+,R3        ;;RESTORE R3
2781  024014  011646                  MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2782  024016  016666  000004  000002  MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2783  024024  012766  024036  000004  MOV     #$TTYIN,4(SP)
2784  024032  000002                  RTI                     ;;RETURN
2785  024034     000                9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2786  024035     000                        .BYTE   0               ;;TERMINATOR
2787  024036  000030             $TTYIN: .BLKB   30              ;;RESERVE 30 BYTES FOR TTY INPUT
2788  024066  177607  000377     $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
2789  024072  052536  005015  000 $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
2790  024077     136  006507  000012 $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
2791  024104  005015  053523  020122 $MSWR:  .ASCIZ  <15><12>/SWR = /
2792  024112  020075     000
2793  024115     040  047040  053505 $MNEW:  .ASCIZ  /  NEW = /
2794  024122  036440  000040
2795                          .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
2796
2797                          ;;*****************************************************************
2798                          ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
2799                          ;*CHANGE IT TO BINARY.
2800                          ;*CALL:
2801                          ;*      RDOCT                   ;;READ AN OCTAL NUMBER
2802                          ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
2803                          ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
2804
2805  024126  011646          $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
2806  024130  016666  000004  000002  MOV     4(SP),2(SP)     ;;INPUT NUMBER
2807  024136  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2808  024140  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2809  024142  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2810  024144  104410          1$:     RDLIN                   ;;READ AN ASCIZ LINE
2811  024146  012600                  MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
2812  024150  005001                  CLR     R1              ;;CLEAR DATA WORD
2813  024152  005002                  CLR     R2
2814  024154  112046          2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
2815  024156  001412                  BEQ     3$              ;;IF ZERO GET OUT
2816  024160  006301                  ASL     R1              ;;*2
2817  024162  006102                  ROL     R2
2818  024164  006301                  ASL     R1              ;;*4
2819  024166  006102                  ROL     R2
2820  024170  006301                  ASL     R1              ;;*8
2821  024172  006102                  ROL     R2
2822  024174  042716  177770          BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
2823  024200  062601                  ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
2824  024202  000764                  BR      2$              ;;LOOP
2825  024204  005726          3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
2826  024206  010166  000012          MOV     R1,12(SP)       ;;SAVE THE RESULT
2827  024212  010237  024226          MOV     R2,$HIOCT
2828  024216  012602                  MOV     (SP)+,R2        ;;POP STACK INTO R2
2829  024220  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
2830  024222  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
2831  024224  000002                  RTI                     ;;RETURN
2832  024226  000000          $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
2833                          .SBTTL  TRAP DECODER
```



```
2834
2835                          ;;*****************************************************************
2836                          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2837                          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2838                          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2839                          ;*GO TO THAT ROUTINE.
2840
2841  024230  010046          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
2842  024232  016600  000002          MOV     2(SP),R0        ;;GET TRAP ADDRESS
2843  024236  005740                  TST     -(R0)           ;;BACKUP BY 2
2844  024240  111000                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
2845  024242  006300                  ASL     R0              ;;POSITION FOR INDEXING
2846  024244  016000  024264          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
2847  024250  000200                  RTS     R0              ;;GO TO ROUTINE
2848
2849
2850                          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
2851
2852  024252  011646          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
2853  024254  016666  000004  000002  MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
2854  024262  000002                  RTI                     ;;RESTORE THE PSW
2855
2856                          .SBTTL  TRAP TABLE
2857
2858                          ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2859                          ;*BY THE "TRAP" INSTRUCTION.
2860
2861                          ;       ROUTINE
2862                          ;       -------
2863  024264  024252          $TRPAD: .WORD   $TRAP2
2864  024266  022270                  $TYPE   ;;CALL=TYPE      TRAP+1(104401)  TTY TYPEOUT ROUTINE
2865  024270  022534                  $TYPOC  ;;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2866  024272  022510                  $TYPOS  ;;CALL=TYPOS     TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2867  024274  022550                  $TYPON  ;;CALL=TYPON     TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2868
2869  024276  023300                  $GTSWR  ;;CALL=GTSWR     TRAP+5(104405)  GET SOFT-SWR SETTING
2870
2871  024300  023210                  $CKSWR  ;;CALL=CKSWR     TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
2872  024302  023512                  $RDCHR  ;;CALL=RDCHR     TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
2873  024304  023602                  $RDLIN  ;;CALL=RDLIN     TRAP+10(104410) TTY TYPEIN STRING ROUTINE
2874  024306  024126                  $RDOCT  ;;CALL=RDOCT     TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
2875                          .SBTTL  POWER DOWN AND UP ROUTINES
2876
2877                          ;;*****************************************************************
2878                          ;POWER DOWN ROUTINE
2879  024310  012737  024450  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
2880  024316  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2881  024324  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2882  024326  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2883  024330  010246                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2884  024332  010346                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2885  024334  010446                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
2886  024336  010546                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
2887  024340  017746  154574          MOV     @SWR,-(SP)      ;PUSH @SWR ON STACK
2888  024344  010637  024454          MOV     SP,$SAVR6       ;SAVE SP
2889  024350  012737  024362  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
```

```
                                        HALT
                                        BR      .-2             ;;HANG UP

                                ;;**************************************************************
                                ;POWER UP ROUTINE
024356  000000                  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
024360  000776
024362  012737  024450  000024          MOV     $SAVR6,SP       ;;GET SP
024370  013706  024454                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
024374  005037  024454          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
024400  005237  024454                  BNE     1$              ;;OF  WORD
024404  001375                          MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
024406  012677  154526                  MOV     (SP)+,R5        ;;POP STACK INTO R5
024412  012605                          MOV     (SP)+,R4        ;;POP STACK INTO R4
024414  012604                          MOV     (SP)+,R3        ;;POP STACK INTO R3
024416  012603                          MOV     (SP)+,R2        ;;POP STACK INTO R2
024420  012602                          MOV     (SP)+,R1        ;;POP STACK INTO R1
024422  012601                          MOV     (SP)+,R0        ;;POP STACK INTO R0
024424  012600                          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
024426  012737  024310  000024          MOV     #340,@#PWRVEC+2 ;;PRIO:7
024434  012737  000340  000026          TYPE                    ;REPORT THE POWER FAILURE
024442  104401                  $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
024444  024456                          RTI
024446  000002                  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
024450  000000                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
024452  000776                  $SAVR6: 0                       ;;PUT THE SP HERE
024454  000000                  $POWER: .ASCIZ  <15><12>"POWER"
024456  005015  047520  042527
024464  000122
                                        .EVEN
000001                                  .END
```



```
AUTSL2  002644    443#    660
BA      001406    275#   1366*   1368*   1370*   1372*   1393    1427
BADINP  003274    504     506     513#
BADONE  010360   1145#
BAD.IN  011772   1306#
BAD.ON  013762   1570#   1586    1588
BASE    001266    219#    835
BASINC  005360    840     852#
BEGCT   017422   1881#   2106
BEGCT1  017430   1884#   2105
BEGIN   002622    421     427#
BIT0  = 000001    114#   1078    1398    1415    1432    1971    1978    1982    1989
BIT00 = 000001    104#    114
BIT01 = 000002    103#    113
BIT02 = 000004    102#    112
BIT03 = 000010    101#    111
BIT04 = 000020    100#    110
BIT05 = 000040     99#    109
BIT06 = 000100     98#    108
BIT07 = 000200     97#    107
BIT08 = 000400     96#    106
BIT09 = 001000     95#    105
BIT1  = 000002    113#   1396    1431    2031    2033    2042    2044
BIT10 = 002000     94#   1395    1412
BIT11 = 004000     93#   1395    1429
BIT12 = 010000     92#    289
BIT13 = 020000     91#   1774    1784    2215    2228
BIT14 = 040000     90#    508     541     639     873    1418    1547
BIT15 = 100000     89#    609    1000    1002    1111    1685
BIT2  = 000004    112#   1413    1431    1928    1934    1938    1946
BIT3  = 000010    111#
BIT4  = 000020    110#    800     802     805     937     939     942
BIT5  = 000040    109#    291     754    1760
BIT6  = 000100    108#    290    1356    1361    1471    1881    1955
BIT7  = 000200    107#    590     594    1113    1881    1891    1913
BIT8  = 000400    106#   1429
BIT9  = 001000    105#    292
BPTVEC= 000014    121#
BUFF    013700    275    1561#
BUFR    022232   2331*   2332    2344*   2347    2358    2379*   2380    2400#
BUSADR  001336    253#    796*    867     927*
CHECK1  005450    871#    881     885
CHKCNT  001350    258#   1014*   1055*
CKSWR = 104406   2871#
CLRDPL  020336   1972    1980    2074#
CNTSIN  001322    242#    705*    976*
COM     012320   1351#   1364
COMMON  012314   1350#
COMND   001316    238#    788*    904*    998
CONTIN  011404   1230    1232#   1298
CONTRL  001332    251#    798*    869     930*
CR    = 000015     29#   1381    2469    2479
CRLF  = 000200     30#    366    2440    2479
CYCLE   004174    534     672#
CYCLE2  004220    678#
CYCL2   004210    674     676#
```

```
CYL     013552  1514*  1518*  1522*  1527#
CYLADR  005302   793    835#   922
CYLAD2  005306   836#   856
CYLBAS  001315   237#
CYLCNT  001342   255#   797
CYLDFF  005312   808    837#   945
CYLTBL  001274   226#   821    836    854
DA      001410   276#  1365*  1374*  1375   1391   1408   1425   1440
DDISP = 177570    36#   181    338
DISPLA  001142   181#   338*   346*
DISPRE  000174   136#   346
DN1DRV  020352  1959   2030   2032   2038   2061   2070   2088#
DO2     004250   546    614    687#
DPL   = 010000   289#  1968
DRACTV  001164   194#   533*
DRCNT1  001311   233#   455*   509    654*   934
DRIVE   001314   236#   691*   710    790   1589*  1601
DRIV0   020614  1160   1175   1198   1262   1272   1279   1290   1848   1881   2152#  2208*  2215*  2217
                2303*  2307*  2311   2319
DRIV1   020616  2153#
DRIV2   020620  2154#
DRIV3   020622  2155#
DRIV4   020624  2156#
DRIV5   020626  2157#
DRIV6   020630  2158#
DRIV7   020632  2159#
DRVCNT  020446  1322*  1326*  1327   1330   2112#
DRV0    001206   198#   467    528    539    607    655    685    792    909    920
DSKADR  001334   252#   698*   730    800    802*   805*   849*   850*   866    925*   926*   933*   937
                939*   942*  1039
DSKTMP  001352   259#   697*   925   1341*  1352   1392   1409   1426   1596*  1631   1660   1729   1734
                1741  1767   1790
DSWR  = 177570    35#   180    337
ECNT    001321   241#   704*   963*   978
EMTVEC= 000030   124#
EM1     020450  1332   1857   1892   1909   1931   1942   1974   1985   2034   2045   2114#  2318
EM10    021532  2194   2280#
EM11    022234  2314   2401#
EM12    022244  2316   2403#
EM2     020461  1860   1895   2116#
EM3     020473  1912   2118#
EM4     020506   945   2120#
EM5     020521  1935   2122#
EM6     020537  1977   2125#
EM7     020560  1988   2128#
EM8     020573  2048   2130#
EM9     020601  2037   2132#
ERRCHK  006102   873    955#
ERRFLG  001360   262#   874    876*   960*   970*
ERRRF   001424   282#  1343*  1420*  1477*  1478
ERRRFC  001426   283#  1344*  1444*  1484*  1487
ERRVEC= 000004   117#   335    336*   347*
ERRWCH  001430   284#  1345*  1452*  1553*  1554
ERRWCS  001432   285#  1346*  1453*  1549*  1550
ERRWF   001422   281#  1342*  1403*  1459*  1460
EXECUT  005402   799    863#   877    891    931
```



```
EXIT    003366   529    538#
EXITA   003654   601#   603    641
EXITB   003662   604#   637
EXITX   003430   540    542    550#
EXIT2   006076   910    949#
EXTFR2  003372   539#   544    549   1135
EXTR    001420   280#  1349*  1397   1414   1430
FAIL    013324  1493#  1551
FAILED  013306  1488   1490#
FFLAG   021526  2195*  2216*  2226   2238   2278#
FIL.DN  003672   607#   610
GD1     012534  1328   1383#
GNS   = ****** U  135    365    377    381    385    389    393    397    401    405    409    414    446
                 450    479    491    498    516    553    563    571    580    619    663    708    716
                 720    744    751    758    778    889    983    989    994   1023   1032   1037   1045
                1051   1088   1092   1097   1105   1121   1148   1157   1185   1189   1210   1214   1218
                1222   1242   1309   1314   1380   1465   1496   1573   1578   1599   1607   1611   1615
                1621   1627   1640   1644   1648   1653   1657   1669   1673   1679   1689   1695   1700
                1704   1708   1713   1717   1721   1725   1754   1764   1778   1788   1808   1811   1819
                1825   1900   2026   2067   2184   2188   2192   2353   2370   2390   2864   2865   2866
                2867   2869   2871   2872   2873   2874
GO      003312   512    523#
GOOT    012502  1377#
GO1     003324   527#   532    537   1004
GO2     003336   525    530#
GO3     003344   531    533#
GTSWR = 104405   360   1193   1229   1318   2220   2869#
HDALGN  021076  2194#  2200   2202   2212   2214   2276
HDRFLG  001320   240#  1012*  1019   1025*
HT    = 000011    27#  2438   2479
IDEX    001324   244#   524*   676    702*
ILEGAL  004142   466    650    660#
INITIL  004524   723    729#   886
INITI2  004536   732#   753    760
INIT.2  010466  1170#
INSYS2  003526   568#   600
IO      012540  1367   1369   1371   1373   1390#  1473   1552
IOTVEC= 000020   122#
KYTEMP  001330   250#   456*   457*   458    460*   483*   484    485    487    501*   502    503*   505
                 507*   508*   511    648    651*   652    653*
LDFLG   004224   679#  1132
LDSEEK  011604  1262#  1297
LF    = 000012    28#  1381   2473   2479
LFLF  = 105212   274#
LOGA    001166   196#   454    511*   523    538    567    679    905   1072   1126
MANSEL  010356  1057   1139#  1757   1829
MASK    005234   613    678    816#   921
MASKER  004030   596    598    632#
MODE    001312   141*   234#   314*   367
MORE    013066  1440#  1443
MOUNT   004316   699    702#
MSKTBL  001256   210#   677   1068
NEXT    012416  1366#  1376
NG      002614   419    422#
NXT1    017604  1890   1905   1908   1921#
NXT2    017700  1923   1929   1936   1939   1955#
```

```
OSPFLG  001326   246#  374*  1206  1224*
PASTBL  001246   208#  1070  1085
PATTRN  001362   263#  792*  796   803*  806*  920*  940*  943*  1016  1728*  1735  1766*  1768
PIRQ  = 177772    34#
PIRQVE= 000240   128#
PRONUM  001313   235#  494*  519*
PRO1    003306   488   519#
PRO2    003160   486   494#  518
PRTTWO  014626  1637#
PR0   = 000000    51#
PR1   = 000040    52#
PR2   = 000100    53#
PR3   = 000140    54#
PR4   = 000200    55#
PR5   = 000240    56#
PR6   = 000300    57#
PR7   = 000340    58#
PS    = 177776    31#  32
PSW   = 177776    32#
PWRVEC= 000024   123#  331*  332*  2879*  2880*  2889*  2895*  2907*  2908*
RBA     001414   278#  1410  1438  1439
RBUFF   013702   278   1562#
RDBUFF  007356   927   1015  1137#  1742  1748  1791  1798  2268
RDCHK   006450   932   1010#
RDCHK0  017020  1785  1790#
RDCHR = 104407   482   500   1580  2741  2872#
RDLIN = 104410   452   2196  2810  2873#
RDLINK  005632   536   547   616   902#  1133
RDOCT = 104411   416   582   1123  2874#
RDTBL   001226   207#
RD1     005656   908#  913   924   947
RD2     005670   907   911#
RD3     005676   912   914#
RD4     005740   923   925#  946
RD5     005746   926#  935   941
RD6     005752   927#
RD7     006052   938   942#
READCS  001356   261#  930   1744  1793
RESTRT  006232   965   978#
RESVEC= 000010   118#
RETFR2  007270  1118#
RETRN2  006446   968   977   1005#
RETRN3  005374   855   857#
RETRN4  004036   633   635#
RETRY   013214  1461  1469#  1479  1489  1556
RFCERR  013262  1441  1484#
RFERR   013244  1419  1477#
RKBA    001374   268#  867*  1393*  1410*  1427*  1735*  1742*  1768*  1791*  2268*  2328
RKCS    001370   266#  736*  738   763*  765   869*  871   955   958*  961   969*  972   1171*
                1173  1177*  1179  1232*  1233  1265*  1267  1282*  1284  1353  1355*  1358  1360*
                1394*  1395*  1396*  1397*  1398*  1399  1401  1411*  1412*  1413*  1414*  1415*  1416
                1418  1428*  1429*  1430*  1431*  1432*  1448  1450  1469*  1470*  1485  1547  1632*
                1634  1661*  1663  1675  1681*  1683  1691  1730*  1732  1737*  1739  1744*  1746
                1770*  1772  1780*  1782  1793*  1795  1845*  1846  1864*  1902*  1903  1963*  1965
                2051*  2052  2055*  2057  2074*  2075  2092*  2094  2230*  2231  2233*  2234  2247
                2249*  2250  2257*  2258  2269*  2271  2321*  2322  2329
```



```
RKDA    001376   269#  735*  746*  762*  866*  1162*  1170  1176*  1263*  1264*  1273*  1280*  1281*
                1291*  1351*  1352*  1391*  1392*  1408*  1409*  1425*  1426*  1490*  1492*  1498  1631*
                1660*  1729*  1734*  1741*  1767*  1790*  1853*  1884*  1885*  1961*  2050  2054*  2088*
                2091*  2229*  2256*  2266*  2304*  2326  2377*
RKDS    001364   264#  747   754   767   966   974   1163  1225  1274  1292  1356  1361  1471
                1760  1854  1866  1886  1896  1921  1925  1957  1968  2004  2013  2060  2089
                2098  2305  2312  2349  2372
RKER    001366   265#  1665  1685  1774  1784
RKWC    001372   267#  968*  1390*  1407*  1424*  1736*  1743*  1769*  1792*  2267*  2327
RWC     001416   279#  1347*  1407  1436
RWS   = 000100   290#  1274  1292  1866  2004  2060  2098  2349
RWSRDY  011356  1207  1225#  1226
SEC     013564  1502*  1506*  1529#
SECCNT  001344   256#  929   1736  1743  1769  1792
SECOND  003452   369   560#
SECONE  006752   675   1066#
SECSYS  003050   468   476#  493   658
SECTBL  001305   228#  906   936
SECT.0  013772   431   1575#
SECT.1  012002   430   1311#
SECT.2  010370   429   1154#
SECT.3  002642   428   442#
SECT.4  017300   432   1844#
SECT.5  020634   433   2181#
SECT.6  021544   434   2302#  2310
SEEKI   001402   271#  1253*  1260
SEEKO   001404   272#  1255*  1259
SEKSET  011570  1254  1258#
SHFCNT  020444  1321*  1323  1325*  2111#
SHF1    013616  1507  1511  1541#
SHF2    013614  1540#
SHF3    013612  1503  1515  1519  1539#
SIN   = 001000   292#  2013
SINST   020200  2014  2042#
SMTME   005614   737   764   870   892#  959   971   1172  1178  1266  1283  1633  1662  1682
                1731  1738  1745  1771  1781  1794  1865  2097
STACK = 001100    22#  316   327   1003
START   001440   142   315#  555   1383  1468  1535  1727
STARTR  001434   139   314#
STFLG   001325   245#  370   373*
STKLMT= 177774    33#
STOP    013604  1524  1534#
STORE   004072   462   648#
STRT1   001774   374#  427
SUR     013576  1510*  1532#
SWR     001140   180#  325   337*  339   345*  358   1191  1196  1227  1235  1316  1323  2218
                2223  2252  2261  2274  2593  2632  2679*  2887  2900*
SWREG   000176   137#  345   358   1191  1227  1316  2218  2593  2632  2655
SWTCH   021530  2223*  2224*  2239  2244  2254  2279#
SW0   = 000001    86#
SW00  = 000001    76#  86
SW01  = 000002    75#  85
SW02  = 000004    74#  84
SW03  = 000010    73#  83
SW04  = 000020    72#  82
SW05  = 000040    71#  81
```

```
SW06  = 000100            70#     80
SW07  = 000200            69#     79
SW08  = 000400            68#     78
SW09  = 001000            67#     77
SW1   = 000002            85#
SW10  = 002000            66#
SW11  = 004000            65#
SW12  = 010000            64#
SW13  = 020000            63#
SW14  = 040000            62#
SW15  = 100000            61#
SW2   = 000004            84#
SW3   = 000010            83#
SW4   = 000020            82#
SW5   = 000040            81#
SW6   = 000100            80#
SW7   = 000200            79#
SW8   = 000400            78#
SW9   = 001000            77#
SYSER   013160          1462#   1480    1557
SYSERR  013674          1555    1557#
S10     012174          1324    1330#
S11     012152          1325#   1382
S12     012272          1347#
S13     012142          1323#   1329
TABLTY  002554           372     411#    423
TBITVE= 000014           119#
TBLFUL  004134           656     658#
TIME    005500           872     879#
TIMR    001346           257#    732*    740*    781*    769*    863*    879*    892*    893*
TIMR2   001323           243#    733*    771     773*    885*    882     884*
TKVEC = 000060           126#   2578*   2579*
TPVEC = 000064           127#
TRAPVE= 000034           125#    329*    330*
TRTVEC= 000014           120#
TRYAGN  011366          1227#   1244
TSTSN1  006154           957     964     966#
TYPE  = 104401           354     375     379     383     387     391     395     399     403     407     412     422     444
                         448     477     484     489     496     502     514     551     561     569     578     617     661
                         706     714     718     742     749     756     776     887     981     987     992    1021    1030
                        1035    1043    1049    1086    1090    1095    1103    1119    1146    1155    1183    1187    1208
                        1212    1216    1220    1240    1307    1312    1331    1332    1378    1463    1494    1523    1571
                        1576    1597    1605    1609    1613    1619    1625    1638    1642    1646    1651    1655    1667
                        1671    1677    1687    1693    1698    1702    1706    1711    1715    1719    1723    1752    1762
                        1776    1786    1804    1809    1817    1823    1857    1860    1874    1892    1895    1898    1909
                        1912    1931    1935    1942    1945    1974    1977    1985    1988    2024    2034    2037    2045
                        2048    2065    2182    2186    2190    2194    2314    2316    2318    2351    2368    2388    2443
                        2539    2599    2653    2654    2657    2670    2681    2700    2748    2754    2759    2763    2768
                        2769    2771    2774    2778    2864#   2909
TYPOC = 104402           1334    1859    1894    1911    1933    1944    1976    1987    2029    2036    2047    2315    2320
                        2393    2656    2865#
TYPON = 104404           1048    1054    1822    1828    2867#
TYPOS = 104403            575     622     711    1027    1040    1100    1108    1582    1602    1814    2866#
WATSK   020030           1983    2002#
WC      001412            277#   1348*   1390    1424
WCERR   013624           1455    1547#
```



```
WCERRR  013140           1451    1455#
WCHERR  013654           1548    1553#
WFERR   013144           1402    1459#
WPS   = 000040            291#   1896    1925
WRDCNT  001340            254#    797*    868     929*
WRITCS  001354            260#    798    1737    1770
WRLINK  005104            535     615     788#
WRPRO   016470           1624    1636    1760#
WRRDCK  016260           1618    1630    1728#   1756
WRTNBY  001317            239#    918*   1026
X       015610           1692    1697#
XX      016024           1710#
$AUTOB  001134            177#    362*   2646    2795
$BDADR  001122            172#
$BDDAT  001126            174#
$BELL   024066           2599    2788#
$CHARC  022504           2445*   2455*   2482    2471*   2476#
$CKSWR  023210           2632#   2871
$CMTAG  001100            160#    322     323
$CM3  = 000000            190#
$ NTLG  024077           2653    2790#
$CNTLU  024072           2670    2763    2789#
$CRLF   001161            191#   1331    1874    2444    2479    2681    2788    2788
$ENDAD  001400            149     270#
$ERFLG  001103            163#
$ERMAX  001115            169#
$ERRPC  001116            170#
$ERRTB  001434            307#
$ERTTL  001112            167#
$FILLC  001156            188#   2448    2479
$FILLS  001155            187#   2479
$GDADR  001120            171#
$GDDAT  001124            173#
$GTSWR  023300           2654#   2869
$HD   = 000003             11      12
$HIOCT  024226           2827*   2832#
$ICNT   001104            164#
$ILLUP  024450           2879    2895    2912#
$INTAG  001135            178#   2651*   2682    2795
$ITEMB  001114            168#
$LF     001162            192#   2479    2778    2788
$LPADR  001106            165#
$LPERR  001110            166#
$MAIL = ****** U          349     358    2432
$MNEW   024115           2657    2793#
$MSWR   024104           2654    2791#
$NULL   001154            186#   2450    2479
$OCNT   022732           2511*   2540*   2553#
$OMODE  022734           2506*   2510*   2515    2518*   2529*   2555#
$PASS   001100            161#   1129*   1134
$POWER  024456           2910    2915#
$PWRDN  024310            331    2879#   2907
$PWRMG  024444           2910#
$PWRUP  024362           2889    2895#
$QUES   001160            190#    422    2479    2700    2771    2788
$RDCHR  023512           2713#   2872
```

```
$RDDEC= ****** U        2875
$RDLIN  023602          2736#   2873
$RDOCT  024126          2805#   2874
$RDSZ = 000030          2729#
$R2A  = ****** U        2875
$SAVRE= ****** U        2875
$SAVR6  024454          2888*   2896    2897*   2898*   2914#
$SETUP= 000114           314#    328     329     331     333     354     355    2591    2596    2597    2627    2795
$STUP = 177777           314#
$SVPC = 000222           147#    152
$SWR  = 160000            11      12#     190    2911
$TKB    001146           183#   2559    2580    2589    2608    2636    2663
$TKCNT  022736          2560#   2575*   2597    2614*   2720    2722*
$TKINT  022746           349    2575#   2649
$TKQEN= 022745          2564#   2622    2725
$TKQIN  022740          2561#   2576*   2577    2620*   2621*   2622    2624*
$TKQOU  022742          2562#   2577*   2723    2724*   2725    2727*
$TKQSR  022744          2563#   2578    2624    2727
$TKS    001144           182#   2559    2581*   2604*   2608    2612*   2634    2650*   2660    2684*
$TKSRV  023016          2578    2589#
$TN   = 000001            11#
$TPB    001152           185#   2468*   2479
$TPFLG  001157           189#   2426    2479
$TPS    001150           184#   2466    2479
$TRAP   024230           329    2841#
$TRAP2  024252          2852#   2863
$TRP  = 000012          2856#   2865#   2866#   2867#   2868#   2869    2870#   2871    2872#   2873#   2874#   2875#
$TRPAD  024264          2846    2863#
$TSTNM  001102           162#
$TTYIN  024036          2738    2739    2751    2769    2783    2787#
$TYPBN= ****** U        2868
$TYPDS= ****** U        2868
$TYPE   022270          2426#   2856    2864
$TYPEC  022440          2447    2454    2461    2466#   2467    2686
$TYPEX  022506          2472    2474    2477#
$TYPOC  022534          2509#   2865
$TYPON  022550          2508    2511#   2867
$TYPOS  022510          2504#   2866
$OFILL  022733          2505*   2509*   2519    2554#
.     = 024466           131#    135#    140#    147     148#    150#    152#    159#    193     196#    207#    208#    226#
                         248#    326     366#    378#    382#    386#    390#    394#    398#    402#    406#    410#    447#
                         451#    564#    572#    620#    717#    745#    890#   1033#   1052#   1098#   1137#   1158#   1234
                        1243#   1315#   1381#   1466#   1497#   1533#   1562#   1600#   1616#   1641#   1649#   1654#   1658#
                        1670#   1709#   1718#   1755#   1765#   1779#   1812#   1826#   1901#   2027#   2151#   2193#   2259
                        2300#   2323    2336    2354#   2362    2371#   2373    2376    2384    2391#   2479    2559    2563#
                        2564    2565#   2787#   2788    2795    2891    2913
```



```
COMMEN      1#    129#
ENDCOM      1#    129#
ERROR      23#
ESCAPE      1#    129#
GETPRI      1#    129#
GETSWR      1#    129#    355#
MORETA    153#    194
MULT        1#    129#
NEWTST      1#    129#
POP         1#    129#   2828    2900    2901
PUSH        1#    129#   2807    2881    2887
REPORT      1#    129#
SCOPE      24#
SETPRI      1#    129#   2716
SETTRA   2856#   2865    2866    2867    2869    2871    2872    2873    2874
SETUP       1#    129#    321
SKIP        1#    129#
SLASH       1#    129#
SPACE     129#
STARS       1#    129#    145     155     193     308     312    2411    2481    2558    2627    2642    2705    2729    2797
         2835    2877    2893
SWRSU       1#    129#    333#
TRMTRP   2856#
TYPBIN      1#    129#
TYPDEC      1#    129#
TYPNAM      1#    129#    350
TYPNUM      1#    129#
TYPOCS      1#    129#    710    1601
TYPOCT      1#    129#   2655
TYPTXT      1#    129#    375     379     383     387     391     395     399     403     407     411     443     448     478
          489     496     513     550     560     568     578     617     661     706     714     718     742     749     756
          775     887     981     986     991    1021    1030    1034    1043    1049    1086    1090    1095    1103    1118
         1145    1154    1183    1187    1208    1212    1216    1220    1240    1306    1311    1377    1462    1493    1570
         1575    1597    1605    1609    1613    1619    1625    1637    1642    1646    1651    1655    1667    1671    1677
         1687    1693    1697    1702    1706    1711    1715    1719    1723    1752    1762    1776    1786    1804    1809
         1817    1823    1898    2023    2065    2182    2186    2190    2351    2368    2388
$$CMRE    153#
$$CMTM    153#
$$ESCA      1#    129#
$$NEWT      1#    129#
$$SET    2856#   2865    2866    2867    2869    2871    2872    2873    2874
$$SKIP      1#    129#
.EQUAT      1#     19
.HEADE      1#
.KT11       1#
.SETUP      1#    314
.SWRH1      1#
.$ACT1      1#    143
.$APTB      1#
.$APTH      1#
.$APTY      1#
.$ASTA      1#
.$CATC      1#    129
.$CMTA      1#    153
.$DB2D      1#
.$DB2O      1#
```

```
.$DIV       1#
.$EOP       1#
.$ERRO      1#
.$ERRT      1#
.$MULT      1#
.$POWE      1#   2875
.$RAND      1#
 $RDDE      1#
.$RDOC      1#   2795
.$READ      1#   2556
.$R2AZ      1#
.$SAVE      1#
.$SB2D      1#
.$SB2O      1#
.$SCOP      1#
.$SIZE      1#
.$SUPR      1#
.$TRAP      1#   2833
.$TYPB      1#
.$TYPD      1#
.$TYPE      1#   2409
.$TYPO      1#   2479
.$40CA      1#
.1170       1#


. ABS.  024466     000


 ERRORS DETECTED:  0

 DZRKIF.BIN,DZRKIF.LST/CRF/SOL=DZRKKE.SML,DZRKIF.P11
 RUN-TIME: 12 16 1 SECONDS
 RUN-TIME RATIO: 102/31=3.3
 CORE USED:  33K  (65 PAGES)
```