RK05/RK11

UTILITY PACKAGE

MD-11-DZRKI-E

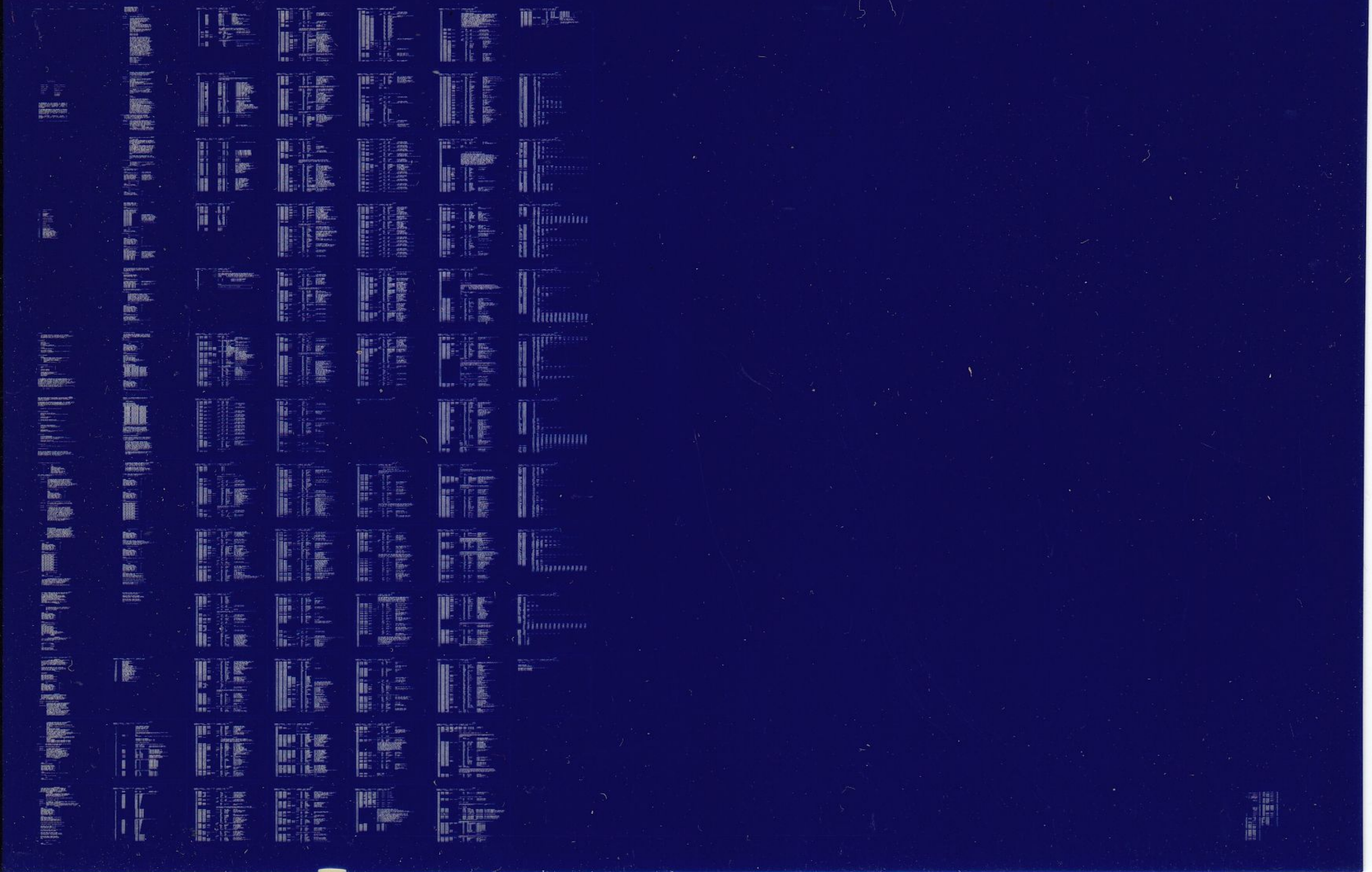
EP-DZRKI-E-DL-C

FICHE 1 OF 1

JUN 1977

digital

MADE IN USA



B01

EOF!DZBMOJSEQ	00010000	770610	PDP10 411	-THOR1DZRKIESEQ	00010000	770610

C01

IDENTIFICATION

PRODUCT CODE:	MAINDEC-11-DZRKI-E-D
PRODUCT NAME:	RK11 UTILITY PACKAGE
DATE CREATED:	MARCH, 1977
MAINTAINER:	DIAGNOSTIC GROUP
AUTHOR:	BOB COLLINS
REVISED BY:	JIM KAPADIA TOM SAWYER CHUCK HESS

D01

EO1

TABLE OF CONTENTS

F01

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
POP-11 PROCESSOR
8K MEMORY
RK11 OR RKV11 CONTROLLER
1-8 RKDS OR RKDSF 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
   RKOS CONTROL PANEL TEST           4
   RKOS 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 L01

                          **********
          ....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 TA ES, 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 M  LY 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 S EKS 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 M01
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
RKDS CONTROL PANEL TEST           4
RKDS CONTROL PANEL TEST #2        5
HEAD ALIGNMENT ROUTINE            6
POWER FAILURE (WRITE) TEST        7

TYPE=
```

****

9.6 SECTION 5 RKDS 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 DR2
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 IS 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
```

```
RKOS CONTROL PANEL TEST                 4
RKOS 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
RKOS CONTROL PANEL TEST                  4
RKOS 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 R ET TIMED OUT                    ;IS NECESSARY, DIAGNOSTIC
DRIVE R ET TIMED OUT                    ;SHOULD BE RUN AGAINST THE
DRIVE R ET TIMED OUT                    ;FAILING DRIVE.
DRIVE R ET 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
   RKOS 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
RKOS CONTROL PANEL TEST                4
RKOS 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 W TE ENABLE
PRESS CO TIN WHEN DRIVE RDY
ERROR! DATA  ITT N BY DRIVE 1 CANNOT BE READ.
  AD R=     7  EX CTD=077400  RECVD=077600
  ADDR=     .7  EXPCTD=077400  RECVD=037600
  ADDR=     .7  EXPCTD=077400  RECVD=037600
  ADDR=     .7  EXPCTD=077400  RECVD=037600
  ADDR=     .7  EXPCTD=077400  RECVD=037600
ERROR! DATA WRITTEN BY DRIVE 1 CA OT BE READ.
  ADDR=     4  EX CTD=077400  RE V =077600
  ADDR=     4  EXPCTD=077400  R CVD=037600
  ADDR=     4  EXPCTD=077400  F VD=037600
  ADDR=     4  EXPCTD=077400  F VD=037600
  ADDR=     4  EXPCTD=077400  RECVD=037600
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=   764  EXPCTD=077400  R CVD=077600
  ADDR=   764  EXPCTD=077400  R CVD=037600
  ADDR=   764  EXPCTD=077400  R CVD=037600
  ADDR=   764  EXPCTD=077400  R VD=037600
  AD R=   764  EXPCTD=077400  R VD=037600
ERROR! D A WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=  2767  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 ERROR REGISTER BEFORE THE PROGRAM CORRECTS ANY ERRORS WHICH WHICH MAY HAVE OCCURRED.

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

3. TO INCREASE OR DECREASE THE NUMBER OF RETRYS ALLOWED BEFORE A DRIVE IS DECLARED DOWN, GO TO THE 'MOUNT' ROUTINE, MODIFY THE SETUP OF LOCATIONS 'ECNT' AND 'CNTSIN' AND YOU HAVE IT!

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

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

```
        RK11 UTILITY PACKAGE


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

TYPE=0

        RK11 UTILITY PACKAGE

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

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

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

```
        RK11 UTILITY PACKAGE

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

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

        RK11 UTILITY PACKAGE

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

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

PACK GOOD.

        RK11 UTILITY PACKAGE

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

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

WRITE PROTECT THE DRIVE THEN PRESS CONTINUE

CLEAR WRITE PROTECT THEN PRESS CONTINUE

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

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

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

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

RK11 UTILITY PACKAGE

MAINDEC-11-DZRKI-E MACY11 27(1006) 17-MAR-77 14:57
DZRKIE.P11 17-MAR-77 14:53 TABLE OF CONTENTS

```
MAINDEC-11-DZRKI-E     MACY11 27(1006)  17-MAR-77  14:57  PAGE 1
DZRKIE.P11    17-MAR-77 14:53

                                .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  TOT,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
```

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


                        ;*"SWITCH REGISTER" SWITCH DEFINITIONS
                        SW15=   100000
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06=   100
                        SW05=   40
                        SW04=   20
                        SW03=   10
                        SW02=   4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
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                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
            000052              .=52
000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
            000222              .=$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
```

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

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

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

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

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

        001330          .EVEN

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

```
  265  001366  177402                  RKER:   .WORD   177402
  266  001370  177404                  RKCS:   .WORD   177404
  267  001372  177406                  RKWC:   .WORD   177406
  268  001374  177410                  RKBA:   .WORD   177410
  269  001376  177412                  RKDA:   .WORD   177412
  270  001400  000000                  SENDAD: .WORD   0
  271  001402  000000                  SEEKI:  .WORD   0
  272  001404  000000                  SEEKO:  .WORD   0
  273
  274          105212                  LFLF=   105212
  275  001406  013656                  BA:     BUFF
  276  001410  000000                  DA:     .WORD   0
  277  001412  000000                  WC:     .WORD   0
  278  001414  013660                  RBA:    RBUFF
  279  001416  000000                  RWC:    .WORD   0
  280  001420  000000                  EXTR:   .WORD   0
  281  001422  000000                  ERRWF:  .WORD   0
  282  001424  000000                  ERRRF:  .WORD   0
  283  001426  000000                  ERRRFC: .WORD   0
  284  001430  000000                  ERRWCH: .WORD   0
  285  001432  000000                  ERRWCS: .WORD   0
  286
  287                                  ;BIT DEFINITIONS
  288
  289          010000                  DPL=BIT12
  290          000100                  RWS=BIT6
  291          000040                  WPS=BIT5
  292          001000                  SIN=BIT9
```

```
MAINDEC-11-DZRKI-E      MACY11 27(1006)  17-MAR-77  14:57  PAGE 7
DZRKIE.P11    17-MAR-77 14:53            ERROR POINTER TABLE

                                         .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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                         ;*NOTE1:          IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC).
                                         ;*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                                 SERRTB:
                                         ;;**************************************************************

                                         ;THE ERROR TABLE IS UNUSED IN THIS PROGRAM

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

```
                                STARTR: CLRB    @#MODE
                                START:  RESET                   ;CLEAR THE BUS
001442  005037  177776                  CLR     PS              ;LOWER PROCESSOR PRIORITY TO
                                                                ;ALLOW TTY KYBD INTERRUPTS

                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
001446  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
001452  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
001454  022706  001140                  CMP     #SWR,R6 ;;DONE?
001460  001374                          BNE     .-6             ;;LOOP BACK IF NO
001462  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
001466  012737  024032  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
001474  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
001502  012737  024112  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
001510  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.
001516  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
001522  012737  001556  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
001530  012737  177570  001140          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
001536  012737  177570  001142          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
001544  022777  177777  177366          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
001552  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT = -1
001554  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
001556  012716  001564          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
001562  000002                          RTI
001564  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
001572  012737  000174  001142          MOV     #DISPREG,DISPLAY
001600  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

001604  004737  022550                  JSR     PC,$TKINT       ;INITIALIZE THE TTY INTERRUPT HANDLER
                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
001610  005227  177777                  INC     #-1             ;;FIRST TIME?
001614  001045                          BNE     67$             ;;BRANCH IF NO
001616  104401  001654                  TYPE    ,68$            ;;TYPE ASCIZ STRING
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
001622  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
001626  001006                          BNE     69$             ;;BRANCH IF YES
001630  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
001636  001005                          BNE     70$             ;;BRANCH IF NO
001640  104405                          GTSWR                   ;;GET SOFT-SWR SETTINGS
001642  000403                          BR      70$
001644  112737  000001  001134  69$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
001652                          70$:
001652  000426                          BR      67$             ;;GET OVER THE ASCIZ
                                ;;68$:  .ASCIZ  <CRLF>/RK11 UTILITY PACKAGE/<15><12>/MAINDEC-11-DZRKI-E/<CRLF>
001730                          67$:
001730  105737  001312                  TSTB    @#MODE
001734  100002                          BPL     1$
001736  000137  003436                  JMP     @#SECOND
001742  105737  001325          1$:     TSTB    STFLG                   ;PRINT ONLY THE FIRST TIME
001746  001402                          BEQ     10$
001750  000137  002540                  JMP     TABLTY
```

```
 370  001754  105237  001325          10$:    INCB    STFLG
 371  001760  105037  001326          STRT1:  CLRB    OSPFLG
 372  001764  104401  001772                  TYPE    ,65$                ;;TYPE ASCIZ STRING
 373  001770  000423                          BR      64$                 ;;GET OVER THE ASCIZ
 374                                  ;;65$:  .ASCIZ  <15><12><15><12>/       NAME                    TYPE/
 375  002040                          64$:
 376  002040  104401  002046                  TYPE    ,67$                ;;TYPE ASCIZ STRING
 377  002044  000421                          BR      66$                 ;;GET OVER THE ASCIZ
 378                                  ;;67$:  .ASCIZ  <15><12>/INDEX                          0/
 379  002110                          66$:
 380  002110  104401  002116                  TYPE    ,69$                ;;TYPE ASCIZ STRING
 381  002114  000421                          BR      68$                 ;;GET OVER THE ASCIZ
 382                                  ;;69$:  .ASCIZ  <15><12>/COMPATIBILITY PACKAGE          1/
 383  002160                          68$:
 384  002160  104401  002166                  TYPE    ,71$                ;;TYPE ASCIZ STRING
 385  002164  000421                          BR      70$                 ;;GET OVER THE ASCIZ
 386                                  ;;71$:  .ASCIZ  <15><12>/OSCILLATING SEEK PACKAGE       2/
 387  002230                          70$:
 388  002230  104401  002236                  TYPE    ,73$                ;;TYPE ASCIZ STRING
 389  002234  000421                          BR      72$                 ;;GET OVER THE ASCIZ
 390                                  ;;73$:  .ASCIZ  <15><12>/FORMATTER-SURFACE VERIFIER     3/
 391  002300                          72$:
 392  002300  104401  002306                  TYPE    ,75$                ;;TYPE ASCIZ STRING
 393  002304  000421                          BR      74$                 ;;GET OVER THE ASCIZ
 394                                  ;;75$:  .ASCIZ  <15><12>/RK05 CONTROL PANEL TEST        4/
 395  002350                          74$:
 396  002350  104401  002356                  TYPE    ,77$                ;;TYPE ASCIZ STRING
 397  002354  000421                          BR      76$                 ;;GET OVER THE ASCIZ
 398                                  ;;77$:  .ASCIZ  <15><12>/RK05 CONTROL PANEL TEST #2      5/
 399  002420                          76$:
 400  002420  104401  002426                  TYPE    ,79$                ;;TYPE ASCIZ STRING
 401  002424  000421                          BR      78$                 ;;GET OVER THE ASCIZ
 402                                  ;;79$:  .ASCIZ  <15><12>/HEAD ALIGNMENT ROUTINE         6/
 403  002470                          78$:
 404  002470  104401  002476                  TYPE    ,81$                ;;TYPE ASCIZ STRING
 405  002474  000421                          BR      80$                 ;;GET OVER THE ASCIZ
 406                                  ;;81$:  .ASCIZ  <15><12>/POWER FAILURE (WRITE) TEST     7/
 407  002540                          80$:
 408  002540                          TABLTY:
 409  002540  104401  002546                  TYPE    ,65$                ;;TYPE ASCIZ STRING
 410  002544  000405                          BR      64$                 ;;GET OVER THE ASCIZ
 411                                  ;;65$:  .ASCIZ  <15><12><15><12>/TYPE=/
 412  002560                          64$:
 413  002560  104411                          RDOCT                       ;GET THE TEST NUMBER FROM THE OPERATOR
 414  002562  012600                          MOV     (SP)+,R0            ;STORE IT IN R0
 415  002564  022700  000010                  CMP     #10,R0              ;VALID NUMBER ?
 416  002570  003403                          BLE     NG                  ;BR IF NOT
 417  002572  006100                          ROL     R0                  ;ALIGN THE NUMBER FOR DISPATCHING
 418  002574  000170  002606                  JMP     @BEGIN(R0)          ;GO TO THE SELECTED TEST
 419  002600  104401  001160          NG:     TYPE    ,$QUES              ;'?'
 420  002604  000755                          BR      TABLTY
 421
 422                                  ;TEST ENTRY DISPATCH TABLE
 423
 424  002606  001760                  BEGIN:  STRT1
 425  002610  002626                          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.

002626  000240                  SECT.3: NOP                     ;NO-OP
002630                          AUTSL2:
002630  104401  002636                  TYPE    ,65$            ;;TYPE ASCIZ STRING
002634  000415                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/TERMINATE WITH '.<CR>'/
002670                          64$:
002670  104401  002676                  TYPE    ,67$            ;;TYPE ASCIZ STRING
002674  000417                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <15><12>/DRIVE NUMBERS ON SYSTEM 1=/
002734                          66$:
002734  104410                          RDLIN
002736  012600                          MOV     (SP)+,R0        ;PICK UP THE ADDRESS OF THE INPUT BUFFER
002740  012701  001166                  MOV     #LOGA,R1        ;GET THE ADDRESS OF THE LOGICAL UNIT TBL.
002744  105037  001311                  CLRB    @#DRCNT1        ;CLEAR THE DRIVE COUNTER
002750  005037  001330          1$:     CLR     @#KYTEMP        ;CLEAR TEMP
002754  112037  001330                  MOVB    (R0)+,@#KYTEMP  ;GET THE FIRST DRIVE #
002760  122737  000054  001330          CMPB    #54,@#KYTEMP    ;IS IT A COMMA THAT WAS TYPED?
002766  001770                          BEQ     1$              ;IF YES GO BACK
002770  162737  000060  001330          SUB     #60,@#KYTEMP    ;MAKE ASCII A DRIVE #
002776  100403                          BMI     2$              ;IF RESULT NEGATIVE BRANCH
003000  004737  004056                  JSR     PC,STORE        ;IF RESULT POSITIVE JUMP
003004  000761                          BR      1$              ;AFTER STORING GET NEXT #
003006  122740  000056          2$:     CMPB    #56,-(R0)       ;WAS NEGATIVE RESULT A TERMINATOR?
003012  001402                          BEQ     3$              ;IF YES BRANCH
003014  004737  004126                  JSR     PC,ILEGAL       ;IF NO BAD CHARACTER JUMP
003020  022701  001206          3$:     CMP     #DRVD,R1        ;IS THE TABLE FULL
003024  001403                          BEQ     SECSYS          ;IF YES BRANCH
003026  012721  100000                  MOV     #100000,(R1)+   ;IF NO FILL TABLE WITH DOWN INDICATOR.
003032  000772                          BR      3$              ;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

003034                          SECSYS:
003034  104401  003042                  TYPE    ,65$            ;;TYPE ASCIZ STRING
003040  000416                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/IS THERE A SECOND SYSTEM?/
003076                          64$:
003076  000240                          NOP                     ;***
003100  104407                          RDCHR                   ;READ A CHARACTER
003102  012637  001330                  MOV     (SP)+,@#KYTEMP  ;GET THE RESPONSE
003106  104401  001330                  TYPE    ,KYTEMP         ;ECHO
```

```
 482  003112  022737  000131  001330          CMP     #131,@#KYTEMP   ;WAS IT A "Y" (FOR YES)?
 483  003120  001411                          BEQ     PRO2            ;IF YES BRANCH TO PROCESSOR 2
 484  003122  022737  000116  001330          CMP     #116,@#KYTEMP   ;WAS RESPONSE LEGAL (N FOR NO)?
 485  003130  001460                          BEQ     PRO1            ;IF LEGAL BRANCH
 486  003132  104401  003140                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 487  003136  000401                          BR      66$             ;;GET OVER THE ASCIZ
 488                                  ;;67$:  .ASCIZ  /?/
 489  003142                          66$:
 490  003142  000734                          BR      SECSYS          ;GO BACK ASK AGAIN
 491  003144  152737  000377  001313  PRO2:   BISB    #377,@#PRONUM   ;SET FLAG TWO PROCESSORS
 492  003152  000240                          NOP                     ;***
 493  003154  104401  003162                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 494  003160  000406                          BR      64$             ;;GET OVER THE ASCIZ
 495                                  ;;65$:  .ASCIZ  <15><12>/DRIVE # =/
 496  003176                          64$:
 497  003176  104407                          RDCHR                   ;READ A CHARACTER
 498  003200  012637  001330                  MOV     (SP)+,@#KYTEMP  ;PICK UP THE RESPONSE
 499  003204  104401  001330                  TYPE    ,KYTEMP         ;ECHO
 500  003210  162737  000060  001330          SUB     #60,@#KYTEMP    ;MAKE IT A NUMBER
 501  003216  100420                          BMI     BADINP          ;IF NOT A NUMBER, BRANCH
 502  003220  022737  000010  001330          CMP     #10,@#KYTEMP    ;IS IT A LEGAL #?
 503  003226  003414                          BLE     BADINP          ;IF NO BRANCH
 504  003230  000337  001330                  SWAB    @#KYTEMP        ;GET THE DRIVE # TO THE HIGH BYTE
 505  003234  052737  040000  001330          BIS     #BIT14,@#KYTEMP ;SET THE SECOND SYSTEM BIT
 506  003242  113705  001311                  MOVB    @#DRCNT1,R5     ;GET THE DRIVE COUNT
 507  003246  006105                          ROL     R5              ;MAKE IT AN INDEX
 508  003250  013765  001330  001166          MOV     @#KYTEMP,LOCA(R5) ;STORE THE SYSTEM #2 WORD
 509  003256  000407                          BR      GO              ;GO DO THE TEST
 510  003260                          BADINP:
 511  003260  104401  003266                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 512  003264  000401                          BR      64$             ;;GET OVER THE ASCIZ
 513                                  ;;65$:  .ASCIZ  /?/
 514  003270                          64$:
 515  003270  000725                          BR      PRO2            ;GO BACK ASK AGAIN
 516  003272  105037  001313          PRO1:   CLRB    @#PRONUM        ;CLEAR THE FLAG ONE PROCESSOR
 517
 518                                  ;THIS IS THE ACTUAL PROGRAM
 519
 520  003276  012700  001166          GO:     MOV     #LOCA,R0        ;GET THE TABLE ADDRESS TO R0
 521  003302  105037  001324                  CLRB    @#IDEX          ;CLRB THE INDEX
 522  003306  000405                          BR      GO2             ;BRANCH AROUND INCREMENT ROUTINE
 523
 524  003310  062700  000002          GO1:    ADD     #2,R0           ;INDEX THRU THE TABLE
 525  003314  022700  001206                  CMP     #DRV0,R0        ;DONE?
 526  003320  001414                          BEQ     EXIT            ;IF YES GET OUT
 527  003322  005710                  GO2:    TST     (R0)            ;IS THE DRIVE ACTIVE
 528  003324  100001                          BPL     GO3             ;IF YES BRANCH
 529  003326  000770                          BR      GO1             ;NO TRY THE NEXT ONE
 530  003330  011037  001164          GO3:    MOV     (R0),@#DRACTV   ;PICK UP THE ACTIVE DRIVE WORD
 531  003334  004737  004160                  JSR     PC,CYCLE        ;CALL CYCLE (PICK UP DRIVE #, CYL BASE, AND CALL MOUNT)
 532  003340  004737  005070                  JSR     PC,WRLINK       ;CALL WRITE LINK TO LOAD REGISTERS FOR WRITE
 533  003344  004737  005616                  JSR     PC,RDLINK       ;CALL READ LINK TO LOAD REGISTERS FOR READ
 534  003350  000757                          BR      GO1             ;GO GET NEXT DRIVE
 535  003352  012700  001166          EXIT:   MOV     #LOCA,R0
 536  003356  022700  001206          EXITFR2: CMP    #DRV0,R0
 537  003362  001414                          BEQ     EXITX
```

```
                                        BIT     #BIT14,(R0)
                                        BNE     EXITX
                                        TST     (R0)+
                                        BMI     EXTFR2
                                        MOV     -(R0),R1
                                        JSR     PC,DO2
                                        JSR     PC,RDLINK
                                        TST     (R0)+
                                        BR      EXTFR2
                                EXITX:
                                        TYPE    ,65$                    ;;TYPE ASCIZ STRING
                                        BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/DONE!/
                                64$:
                                        JMP     @#START                 ;RESTART

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

                                SECOND:
                                        TYPE    ,65$                    ;;TYPE ASCIZ STRING
                                        BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/COMPATIBILITY-SYSTEM#2/
                                64$:
                                        MOV     #200,R4                 ;SET UP MASK BIT IN R4
                                        MOV     #1,R3                   ;SET UP WORD COUNTER IN R3
                                        MOV     #LOGA,R2                ;GET THE TABLE ADDRESS
                                INSYS2:
                                        TYPE    ,65$                    ;;TYPE ASCIZ STRING
                                        BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/WORD /
                                64$:
                                        NOP                             ;***
                                        MOV     R3,-(SP)                ;GET READY TO TYPE WORD COUNTER
                                        TYPOS
                                        .BYTE   6
                                        .BYTE   0
                                        TYPE    ,67$                    ;;TYPE ASCIZ STRING
                                        BR      66$                     ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /=/
                                66$:
                                        RDOCT
                                        MOV     (SP)+,R0                ;PICK UP THE OCTAL WORD
                                        MOVB    R0,R1                   ;GET THE FIRST DRIVE TO R1
                                        SWAB    R0                      ;GET THE SECOND DRIVE TO R0 LOW BYTE
                                        BIC     #177400,R0              ;CLEAR THE UNUSED BITS
                                        BIC     #177400,R1              ;CLEAR THE UNUSED BITS
                                        ROR     R0                      ;ROTATE RIGHT R0
                                        BCC     2$                      ;IF CARRY IS CLEAR BRANCH
                                        BIS     #BIT7,R0                ;SET DOWN BIT IF CARRY SET
                                        CLC                             ;AND CLEAR THE CARRY BIT
                                2$:     ROR     R1                      ;NOW DO THE SAME FOR R1
                                        BCC     3$                      ;IF NO ERROR, BRANCH
                                        BIS     #BIT7,R1                ;IF ERROR SET THE BIT
                                3$:     MOVB    R1,1(R2)                ;SET DRIVE FIRST IN TABLE
                                        JSR     PC,MASKER               ;CALL THE MASK CONTROL SUBROUTINE
```

```
                                        MOVB    R0,1(R2)        ;SET DRIVE SECOND IN TABLE (NEXT WORD)
                                        JSR     PC,MASKER       ;CALL THE MASK CONTROL SUBROUTINE
                                        INC     R3              ;INCREMENT THE WORD COUNTER
                                        BR      INSYS2          ;GO BACK AND GET NEXT WORD
                                EXITA:  BIS     R4,(R2)         ;FILL THE TABLE (LAST WORD)
                                        ROR     R4              ;WITH ALL BITS SET
                                        BCC     EXITA           ;GO BACK IF NOT DONE
                                EXITB:  BIC     #174000,(R2)    ;CLEAR DOWN AND SECOND SYSTEM BITS AT TABLE
                                        MOV     R2,R0           ;GET ADDRESS TO R0 (CURRENT TABLE)
                                        TST     (R2)+           ;ADD 2 TO THE POINTER
                                FIL.DN: CMP     #DRV0,R2        ;TABLE FULL?
                                        BEQ     2$              ;IF YES BRANCH
                                        MOV     #BIT15,(R2)+    ;FILL THE TABLE
                                        BR      FIL.DN          ;GO BACK TRY AGAIN
                                2$:     MOV     (R0),R1         ;GET THE WORD TO R1
                                        MOVB    R1,R2
                                        JSR     PC,MASK
                                        JSR     PC,DO2          ;FORM DRIVE # FOR MOUNT
                                        JSR     PC,WRLINK       ;WRITE NEW INFO
                                        JSR     PC,RDLINK       ;READ SAMPLE (ALL DRIVES)
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/DONE SYSTEM 2 RESTART SYSTEM 1, TYPE WORD /
                                64$:
                                        MOV     (R0),-(SP)      ;GET WORD FOR SYSTEM 1 AND TYPE
                                        TYPOS
                                        .BYTE   6
                                        .BYTE   1
                                1$:     HALT                    ;HALT (FINISHED SYSTEM #2)
                                        BR      1$              ;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.

                                MASKER: TST     (R2)            ;IS THE DRIVE UP
                                        BMI     RETRN4          ;IF NO, BRANCH
                                        MOVB    R4,(R2)         ;MOVE THE MASK BIT IN THE TABLE
                                RETRN4: ROR     R4              ;ROTATE THE MASK
                                        BCC     1$              ;DONE?,IF NO,BRANCH
                                        MOV     #EXITB,(SP)     ;SET UP FOR RETURN
                                        RTS     PC
                                1$:     BIT     #BIT14,(R2)     ;IS THIS SYSTEM # 2'S DRIVE?
                                        BEQ     2$              ;IF NO,BRANCH
                                        MOV     #EXITA,(SP)     ;SET UP FOR RETURN
                                        RTS     PC
                                2$:     ADD     #2,R2           ;INDEX THRU LOGA TABLE
                                        RTS     PC              ;RETURN

                                ;ROUTINE TO BUILD THE ACTIVE LOGICAL UNIT TABLE

                                STORE:  CMP     #10,@#KYTEMP    ;IS INPUT A LEGAL NUMBER?
                                        NOP                     ;***
                                        BLE     ILEGAL          ;IF NOT BRANCH
                                        SWAB    @#KYTEMP        ;ALIGN DRIVE # FOR TABLE
                                        MOV     @#KYTEMP,(R1)+  ;PUT THE WORD IN THE TABLE
```

```
                                        CLR     @#KYTEMP        ;CLEAR THE TEMP WORD
                                        INCB    @#DRCNT1        ;INCREMENT THE COUNTER
                                        CMP     #DRVO,R1        ;IS THE TABLE FULL?
                                        BEQ     TBLFUL          ;IF YES BRANCH
                                        RTS     PC              ;IF NO RETURN
                                TBLFUL: MOV     #SECSYS,(SP)    ;IF TABLE FULL SET UP FOR SYSTEM #2
                                        RTS     PC              ;RETURN
                                ILEGAL: MOV     #AUTSL2,(SP)    ;IF ILLEGAL RESPONSE, GO BACK
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /ILLEGAL INPUT/
                                64$:
                                        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

                                CYCLE:  MOV     (R0),R1         ;GET THE LOGICAL UNIT ACTIVE WORD TO R1
                                        BIT     #BIT14,R1       ;IS IT ON SYSTEM #2
                                        BEQ     CYCL2           ;IF NO BRANCH
                                        JMP     @#SECONE        ;GO TO SYSTEM #2
                                CYCL2:  MOVB    @#IDEX,R3       ;GET THE INDEX VALUE
                                        MOVB    MSKTBL(R3),R2   ;GET THE MASK TO R2
                                CYCLE2: JSR     PC,MASK         ;CALL THE MASK SUBROUTINE
                                LDFLG:  MOV     #LOGA,R3        ;GET TABLE ADDRESS TO R3
                                2$:     CMP     R0,R3           ;IS ACTIVE ADDRESS=TO FIRST ADDRESS
                                        BNE     3$              ;IF NO BRANCH
                                        BIS     R2,(R3)+        ;IF YES SET BITS TO SHOW WRITE
                                        BR      4$              ;BRANCH TO SEE IF DONE
                                3$:     BIC     R2,(R3)+        ;CLEAR BITS TO SHOW OVER-WRITE
                                4$:     CMP     #DRVO,R3        ;DONE
                                        BNE     2$              ;IF NO GO BACK
                                DO2:    SWAB    R1              ;GET DRIVE # TO LOW BYTE
                                        NOP                     ;***
                                        MOVB    R1,R2           ;GET IT TO R2
                                        BIC     #370,R2         ;CLEAR THE UNUSED BITS
                                        MOVB    R2,@#DRIVE      ;GET THE DRIVE #
                                        ROL     R2              ;SHIFT THE DRIVE # TO
                                        ROL     R2              ;ALIGN IT FOR THE DRIVE ADDR.
                                        ROL     R2              ;KEEP IT MOVING!
                                        ROL     R2              ;A LITTLE MORE!
                                        ROL     R2              ;THERE IT IS
                                        MOVB    R2,@#DSKTMP+1
                                        MOVB    R2,@#DSKADR+1   ;GET IT TO DISK ADDR. TEMP.
                                        JSR     PC,MOUNT        ;CALL MOUNT
                                        RTS     PC              ;RETURN

                                MOUNT:  INCB    @#IDEX          ;INCREMENT THE INDEX
                                        NOP                     ;***
                                        MOVB    #5,@#ECNT       ;SET ERROR CNTR TO 5
                                        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$:
004362                                  MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
004362  013746  001314                    TYPOS                   ;;GO TYPE--OCTAL ASCII
004366  104403                          .BYTE   1               ;;TYPE 1 DIGIT(S)
004370     001                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
004371     000                          TYPE    ,67$            ;;TYPE ASCIZ STRING
004372  104401  004400                  BR      66$             ;;GET OVER THE ASCIZ
004376  000415                  ;;67$:  .ASCIZ  <15><12>/MAKE PACK WRITE ENABLE/
004432                          66$:
004432  104401  004440                  TYPE    ,69$            ;;TYPE ASCIZ STRING
004436  000420                          BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <15><12>/PRESS CONTINUE WHEN DRIVE RDY/
004500                          68$:
004500  000000                          HALT
004502  004737  004510                  JSR     PC,INITIL       ;CALL INITIALIZER
004506  000207                          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

004510  010046                  INITIL: MOV     R0,-(SP)        ;SAVE R0
004512  013700  001334                  MOV     @#OSKADR,R0     ;GET THE DRIVE # TO R0
004516  042700  001777                  BIC     #1777,R0        ;CLEAR THE UNUSED BITS
004522  005037  001346          INITI2: CLR     @#TIMR          ;CLEAR THE TIMER
004526  105037  001323                  CLRB    @#TIMR2
004532  000240                          NOP                     ;***
004534  010077  174636                  MOV     R0,@RKDA        ;GET DRIVE # TO 'DA' REGISTER
004540  012777  000001  174622          MOV     #1,@RKCS        ;ISSUE CONTROL RESET + GO
004546  004737  005600                  JSR     PC,SMTME
004552  105777  174612          1$:     TSTB    @RKCS           ;DID CONTROL READY SET
004556  100423                          BMI     2$              ;IF YES BRANCH
004560  005237  001346                  INC     @#TIMR          ;IF NO INCREMENT THE TIMER
004564  001372                          BNE     1$              ;IF TIMER NOT ZERO BRANCH
004566  104401  004574                  TYPE    ,65$            ;;TYPE ASCIZ STRING
004572  000415                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/CONTROL RESET TIME OUT/
004626                          64$:
004626  010077  174544          2$:     MOV     R0,@RKDA        ;DRIVE NUMBER TO 'DA' REG.
004632  105777  174526                  TSTB    @RKDS           ;IS DRIVE READY
004636  100415                          BMI     3$              ;IF YES BRANCH
004640  104401  004646                  TYPE    ,67$            ;;TYPE ASCIZ STRING
004644  000411                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <15><12>/DRIVE NOT READY/
004670                          66$:
004670  000714                          BR      INITI2          ;GO BACK TRY AGAIN
004672  032777  000040  174464  3$:     BIT     #BIT5,@RKDS     ;IS DRIVE WRITE LOCKED?
004700  001420                          BEQ     4$              ;IF NO, BRANCH
004702  104401  004710                  TYPE    ,69$            ;;TYPE ASCIZ STRING
004706  000414                          BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <15><12>/DRIVE WRITE PROTECTED/
004740                          68$:
004740  000670                          BR      INITI2          ;YES, GO BACK TRY AGAIN
004742  005037  001346          4$:     CLR     @#TIMR          ;CLEAR THE TIMER
004746  010077  174424                  MOV     R0,@RKDA        ;GET THE DRIVE # TO 'DA' REGISTER
004752  012777  000015  174410          MOV     #15,@RKCS       ;ISSUE DRIVE RESET + GO
004760  004737  005600                  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$:     MOV     #PATTRN,@#BUSADR ;GET THE ADDRESS OF THE OUTPUT
                                        MOV     @#CYLCNT,@#RCCNT ;GET THE WORD COUNT
                                        MOV     @#WRITCS,@#CONTRL ;GET THE CONTROL + STATUS WORD
                                        JSR     PC,EXECUT       ;CALL EXECUTE
                                        BIT     #BIT4,@#DSKADR  ;WAS THIS WRITE SURFACE "1"?
                                        BNE     3$              ;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
                                        NOP                     ;***
                                        MOV     #200,R3         ;SET UP THE COMPARE MASK
                                        CLR     R4              ;CLR THE INDEX COUNTER
```

```
                                        MOV     #CYLTBL,R5      ;GET THE CYLINDER TABLE ADDRESS
                                1$:     BIT     R2,R3           ;IS THE MASK BIT SELECTED IN BASE
                                        BEQ     2$              ;IF NO BRANCH
                                        MOVB    R4,(R5)+        ;MOVE THE CYLINDER BASE TO THE TABLE
                                2$:     INCB    R4              ;INCREMENT THE BASE
                                        ROR     R3              ;ROTATE THE COMPARE MASK
                                        BCC     1$              ;IF NOT DONE GO BACK
                                        MOVB    #377,(R5)       ;IF DONE LOAD FINISH FLAG
                                        MOV     (SP)+,R5        ;RESTORE R5
                                        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

                                CYLADR: MOV     #BASE,R3        ;GET THE CYLINDER TABLE ADDRESS
                                CYLAD2: MOV     #CYLTBL,R2      ;GET THE SELECTED CYL BASE ADDR.
                                CYLOFF: MOVB    (R2),R4         ;GET THE SELECTED CYL VALUE TO R4
                                        NOP                     ;***
                                        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
                                        BIC     #017777,@#DSKADR;CLEAR ALL BUT DRIVE NUMBER
                                        BIS     R4,@#DSKADR     ;PUT IT IN DSKADR
                                        RTS     PC
                                BASINC: INC     R3              ;PICK UP ADDRESS OF NEXT BASE CYL.
                                        NOP                     ;***
                                        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

                                EXECUT: CLR     @#TIMR          ;CLEAR THE TIMER
                                        NOP                     ;HALT HERE TO CHECK COMMAND ABOUT TO BE EXECUTED
                                        CLRB    @#TIMR2         ;CLEAR SECOND TIMER
                                        MOV     @#DSKADR,@RKDA  ;LOAD THE DISK ADDRESS REGISTER
                                        MOV     @#BUSADR,@RKBA  ;LOAD THE BUS ADDRESS REGISTER
                                        MOV     @#WRDCNT,@RKWC  ;LOAD THE WORD COUNT REGISTER
                                        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
```

```
 874  005460  000742                          BR      EXECUT          ;TRY AGAIN
 875  005462  000207                  1$:     RTS     PC
 876  005464  005237  001346          TIME:   INC     @#TIMR
 877  005470  000240                          NOP                     ;***
 878  005472  001360                          BNE     CHECK1
 879  005474  105737  001323                  TSTB    @#TIMR2         ;SECOND TIMEOUT?
 880  005500  001013                          BNE     1$              ;IF YES BRANCH
 881  005502  105237  001323                  INCB    @#TIMR2         ;INDICATE SECOND TIMEOUT
 882  005506  000752                          BR      CHECK1          ;GO BACK
 883  005510  004737  004510          1$:     JSR     PC,INITIL
 884  005514  104401  005522                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 885  005520  000426                          BR      64$             ;;GET OVER THE ASCIZ
 886                                  ;;65$:  .ASCIZ  <15><12>/TIMED OUT ON OPERATION RETRY IN PROGRESS/
 887  005576                          64$:
 888  005576  000673                          BR      EXECUT
 889  005600  012737  000500  001346  SMTME:  MOV     #500,@#TIMR
 890  005606  005337  001346          1$:     DEC     @#TIMR
 891  005612  001375                          BNE     1$
 892  005614  000207                          RTS     PC
 893
 894                                  ;THIS ROUTINE CHECKS TO SEE IF A DRIVE IS ACTIVE FROM A
 895                                  ;WRITE OPERATION, IT GETS THE READ MASK TO R3, AND THE
 896                                  ;EXPECTED DATA PATTERN TO "PATTERN", AND THEN CALLS ADDRESS
 897                                  ;CONTROL.
 898
 899  005616  010046                  RDLINK: MOV     R0,-(SP)        ;SAVE R0
 900  005620  000240                          NOP                     ;***
 901  005622  152737  000377  001316          BISB    #377,@#COMND    ;INDICATE READ OPERATION
 902  005630  012701  001166                  MOV     #LOGA,R1        ;GET THE TABLE ADDRESS TO R1
 903  005634  012705  001305                  MOV     #SECTBL,R5      ;GET THE SECTOR TABLE ADDRESS
 904  005640  000405                          BR      RD2             ;SKIP OVER THE INDEX, FIRST PASS
 905  005642  062701  000002          RD1:    ADD     #2,R1           ;ADD 2 TO THE ADDRESS
 906  005646  022701  001206                  CMP     #DRV0,R1        ;ARE YOU THRU THE ENTIRE TABLE
 907  005652  001503                          BEQ     EXIT2           ;IF YES EXIT
 908  005654  005711                  RD2:    TST     (R1)            ;IS THE DRIVE ACTIVE
 909  005656  100001                          BPL     RD3             ;IF YES BRANCH
 910  005660  000770                          BR      RD1             ;IF NO GET NEXT WORD
 911  005662  011100                  RD3:    MOV     (R1),R0         ;GET THE ACTIVE WORD TO R0
 912  005664  010002                          MOV     R0,R2           ;COPY THE WORD
 913  005666  000300                          SWAB    R0              ;GET THE DRIVE # TO THE LOW BYTE
 914  005670  042700  177770                  BIC     #177770,R0      ;CLR ALL BUT THE DRIVE #
 915  005674  110037  001317                  MOVB    R0,@#WRTNBY     ;SAVE THE DRIVE #
 916  005700  006100                          ROL     R0              ;MAKE IT AN INDEX
 917  005702  016037  001206  001362          MOV     DRV0(R0),@#PATTRN ;PICK UP THE DATA PATTERN
 918  005710  004737  005220                  JSR     PC,MASK         ;CALL THE MASK SUBROUTINE
 919  005714  004737  005266                  JSR     PC,CYLADR       ;GO FORM A CYLINDER ADDRESS
 920  005720  000401                          BR      RD4             ;RETURN HERE IF NOT LAST BASE ADDR.
 921  005722  000747                          BR      RD1             ;IF LAST BASE ADDRESS, RETURN HERE
 922  005724  053737  001352  001334  RD4:    BIS     @#OSKTMP,@#OSKADR ;SET THE DRIVE # BITS IN DISK ADDR.
 923  005732  152537  001334          RD5:    BISB    (R5)+,@#OSKADR  ;SET THE SECTOR BITS IN DISK ADDR.
 924  005736  012737  007342  001336  RD6:    MOV     #IOBUFF,@#BUSADR ;GET THE BUFFER ADDR.
 925  005744  000240                          NOP                     ;***
 926  005746  013737  001344  001340          MOV     @#SECCNT,@#WRDCNT ;GET THE WORD COUNT
 927  005754  013737  001356  001332          MOV     @#READCS,@#CONTRL ;GET THE READ CONTROL WORD
 928  005762  004737  005366                  JSR     PC,EXECUT       ;DO THE READ
 929  005766  004737  006434                  JSR     PC,RDCHK        ;CHECK THE DATA
```

```
                                                    ;CLR THE SECTOR BITS IN DISK ADDR.
                                                    ;WAS THIS THE LAST SECTOR?
                                                    ;IF NO GO BACK
                                                    ;IF YES RESET SECTOR POINTER
                                                    ;WAS IT SURFACE "1" THAT WAS READ?
                                                    ;IF YES, BRANCH
                                                    ;NO, SET SURFACE "1" BIT
                                                    ;MAKE HEAD "1" PATTERN
                                                    ;GO BACK AND EXECUTE
                                                    ;CLEAR THE SURFACE BIT
                                                    ;MAKE IT SURFACE 0 DATA
                                                    ;INCREMENT THE SELECTED CYL TABLE POINTER
                                                    ;GO FORM NEXT ADDRESS
                                                    ;IF HERE IT IS NOT THE LAST BASE ADDR
                                                    ;IF HERE GET NEXT WORD-DRIVE FINISHED

                                                    ;RESTORE R0
                                                    ;RETURN TO MAIN LINE CODE

                                                    ;THE ERROR CHECK ROUTINE CHECKS IF AN ERROR OCCURRED
                                                    ;ON WRITING OR READING.

                                                    ;HARD ERROR OR ERROR SET?
                                                    ;HALT HERE TO EXAMINE ERROR REG.,ECT.
                                                    ;IF NO, GO TEST 'SIN' BIT
                                                    ;IF YES, ISSUE CNTROL RESET + GO

                                                    ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)
                                                    ;CNTROL READY BIT SET (FROM CNTROL RESET)
                                                    ;IF NO WAIT.  (IF HUNG HERE RUN STATIC)
                                                    ;DECREMENT ERROR COUNTER
                                                    ;HAVE ERROR BITS SET 5 TIMES?
                                                    ;IF HERE 5 ERRORS HAVE OCCURRED
                                                    ;SEEK INCOMPLETE SET?
                                                    ;***
                                                    ;BRANCH IF NO
                                                    ;IF YES, ISSUE DRIVE RESET, GO
                                                    ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)

                                                    ;"R/W/S READY" BIT SET?
                                                    ;IF NO WAIT! (IF HUNG HERE RUN STATIC)
                                                    ;DECREMENT SEEK INCOMPLETE COUNTER
                                                    ;IF 3 'SIN' ERRORS FALL THROUGH

                                                    ;***

                                                    ;;TYPE ASCIZ STRING
                                                    ;;GET OVER THE ASCIZ
                                                    ;;65$:  .ASCIZ <15><12>/3 'SIN' ERRORS OCCURRED/
                                                    ;64$:

                                                    ;;TYPE ASCIZ STRING
                                                    ;;GET OVER THE ASCIZ
```

```
                                        ;;67$:  .ASCIZ  <15><12>/5 ERRORS OCCURRED/
                                        66$:
006322                                  2$:
006322
006322  104401  006330                          TYPE    ,69$            ;;TYPE ASCIZ STRING
006326  000422                          BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  / DRIVE DECLARED DOWN!! NOT TESTED !/
006374                                  68$:
006374  105737  001316                          TSTB    @#COMND         ;TEST THE COMMAND
006400  100006                          BPL     3$              ;IF WRITE, BRANCH
006402  062706  000004                  ADD     #4,SP           ;POINT TO SAVE R0
006406  012600                          MOV     (SP)+,R0        ;GET R0 BACK
006410  052710  100000                  BIS     #BIT15,(R0)     ;SET THE DOWN BIT
006414  000207                          RTS     PC              ;RETURN TO MAIN CODE
006416  052710  100000          3$:     BIS     #BIT15,(R0)     ;SET THE DOWN BIT
006422  012706  001100                  MOV     #STACK,SP       ;RESTORE THE STACK
006426  000137  003310                  JMP     @#G01
006432  000207                  RETRN2: RTS     PC

                                        ;THIS ROUTINE CHECKS A SECTORS WORTH OF DATA ON A READ OPERATION
                                        ;IT ALLOWS 5 ERROR PRINTOUTS PER SECTOR

006434  010446                  RDCHK:  MOV     R4,-(SP)        ;SAVE   R4
006436  010546                          MOV     R5,-(SP)        ;SAVE R5 FOR RDLINK
006440  105037  001320                  CLRB    @#HDRFLG        ;CLEAR THE PRINT HEADER FLAG
006444  000240                          NOP                     ;***
006446  012737  000005  001350          MOV     #5,@#CHKCNT     ;PUT ERROR COUNT IN CHECK COUNT
006454  012704  007342                  MOV     #RDBUFF,R4      ;GET THE TABLE ADDRESS TO R4
006460  013705  001362                  MOV     @#PATTRN,R5     ;GET THE EXPECTED DATA TO R5
006464  020524                  1$:     CMP     R5,(R4)+        ;ARE THEY THE SAME
006466  001515                          BEQ     3$              ;IF YES BRANCH
006470  105737  001320                  TSTB    @#HDRFLG        ;IS THE HEADER FLAG CLEAR
006474  001046                          BNE     2$              ;IF NO BRANCH
006476  104401  006504                  TYPE    ,65$            ;;TYPE ASCIZ STRING
006502  000420                          BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <15><12>/ERROR! DATA WRITTEN BY DRIVE /
006544                                  64$:
006544  152737  000377  001320          BISB    #377,@#HDRFLG   ;SET THE HEADER FLAG
006552  113746  001317                  MOVB    @#WRTNBY,-(SP)  ;PICK UP THE DRIVE # THAT WROTE
006556  104403                          TYPOS
006560     001                          .BYTE   1
006561     000                          .BYTE   0
006562  104401  006570                  TYPE    ,67$            ;;TYPE ASCIZ STRING
006566  000411                          BR      66$             ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  / CANNOT BE READ./
006612                                  66$:
006612                                  2$:
006612  104401  006620                  TYPE    ,69$            ;;TYPE ASCIZ STRING
006616  000405                          BR      68$             ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <15><12>/  ADDR=/
006632                                  68$:
006632  013746  001334                  MOV     @#DSKADR,-(SP)  ;PICK UP THE ADDRESS THAT FAILED
006636  104403                          TYPOS
006640     006                          .BYTE   6
006641     001                          .BYTE   1
006642  104401  006650                  TYPE    ,71$            ;;TYPE ASCIZ STRING
006646  000405                          BR      70$             ;;GET OVER THE ASCIZ
```

```
1042
1043  006662                          ;;71$:  .ASCIZ  /  EXPCTD=/
1044  006662  010546                  70$:    MOV     R5,-(SP)        ;PICK UP THE EXPECTED DATA (GOOD)
1045  006664  104404                          TYPON
1046  006666  104401  006674                  TYPE    ,73$            ;;TYPE ASCIZ STRING
1047  006672  000405                          BR      72$             ;;GET OVER THE ASCIZ
1048                                  ;;73$:  .ASCIZ  /  RECVD=/
1049  006706                          72$:
1050  006706  016446  177776                  MOV     -2(R4),-(SP)    ;PICK UP THE RECEIVED DATA (BAD)
1051  006712  104404                          TYPON
1052  006714  005337  001350                  DEC     @#CHKCNT        ;DECREMENT THE CHECK COUNT
1053  006720  001403                          BEQ     4$              ;IF ZERO, BRANCH
1054  006722  022704  010342          3$:     CMP     #MANSEL,R4      ;DONE ALL CHECKS?
1055  006726  001256                          BNE     1$              ;IF NO, GO BACK
1056  006730  012605                  4$:     MOV     (SP)+,R5        ;RESTORE R5
1057  006732  012604                          MOV     (SP)+,R4        ;RESTORE R4
1058  006734  000207                          RTS     PC              ;RETURN TO CALLER
1059
1060                                  ;THIS ROUTINE BUILDS A TABLE OF PARAMETERS TO PASS TO THE SECOND SYSTEM
1061                                  ;IT PACKS THE INFO FOR TWO DRIVES INTO ONE WORD
1062
1063  006736  005003                  SECONE: CLR     R3              ;CLEAR THE WORD COUNTER
1064  006740  000240                          NOP                     ;***
1065  006742  012702  001256                  MOV     #MSKTBL,R2
1066  006746  005042                  6$:     CLR     -(R2)
1067  006750  022702  001246                  CMP     #PASTBL,R2
1068  006754  001374                          BNE     6$
1069  006756  012700  001166                  MOV     #LOGA,R0        ;GET THE ACTIVE TABLE ADDRESS
1070  006762  012001                  1$:     MOV     (R0)+,R1        ;PICK UP THE WORD
1071  006764  000301                          SWAB    R1              ;GET THE DRIVE # TO THE LOW BYTE
1072  006766  042701  177400                  BIC     #177400,R1      ;CLEAR THE UNWANTED BITS
1073  006772  106101                          ROLB    R1              ;ROTATE THE BYTE, WAS DOWN SET?
1074  006774  103002                          BCC     2$              ;IF NO BRANCH
1075  006776  052701  000001                  BIS     #BIT0,R1        ;SHOW THE DRIVE AS DOWN IN THE TABLE
1076  007002  105701                  2$:     TSTB    R1              ;IS THIS THE SYSTEM #2 DRIVE?
1077  007004  100404                          BMI     3$              ;BRANCH IF YES
1078  007006  142701  000360                  BICB    #360,R1         ;CLEAR THE UNUSED BITS
1079  007012  110122                          MOVB    R1,(R2)+        ;GET THIS # TO THE PASS TABLE
1080  007014  000762                          BR      1$              ;GET THE NEXT WORD FROM ACTIVE TABLE
1081  007016  110112                  3$:     MOVB    R1,(R2)         ;GET THE LAST DRIVE TO THE PASS TABLE
1082  007020  012702  001246                  MOV     #PASTBL,R2      ;RESTORE THE TABLE POINTER
1083  007024  104401  007032                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1084  007030  000425                          BR      64$             ;;GET OVER THE ASCIZ
1085                                  ;;65$:  .ASCIZ  <15><12>/LOAD AND START ADDRESS 210 ON SYSTEM #2/
1086  007104                          64$:
1087  007104  104401  007112                  TYPE    ,67$            ;;TYPE ASCIZ STRING
1088  007110  000424                          BR      66$             ;;GET OVER THE ASCIZ
1089                                  ;;67$:  .ASCIZ  <15><12>/AND TYPE THE BELOW WHEN ASKED FOR IT./
1090  007162                          66$:
1091  007162  005203                  4$:     INC     R3              ;INCREMENT THE WORD COUNTER
1092  007164  104401  007172                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1093  007170  000405                          BR      68$             ;;GET OVER THE ASCIZ
1094                                  ;;69$:  .ASCIZ  <15><12>/WORD  /
1095  007204                          68$:
1096  007204  010346                          MOV     R3,-(SP)        ;GET THE WORD COUNT ON THE STACK
1097  007206  104403                          TYPOS
```

```
1098  007210  006                             .BYTE   6
1099  007211  000                             .BYTE   0
1100  007212  104401  007220                  TYPE    ,71$            ;;TYPE ASCIZ STRING
1101  007216  000401                          BR      70$             ;;GET OVER THE ASCIZ
1102                                  ;;71$:  .ASCIZ  /=/
1103  007222                          70$:
1104  007222  012246                          MOV     (R2)+,-(SP)     ;GET THE FIRST TO THE STACK
1105  007224  104403                          TYPOS
1106  007226  006                             .BYTE   6
1107  007227  001                             .BYTE   1
1108  007230  032762  100000  177776          BIT     #BIT15,-2(R2)   ;WAS THIS THE TABLE TERMINATOR
1109  007236  001004                          BNE     5$              ;BRANCH IF YES
1110  007240  032762  000200  177776          BIT     #BIT7,-2(R2)    ;TERMINATOR?
1111  007246  001745                          BEQ     4$              ;IF NO BRANCH
1112  007250  005740                  5$:     TST     -(R0)
1113  007252  000000                          HALT
1114
1115  007254                          RETFR2:
1116  007254  104401  007262                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1117  007260  000404                          BR      64$             ;;GET OVER THE ASCIZ
1118                                  ;;65$:  .ASCIZ  <15><12>/WORD=/
1119  007272                          64$:
1120  007272  104411                          RDOCT
1121  007274  012602                          MOV     (SP)+,R2        ;GET THE WORD FROM SYSTEM 2 TO TABLE
1122  007276  042702  177400                  BIC     #177400,R2
1123  007302  012704  001166                  MOV     #LOGA,R4        ;SET POINTER LOOK FOR FIRST "UP" DRIVE
1124  007306  005724                  1$:     TST     (R4)+           ;DRIVE UP?
1125  007310  100776                          BMI     1$              ;IF NO BRANCH
1126  007312  010437  001100                  MOV     R4,@#$PASS
1127  007316  014401                          MOV     -(R4),R1
1128  007320  000240                          NOP                     ;***
1129  007322  004737  004210                  JSR     PC,LDFLG        ;CALL DO2+
1130  007326  004737  005616                  JSR     PC,RDLINK       ;CALL READ CHECK
1131  007332  013700  001100                  MOV     @#$PASS,R0
1132  007336  000137  003356                  JMP     @#EXTFR2        ;GO TO END OF TEST
1133
1134  007342  000400                  RDBUFF: .BLKW   400
1135
1136  010342  000240                  MANSEL: NOP                     ;TABLE TERMINATOR
1137
1138
1139
1140
1141
1142  010344                          BADONE:
1143  010344  104401  010352                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1144  010350  000401                          BR      64$             ;;GET OVER THE ASCIZ
1145                                  ;;65$:  .ASCIZ  /?/
1146  010354                          64$:
1147
1148
1149                                          .SBTTL  OSCILLATING SEEK ROUTINE
1150
1151  010354                          SECT.2:
1152  010354  104401  010362                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1153  010360  000416                          BR      64$             ;;GET OVER THE ASCIZ
```

```
1154                                        ;;65$:  .ASCIZ  <15><12>/OSCILLATING SEEK PACKAGE/
1155  010416                                64$:
1156
1157  010416  012700  020474                       MOV     #DRIV0,R0       ;FIND OUT WHICH DRIVES ARE
1158  010422  005001                               CLR     R1              ;PRESENT AND PUT THE
1159  010424  010177  170746               1$:     MOV     R1,@RKDA        ;DRIVE #'S IN A TABLE STARTING
1160  010430  105777  170730                       TSTB    @RKDS           ;AT 'DRIV0'. BITS 15-13 CONTAINS
1161  010434  100001                               BPL     2$              ;THE DRIVE #
1162  010436  010120                               MOV     R1,(R0)+
1163  010440  062701  020000               2$:     ADD     #20000,R1
1164  010444  001367                               BNE     1$
1165  010446  012710  177777                       MOV     #-1,(R0)        ;SET THE TERMINATOR TO THE TABLE
1166
1167  010452  013702  001376               INIT.2: MOV     RKDA,R2
1168  010456  012777  000001  170704               MOV     #1,@RKCS        ;ISSUE CONTROL RESET + GO !
1169  010464  004737  005600                       JSR     PC,$MTME
1170  010470  105777  170674               1$:     TSTB    @RKCS           ;DID CONTROL READY SET?
1171  010474  100375                               BPL     1$              ;IF NO WAIT! (IF HUNG RUN STATIC)
1172  010476  012700  020474                       MOV     #DRIV0,R0
1173  010502  012077  170670               3$:     MOV     (R0)+,@RKDA
1174  010506  012777  000015  170654               MOV     #15,@RKCS       ;ISSUE DRIVE RESET + GO!
1175  010514  004737  005600                       JSR     PC,$MTME
1176  010520  105777  170644               2$:     TSTB    @RKCS
1177  010524  100375                               BPL     2$
1178  010526  022710  177777                       CMP     #-1,(R0)
1179  010532  001363                               BNE     3$
1180  010534  104401  010542                       TYPE    ,65$            ;;TYPE ASCIZ STRING
1181  010540  000432                               BR      64$             ;;GET OVER THE ASCIZ
1182                                        ;;65$:  .ASCIZ  <15><12>/SET SW0 TO SW7 TO SELECT DRIVES FOR TEST AND CONT/
1183  010626                                64$:
1184  010626  104401  010634                       TYPE    ,67$            ;;TYPE ASCIZ STRING
1185  010632  000426                               BR      66$             ;;GET OVER THE ASCIZ
1186                                        ;;67$:  .ASCIZ  <15><12>/RESET SW0 TO SW7 TO TEST ALL AVAIL DRIVES/
1187  010710                                66$:
1188  010710  022737  000176  001140       7$:     CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER IN USE ?
1189  010716  001002                               BNE     9$              ;BR IF NOT
1190  010720  104405                               GTSWR                   ;REQUEST NEW CONTENTS FOR SWITCH REG
1191  010722  000401                               BR      8$              ;CONTINUE
1192  010724  000000                       9$:     HALT                    ;WAIT FOR OPERATOR TO ENTER NEW SWR VALUE
1193  010726  117704  170206               8$:     MOVB    @SWR,R4         ;SW0 TO SW7 TO R4
1194  010732  001413                               BEQ     4$              ;NONE SET, SO TEST ALL
1195  010734  012700  020474                       MOV     #DRIV0,R0       ;TABLE TO STORE DRIVE ADDRS
1196  010740  005001                               CLR     R1              ;ADDR OF DRIVE
1197  010742  006004                       6$:     ROR     R4              ;NEXT SWITCH TO CARRY
1198  010744  103001                               BCC     5$              ;SWITCH NOT SET
1199  010746  010120                               MOV     R1,(R0)+        ;SWITCH SET, SO MOVE ADDR TO TABLE
1200  010750  062701  020000               5$:     ADD     #20000,R1       ;ADDR OF NEXT DRIVE
1201  010754  001372                               BNE     6$              ;ALL DONE WHEN ZERO
1202  010756  012720  177777                       MOV     #-1,(R0)+       ;TABLE TERMINATOR
1203  010762  105737  001326               4$:     TSTB    OSPFLG          ;TYPED ONCE?
1204  010766  001165                               BNE     RWSRDY          ;IF YES, DONT RETYPE
1205  010770  104401  010776                       TYPE    ,69$            ;;TYPE ASCIZ STRING
1206  010774  000432                               BR      68$             ;;GET OVER THE ASCIZ
1207                                        ;;69$:  .ASCIZ  <15><12>/TOGGLE THE "FIRST CYLINDER ADDRESS" (OUTER LIMIT)/
1208  011062                                68$:
1209  011062  104401  011070                       TYPE    ,71$            ;;TYPE ASCIZ STRING
```

```
1210  011066  000441                          BR      70$             ;;GET OVER THE ASCIZ
1211                                ;;71$:    .ASCIZ  <15><12>/INTO THE LOW BYTE (BIT0-7) OF THE SWITCH REGISTER AND THE "LAST
1212  011172                        70$:
1213  011172  104401  011200                  TYPE    ,73$            ;;TYPE ASCIZ STRING
1214  011176  000430                          BR      72$             ;;GET OVER THE ASCIZ
1215                                ;;73$:    .ASCIZ  <15><12>/CYLINDER ADDRESS" (INNER LIMIT) INTO THE HIGH/
1216  011260                        72$:
1217  011260  104401  011266                  TYPE    ,75$            ;;TYPE ASCIZ STRING
1218  011264  000424                          BR      74$             ;;GET OVER THE ASCIZ
1219                                ;;75$:    .ASCIZ  <15><12>/       BYTE (BIT8-15), THEN PRESS CONTINUE./
1220  011336                        74$:
1221  011336  105237  001326                  INCB    OSPFLG
1222  011342  032777  000100  170014  RWSRDY: BIT     #100,@RKDS      ;"READ/WRITE/SEEK READY" BIT SET?
1223  011350  001774                          BEQ     RWSRDY          ;IF NO WAIT! (IF HUNG RUN STATIC)
1224  011352  022737  000176  001140  TRYAGN: CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER  IN USE ?
1225  011360  001002                          BNE     1$              ;BR IF NOT
1226  011362  104405                          GTSWR                   ;GET NEW CONTENTS
1227  011364  000401                          BR      CONTIN          ;CONTINUE
1228  011366  000000                  1$:     HALT
1229  011370  012777  000001  167772  CONTIN: MOV     #1,@RKCS        ;CONTROL RESET
1230  011376  105777  167766                  TSTB    @RKCS
1231  011402  100375                          BPL     .-4
1232  011404  013705  001140                  MOV     SWR,R5          ;GET THE ADDRESS OF THE SWITCH REG. TO R5
1233  011410  112501                  1$:     MOVB    (R5)+,R1        ;GET A BYTE TO R1
1234  011412  042701  177400                  BIC     #177400,R1      ;CLEAR THE UNUSED BITS
1235  011416  022701  000312                  CMP     #312,R1         ;IS ADDRESS LEGAL?
1236  011422  100034                          BPL     2$              ;BRANCH IF YES
1237  011424  104401  011432                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1238  011430  000430                          BR      64$             ;;GET OVER THE ASCIZ
1239                                ;;65$:    .ASCIZ  <15><12>/INVALID ADDRESS IN SWITCH REGISTER TRY AGAIN/
1240  011512                        64$:
1241  011512  000717                          BR      TRYAGN          ;GO BACK FOR NEW PARAMETERS
1242  011514  006101                  2$:     ROL     R1              ;ROTATING THIS REGISTER
1243  011516  006101                          ROL     R1              ;MAKES THE BYTE FROM THE
1244  011520  006101                          ROL     R1              ;SWITCH REGISTER LINE UP
1245  011522  006101                          ROL     R1              ;WITH THE CYLINDER BITS OF
1246  011524  006101                          ROL     R1              ;THE RKDA REGISTER.
1247  011526  042701  160037                  BIC     #160037,R1      ;CLEAR THE UNUSED BITS
1248  011532  032705  000001                  BIT     #1,R5           ;IS THIS THE LOW BYTE?
1249  011536  001003                          BNE     3$              ;BRANCH IF YES
1250  011540  010137  001402                  MOV     R1,@#SEEKI      ;STORE THE INNER LIMIT OF THE SEEK
1251  011544  000403                          BR      SEKSET          ;START SEEKS
1252  011546  010137  001404          3$:     MOV     R1,@#SEEKO      ;STORE THE OUTER LIMIT OF THE SEEK
1253  011552  000716                          BR      1$              ;GO BACK AND GET HIGH BYTE.
1254
1255  011554  012703  000050          SEKSET: MOV     #50,R3          ;GET THE NUMBER OF SEEK CYCLES TO R3
1256  011560  013704  001404                  MOV     @#SEEKO,R4      ;ADD THE OUTER LIMIT CYLINDER ADDRESS
1257  011564  013705  001402                  MOV     @#SEEKI,R5      ;ADD THE INNER LIMIT CYLINDER ADDRESS
1258
1259  011570  012700  020474          LDSEEK: MOV     #DRIV0,R0       ;INITIALIZE POINTER
1260  011574  010477  167576          5$:     MOV     R4,@RKDA        ;GET INNER LIMIT CYL ADRES
1261  011600  052077  167572                  BIS     (R0)+,@RKDA     ;SET DRIVE ADRES
1262  011604  012777  000011  167556          MOV     #11,@RKCS       ;ISSUE SEEK+GO! (FOR INNER LIMIT)
1263  011612  004737  005600                  JSR     PC,$MTME
1264  011616  105777  167546          3$:     TSTB    @RKCS
1265  011622  100375                          BPL     3$
```

```
1266
1267  011624  022710  177777            CMP     #-1,(R0)        ;ALL DRIVES DONE?
1268  011630  001361                    BNE     5$              ;NO
1269  011632  012700  020474            MOV     #DRIV0,R0
1270  011636  012077  167534    6$:     MOV     (R0)+,@RKDA     ;ADRES THE DRIVE
1271  011642  032777  000100  167514    BIT     #RWS,@RKDS      ;SEEK DONE?
1272  011650  001774            7$:     BEQ     7$              ;NO, WAIT
1273  011652  022710  177777            CMP     #-1,(R0)        ;ALL DRIVES DONE?
1274  011656  001367                    BNE     6$              ;NO
1275
1276  011660  012700  020474            MOV     #DRIV0,R0
1277  011664  010577  167506    8$:     MOV     R5,@RKDA        ;SET OUTER CYL ADRES
1278  011670  0[illegible]  167502      BIS     (R0)+,@RKDA     ;SET DRIVE ADRES
1279  011674  012777  000011  167466    MOV     #11,@RKCS       ;ISSUE SEEK+GO! (FOR OUTER LIMIT)
1280  011702  004737  005600            JSR     PC,$MTME
1281  011706  105777  167456    9$:     TSTB    @RKCS
1282  011712  100375                    BPL     9$
1283
1284  011714  022710  177777            CMP     #-1,(R0)        ;ALL DONE?
1285  011720  001361                    BNE     8$              ;NO
1286
1287  011722  012700  020474            MOV     #DRIV0,R0
1288  011726  012077  167444    10$:    MOV     (R0)+,@RKDA     ;SET DRIVE ADRES
1289  011732  032777  000100  167424    BIT     #RDY,@RKDS      ;SEEK DONE?
1290  011740  0[illegible]1774            BEQ     11$
1291  011742  0 2710  177777            CMP     #-1,(R0)        ;ALL DONE?
1292  011746  0 1367                    BNE     10$
1293  011750  0[illegible]5303            DEC     R3              ;DONE 50 SEEK CYCLES (100 SEEKS)
1294  011752  001306                    BNE     LDSEEK          ;IF NO BRANCH (KEEP CYCLING)
1295  011754  000605                    BR      CONTIN          ;CHECK SWR FOR CHANGE AND CONTINUE
1296
1297
1298
1299
1300
1301                                    .SBTTL  FORMATTER-SURFACE VERIFIER
1302
1303  011756                    BAD.IN:
1304  011756  104401  0'1764           TYPE    ,65$            ;;TYPE ASCIZ STRING
1305  011762  000401                    BR      64$             ;;GET OVER THE ASCIZ
1306                            ;;65$:  .ASCIZ  /?/
1307  011766                    64$:
1308  011766                    SECT.1:
1309  011756  104401  011.74           TYPE    ,65$            ;;TYPE ASCIZ STRING
1310  011772  000441                    BR      64$             ;;GET OVER THE ASCIZ
1311                            ;;65$:  .ASCIZ  <15><12>/FORMATTER-SURFACE VERIFIER,SET SW REG FOR DRV #'S. PRESS CONT./
1312  012076                    64$:
1313  012076  022737  000176  001140        CMP     #SWREG,SWR      ;SOFTWARE SWITCH REGISTER IN USE ?
1314  012104  001002                    BNE     1$              ;BR IF NOT
1315  012106  104405                    GTSWR                   ;GET SWITCH REGISTER VALUE
1316  012110  [illegible]0401            BR      2$              ;CONTINUE
1317  012112  000000            1$:     HALT                    ;WAIT FOR 'CONTINUE'
1318  012114  012737  000001  020324    2$:     MOV     #1,SHFCNT       ;SET SHIFT COUNT
1319  012122  005037  020326            CLR     DRVCNT          ;CLEAR DRIVE COUNT
1320  012126  033777  020324  167004    S13:    BIT     SHFCNT,@SWR     ;IS THIS SW SET?
1321  012134  001011                    BNE     S10             ;YES FORMAT THIS DRIVE
```

```
1322  012136  006337  020324          S11:    ASL     SHFCNT
1323  012142  005237  020326                  INC     DRVCNT
1324  012146  022737  000010  020326          CMP     #10,DRVCNT      ;ALL DONE?
1325  012154  001556                          BEQ     GO1
1326  012156  000763                          BR      S13
1327  012160  013700  020326          S10:    MOV     DRVCNT,R0
1328  012164  104401  001161                  TYPE    ,$CRLF
1329  012170  104401  020330                  TYPE    ,EM1            ;TYPE 'DRIVE'
1330  012174  010046                          MOV     R0,-(SP)        ;TYPE DRIVE #
1331  012176  104402                          TYPOC
1332  012200  000300                          SWAB    R0
1333  012202  006100                          ROL     R0
1334  012204  006100                          ROL     R0
1335  012206  006100                          ROL     R0
1336  012210  006100                          ROL     R0
1337  012212  006100                          ROL     R0
1338  012214  016037  001352                  MOV     R0,@#DSKTMP
1339  012220  012737  000000  001422          MOV     #0,ERRWF
1340  012226  012737  000000  001424          MOV     #0,ERRCF        ;CLEAR OUT ERROR COUNTS.
1341  012234  012737  000000  001426          MOV     #0,ERRFC
1342  012242  012737  000000  001430          MOV     #0,ERRCH
1343  012250  012737  000000  001432          MOV     #0,ERRCS
1344  012256  012737  177750  001416  S12:    MOV     #-24.,RWC       ;SET WORD COUNT FOR READ FORMAT.
1345  012264  012737  164000  001412          MOV     #-6144.,WC      ;SET UP WORD COUNT FOR WRITES.
1346  012272  012737  000000  001420          MOV     #0,EXTR         ;CLEAR EXTRA BIT FOR 12 SECTOR PACK.
1347  012300  012701  177772          COMMON: MOV     #-6.,R1         ;SET UP LOOP COUNT FOR THE CLEANER.
1348  012304  012777  014500  167064  COM:    MOV     #14500,@RKDA    ;SET UP FOR A SEEK TO 202.
1349  012312  057777  001352  167056          BIS     @#DSKTMP,@RKDA
1350  012320  105777  167044          1$:     TSTB    @RKCS           ;IS THE CONTROLER READY?
1351  012324  100375                          BPL     1$              ;NO SO WAIT.
1352  012326  012777  000011  167034          MOV     #11,@RKCS       ;DO THE SEEK.
1353  012334  032777  000100  167022  2$:     BIT     #BIT6,@RKDS     ;IS THE SEEK DONE?
1354  012342  001774                          BEQ     2$              ;NO SO WAIT.
1355  012344  105777  167020          3$:     TSTB    @RKCS           ;IS CONTROLER READY?
1356  012350  100375                          BPL     3$              ;NO
1357  012352  012777  000015  167010          MOV     #15,@RKCS       ;DO A DRIVE RESET.
1358  012360  032777  000100  166776  4$:     BIT     #BIT6,@RKDS     ;IS DRIVE RESET DONE.
1359  012366  001774                          BEQ     4$              ;NO
1360  012370  005201                          INC     R1              ;COUNT THE CLEANER LOOP.
1361  012372  001344                          BNE     COM             ;MORE TO GO.
1362  012374  012737  000000  001410          MOV     #0,DA           ;START OUT AT CYL. 0.
1363  012402  012777  177777  166776  NEXT:   MOV     #177777,@BA     ;PUT ALL ONE'S IN BUFFER.
1364  012410  004137  012516                  JSR     R1,IO           ;GO DO THE DISK THING.
1365  012414  012777  000000  166764          MOV     #0,@BA          ;PUT ALL ZERO'S IN BUFFER.
1366  012422  004137  012516                  JSR     R1,IO           ;GO DO IT AGIAN.
1367  012426  012777  125252  166752          MOV     #125252,@BA     ;PUT A ALT. PATTERN IN BUFFER.
1368  012434  004137  012516                  JSR     R1,IO           ;ONCE MORE.
1369  012440  005177  166742                  COM     @BA             ;COMPLEMENT THE LAST PATTERN.
1370  012444  004137  012516                  JSR     R1,IO           ;AND AGAIN.
1371  012450  062737  000040  001410          ADD     #40,DA          ;INCREMENT TO THE NEXT CYL.
1372  012456  022737  014540  001410          CMP     #14540,DA       ;ARE WE DONE WITH THIS ONE?
1373  012464  001346                          BNE     NEXT            ;NO SO DO THE NEXT CYL.
1374  012466                          GOOT:
1375  012466  104401  012474                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1376  012472  000406                          BR      64$             ;;GET OVER THE ASCIZ
1377                          ;;65$:  .ASCIZ  <CR><LF>/PACK GOOD/
```

```
1378  012510                          64$:
1379  012510  000612                          BR      S11
1380  012512  000137  001440          GD1:    JMP     @#START         ;RESTART
1381                                  ;
1382                                  ;
1383                                  ;DISK I/O SUBROUTINE.
1384                                  ;
1385                                  ;
1386                                  ;SET UP FOR A WRITE/FORMAT.
1387  012516  013777  001412  166646  10:      MOV     WC,@RKWC        ;SET UP THE WORD COUNT REG.
1388  012524  013777  001410  166644          MOV     ^A;[illegible]9  ;SET UP THE DISK ADDRESS.
1389  012532  053777  001352  166636          BIS     @#[illegible]3,@RKDA  ;SET THE UNIT NUMBER UP.
1390  012540  013777  001406  166626          MOV     BA,@RKBA        ;SET UP THE BUSS ADDRESS.
1391  012546  012777  000000  166614          MOV     #0,@RKCS        ;CLEAR THE CONTROL REG. FOR SET UP.
1392  012554  052777  0[illegible]00  166606  BIS     #BIT10+BIT11,@RKCS ;SET FORMAT&INHIBT INC. BITS.
1393  012562  052777  0[illegible]02  166[illegible]00  BIS     #BIT1,@RKCS     ;SET UP WRITE FUN.
1394  012570  053777  001420  166[illegible]72  BIS     EXTR,@RKCS      ;SET UP 12OR16 SECTOR PACK.
1395  012576  052777  000001  166564          BIS     #BIT0,@RKCS     ;GO DO THE WRITE FORMAT.
1396  012604  105777  166560          1$:     TSTB    @RKCS           ;IS WRITE FORMAT DONE?
1397  012610  100375                          BPL     1$              ;NO SO WAIT.
1398  012612  005777  166552                  TST     @RKCS           ;WAS THERE A ERROR?
1399  012616  100541                          BMI     WFERR           ;YES GO SERVICE IT.
1400  012620  012737  000000  001422          MOV     #0,ERRWF        ;CLEAR OUT THE ERROR COUNTER.
1401                                  ;
1402                                  ;SET UP FOR A READ/FORMAT
1403                                  ;
1404  012626  013777  001416  166536          MOV     RWC,@RKWC       ;SET UP WORD COUNT REG.
1405  012634  013777  001410  166534          MOV     DA,@RKDA        ;SET UP DISK ADDRESS.
1406  012642  053777  001352  166526          BIS     @#DSKTMP,@RKDA  ;SET THE UNIT NUMBER .
1407  012650  013777  001414  166516          MOV     RBA,@RKBA       ;SET UP THE BUSS ADDRESS.
1408  012656  012777  000000  166504          MOV     #0,@RKCS        ;CLEAR THE CONTROL REG.
1409  012664  052777  002000  166476          BIS     #BIT10,@RKCS    ;SET THE FORMAT BIT.
1410  012672  052777  000004  166470          BIS     #BIT2,@RKCS     ;SET UP READ FUN.
1411  012700  053777  001420  166462          BIS     EXTR,@RKCS      ;SET UP 12 OR 16 SECTOR PACK.
1412  012706  052777  000001  166454          BIS     #BIT0,@RKCS     ;GO DO THE READ FORMAT.
1413  012714  105777  166450          2$:     TSTB    @RKCS           ;IS THE READ FORMAT DONE?
1414  012720  100375                          BPL     2$              ;NO SO WAIT.
1415  012722  032777  040000  166440          BIT     #BIT14,@RKCS    ;WAS THERE A ERROR?
1416  012730  001134                          BNE     RFERR           ;YES GO SERVICE IT.
1417  012732  012737  000000  001424          MOV     #0,ERRRF        ;CLEAR OUT THE ERROR COUNT.
1418                                  ;
1419                                  ;SET UP FOR A WRITE CHECK.
1420                                  ;
1421  012740  013777  001412  166424          MOV     WC,@RKWC        ;SET UP WORD COUNT REG.
1422  012746  013777  001410  166422          MOV     DA,@RKDA        ;SET UP DISK ADDRESS.
1423  012754  053777  001352  166414          BIS     @#DSKTMP,@RKDA  ;SET UP THE UNIT NUMBER
1424  012762  013777  001406  166404          MOV     BA,@RKBA        ;SET UP BUSS ADDRESS.
1425  012770  012777  000000  166372          MOV     #0,@RKCS        ;CLEAR THE CONTROL REG.
1426  012776  052777  004400  166364          BIS     #BIT11+BIT8,@RKCS ;SET INHIBIT INCR.&STOP ON SOFT ERROR BITS.
1427  013004  053777  001420  166356          BIS     EXTR,@RKCS      ;SET 12 OR 16 SECTOR PACK.
1428  013012  052777  000006  166350          BIS     #BIT1+BIT2,@RKCS ;SET UP WRITE CHECK FUN.
1429  013020  052777  000001  166342          BIS     #BIT0,@RKCS     ;GO DO THE WRITE CHECK.
1430                                  ;
1431                                  ;CHECK HEADERS READ BY THE READ/FORMAT.
1432                                  ;
1433  013026  013703  001416                  MOV     RWC,R3          ;PUT NUMBER OF WORDS TO
```

```
1434  013032  005403                          NEG     R3              ;CHECK IN REG 3.
1435  013034  063703  001414                  ADD     RBA,R3          ;SET REG 3 TO THE LAST WORD TO BE CHECKED.
1436  013040  013702  001414                  MOV     RBA,R2          ;SET REG 2 TO STARTING ADD. OF BUFF.
1437  013044  023722  001410          MORE:   CMP     DA,(2)+         ;CHECK THAT HEADER IS RIGHT.
1438  013050  001073                          BNE     RFCERR          ;THIS HEADER WAS WRONG GO SERVICE IT.
1439  013052  020302                          CMP     R3,R2           ;ARE WE DONE?
1440  013054  001373                          BNE     MORE            ;NO GO CHECK THE NEXT ONE.
1441  013056  012737  000000  001426          MOV     #0,ERRRFC       ;CLEAR OUT THE ERROR COUNT.
1442                                  ;
1443                                  ;LETS CHECK ON THE WRITE CHECK WE STARTED.
1444                                  ;
1445  013064  105777  166300          1$:     TSTB    @RKCS
1446  013070  100375                          BPL     1$              ;THE CONTROLER IS STILL BUSY.
1447  013072  005777  166272                  TST     @RKCS           ;WAS THERE A ERROR?
1448  013076  100407                          BMI     WCERRR          ;YES GO SERVICE IT.
1449  013100  012737  000000  001430          MOV     #0,ERRWCH       ;CLEAR OUT THE
1450  013106  012737  000000  001432          MOV     #0,ERRWCS       ;ERROR COUNTERS.
1451  013114  000201                          RTS     R1              ;RETURN TO THE MAIN LINE.
1452  013116  000137  013602          WCERRR: JMP     WCERR
1453                                  ;
1454                                  ;ERRORS FOR WRITE FORMAT.
1455                                  ;
1456  013122  005237  001422          WFERR:  INC     ERRWF           ;ADD ONE TO THE ERROR COUNT.
1457  013126  022737  000004  001422          CMP     #4,ERRWF        ;HAS IT HAPPEND 4 TIMES ON THIS CYL.
1458  013134  001016                          BNE     RETRY           ;NO.
1459  013136                          SYSER:
1460  013136  104401  013144                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1461  013142  000410                          BR      64$             ;;GET OVER THE ASCIZ
1462                                  ;;65$:  .ASCIZ  <15><12>/SYSTEM ERROR/
1463  013164                          64$:
1464  013164  000000                          HALT                    ;LET THE TECH. BREATH.
1465  013166  000137  001440                  JMP     START           ;RESTART THE TEST.
1466  013172  012777  000000  166170  RETRY:  MOV     #0,@RKCS        ;CLEAR OUT THE CONTROL REG.
1467  013200  012777  000015  166162          MOV     #15,@RKCS       ;DO A DRIVE RESET.
1468  013206  032777  000100  166150  1$:     BIT     #BIT6,@RKDS     ;IS IT DONE.
1469  013214  001774                          BEQ     1$              ;NO SO WAIT.
1470  013216  000137  012516                  JMP     IO              ;TRY AGAIN.
1471                                  ;
1472                                  ;ERRORS FOR READ/FORMAT.
1473                                  ;
1474  013222  005237  001424          RFERR:  INC     ERRRF           ;ADDONE TO ERROR COUNT.
1475  013226  022737  000004  001424          CMP     #4,ERRRF        ;HAS IT HAPPEND 4 TIMES ON THIS CYL?
1476  013234  001356                          BNE     RETRY           ;NO DO IT AGAIN.
1477  013236  000737                          BR      SYSER           ;YES SO TELL HIM SO.
1478                                  ;
1479                                  ;READ/FORMAT ERRORS FOUND BY SOFTWARE CHECKS.
1480                                  ;
1481  013240  005237  001426          RFCERR: INC     ERRRFC          ;ADD ONE TO ERROR COUNT.
1482  013244  105777  166120          1$:     TSTB    @RKCS           ;WAIT FOR THE WRITE CHECK.
1483  013250  100375                          BPL     1$
1484  013252  022737  000004  001426          CMP     #4,ERRRFC       ;IS IT 4?
1485  013260  001401                          BEQ     FAILED          ;PUT OUT FAILED MESSAGE.
1486  013262  000743                          BR      RETRY           ;NO SO GO TRY IT AGAIN.
1487  013264  042777  000037  166104  FAILED: BIC     #37,@RKDA       ;PUT WHICH SECTORS HEADER
1488  013272  042702  177740                  BIC     #177740,R2
1489  013276  060277  166074                  ADD     R2,@RKDA        ;FAILED IN RKDA FOR THE MESSAGE.
```

```
1490  013302
1491  013302  104401  013310                  FAIL:
                                                      TYPE    ,65$            ;;TYPE ASCIZ STRING
1492  013306  000417                                 BR      64$             ;;GET OVER THE ASCIZ
1493                                          ;;65$:  .ASCIZ  <15><12>/PACK FAILED AT (IN OCTAL) /
1494  013346                                  64$:
1495  013346  017701  166024                         MOV     @RKDA,R1        ;GENERAT THE CYL,SECTOR,SURFACE
1496  013352  010102                                 MOV     R1,R2   ;MESSAGE FROM RKDA
1497  013354  042702  177770                         BIC     #177770,R2
1498  013360  062702  000260                         ADD     #260,R2
1499  013364  110237  013543                         MOVB    R2,SEC+1
1500  013370  004337  013570                         JSR     R3,SHF3
1501  013374  042702  177776                         BIC     #177776,R2
1502  013400  062702  000260                         ADD     #260,R2
1503  013404  110237  013542                         MOVB    R2,SEC
1504  013410  004337  013574                         JSR     R3,SHF1
1505  013414  042702  177776                         BIC     #177776,R2
1506  013420  062702  000260                         ADD     #260,R2
1507  013424  110237  013554                         MOVB    R2,SUR
1508  013430  004337  013574                         JSR     R3,SHF1
1509  013434  042702  177770                         BIC     #177770,R2
1510  013440  062702  000260                         ADD     #260,R2
1511  013444  110237  013532                         MOVB    R2,CYL+2
1512  013450  004337  013570                         JSR     R3,SHF3
1513  013454  042702  177770                         BIC     #177770,R2
1514  013460  062702  000260                         ADD     #260,R2
1515  013464  110237  013531                         MOVB    R2,CYL+1
1516  013470  004337  013570                         JSR     R3,SHF3
1517  013474  042702  177774                         BIC     #177774,R2
1518  013500  062702  000260                         ADD     #260,R2
1519  013504  110237  013530                         MOVB    R2,CYL
1520  013510  104401  013520                         TYPE    ,1$             ;TYPE OUT THE GENERATED MESSAGE
1521  013514  000137  013562                         JMP     STOP            ;BYPASS THE INLINE MESSAGE
1522  013520  005015  054503  027114  1$:     .ASCII  <15><12>/CYL.   /
1523  013526  020040
1524  013530  030060  020060          CYL:    .ASCII  /000 /
1525  013534  042523  027103  020040        .ASCII  /SEC.  /
1526  013540  030060     040     SEC:    .ASCII  /00 /
1527  013544     123  051125  027106        .ASCII  /SURF.  /
1528  013552  020040
1529  013554  020060  005015     000  SUR:    .ASCIZ  /0 /<15><12>
1530          013562                        .EVEN
1531  013562  000000                  STOP:   HALT                    ;LET OPER DO HIS THING.
1532  013564  000137  001440                  JMP     START           ;RESTART THE TEST.
1533                                          ;
1534                                  ;SHIFT SUBROUTINES.
1535                                          ;
1536  013570  006201                  SHF3:   ASR     R1              ;HERE FOR A SHIFT OF 3.
1537  013572  006201                  SHF2:   ASR     R1              ;HERE FOR A SHIFT OF 2.
1538  013574  006201                  SHF1:   ASR     R1              ;HERE FOR A SHIFT OF 1.
1539  013576  010102                          MOV     R1,R2           ;PUT RESULTES IN THE WORKING REG.
1540  013600  000203                          RTS     R3
1541                                          ;
1542                                  ;ERRORS FOR WRITE CHECK.
1543                                          ;
1544  013602  032777  040000  165560  WCERR:  BIT     #BIT14,@RKCS    ;WAS IT A HARD ERROR.
1545  013610  001010                          BNE     WCHERR          ;YES GO PROSSES IT.
```

```
1546  013612  005237  001432          INC     ERRWCS          ;ADD 1 TO THE SOFT ERROR COUNT.
1547  013616  022737  000004  001432  CMP     #4,ERRWCS       ;HAS THERE BEEN 4 OF THEM?
1548  013624  001626                  BEQ     FAIL            ;YES PUT OUT FAILED MESSAGE.
1549  013626  000137  012516          JMP     IO              ;NO SO TRY AGAIN.
1550  013632  005237  001430  WCHERR: INC     ERRWCH          ;ADD 1 TO THE HARD ERROR COUNT.
1551  013636  022737  000004  001430  CMP     #4,ERRWCH       ;HAS THERE BEEN 4 OF THEM?
1552  013644  001402                  BEQ     SYSERR          ;YES PUT OUT SYSTEM ERROR MESSAGE.
1553  013646  000137  013172          JMP     RETRY           ;NO SO TRY AGAIN.
1554  013652  000137  013136  SYSERR: JMP     SYSER
1555                          ;
1556                          ;BUFFERS.
1557                          ;
1558  013656  000000          BUFF:   .WORD   0
1559  013660  000030          RBUFF:  .BLKW   30
1560
1561
1562
1563
1564
1565                                  .SBTTL  RK05 CONTROL PANEL TEST
1566
1567  013740                  BAD.ON:
1568  013740  104401  013746          TYPE    ,65$            ;;TYPE ASCIZ STRING
1569  013744  000401                  BR      64$             ;;GET OVER THE ASCIZ
1570                          ;;65$:  .ASCIZ  /?/
1571  013750                  64$:
1572  013750                  SECT.0:
1573  013750  104401  013756          TYPE    ,65$            ;;TYPE ASCIZ STRING
1574  013754  000424                  BR      64$             ;;GET OVER THE ASCIZ
1575                          ;;65$:  .ASCIZ  <15><12>/RK05 CONTROL PANEL TEST, WHICH DRIVE?/
1576  014026                  64$:
1577  014026  104407                  RDCHR
1578  014030  011600                  MOV     (SP),R0
1579  014032  104403                  TYPOS
1580  014034     001                  .BYTE   1
1581  014035     000                  .BYTE   0
1582  014036  162700  000060          SUB     #60,R0
1583  014042  100736                  BMI     BAD.ON
1584  014044  022700  000010          CMP     #10,R0
1585  014050  003733                  BLE     BAD.ON
1586  014052  110037  001314          MOVB    R0,@#DRIVE
1587  014056  000300                  SWAB    R0
1588  014060  006100                  ROL     R0
1589  014062  006100                  ROL     R0
1590  014064  006100                  ROL     R0
1591  014066  006100                  ROL     R0
1592  014070  006100                  ROL     R0
1593  014072  010037  001352          MOV     R0,@#DSKTMP
1594  014076  104401  014104          TYPE    ,67$            ;;TYPE ASCIZ STRING
1595  014102  000414                  BR      66$             ;;GET OVER THE ASCIZ
1596                          ;;67$:  .ASCIZ  <15><12>/MOUNT PACK ON DRIVE#/
1597  014134                  66$:
1598  014134  013746  001314          MOV     DRIVE,-(SP)     ;;SAVE DRIVE FOR TYPEOUT
1599  014140  104403                  TYPOS                   ;;GO TYPE--OCTAL ASCII
1600  014142     001                  .BYTE   1               ;;TYPE 1 DIGIT(S)
1601  014143     000                  .BYTE   0               ;;SUPPRESS LEADING ZEROS
```

```
1602  014144  104401  014152                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1603  014150  000425                          BR      68$             ;;GET OVER THE ASCIZ
1604                                  ;;69$:  .ASCIZ  <15><12>/PLACE DRIVE IN RUN ;SHOULD SEE THE RUN,/
1605  014224                          68$:
1606  014224  104401  014232                  TYPE    ,71$            ;;TYPE ASCIZ STRING
1607  014230  000423                          BR      70$             ;;GET OVER THE ASCIZ
1608                                  ;;71$:  .ASCIZ  <15><12>/POWER, AND ON CYLINDER LAMPS LIGHT./
1609  014300                          70$:
1610  014300  104401  014306                  TYPE    ,73$            ;;TYPE ASCIZ STRING
1611  014304  000425                          BR      72$             ;;GET OVER THE ASCIZ
1612                                  ;;73$:  .ASCIZ  <15><12>/MAKE DRIVE WRITE ENABLE PRESS CONTINUE/
1613  014360                          72$:
1614  014360  000000                          HALT
1615  014362  004737  016140                  JSR     PC,WRROCK       ;GO WRITE 0'S, RD 0'S, CHECK
1616  014366  104401  014374                  TYPE    ,75$            ;;TYPE ASCIZ STRING
1617  014372  000430                          BR      74$             ;;GET OVER THE ASCIZ
1618                                  ;;75$:  .ASCIZ  <15><12><15><12>/WRITE PROTECT THE DRIVE THEN PRESS CONTINUE/
1619  014454                          74$:
1620  014454  000000                          HALT
1621  014456  004737  016350                  JSR     PC,WRPRO        ;GO TRY OVERWRITE
1622  014462  104401  014470                  TYPE    ,77$            ;;TYPE ASCIZ STRING
1623  014466  000426                          BR      76$             ;;GET OVER THE ASCIZ
1624                                  ;;77$:  .ASCIZ  <15><12><15><12>/CLEAR WRITE PROTECT THEN PRESS CONTINUE/
1625  014544                          76$:
1626  014544  000000                          HALT
1627  014546  004737  016140                  JSR     PC,WRROCK       ;GO WRITE 0'S, RD 0'S, CHECK
1628  014552  013777  001352  164616          MOV     @#DSKTMP,@RKDA  ;SET UP DRIVE#
1629  014560  012777  000017  164602          MOV     #17,@RKCS       ;FUNCTION WRITE PROTECT
1630  014566  004737  005600                  JSR     PC,SMTME        ;GO WAIST TIME FOR RK11-C
1631  014572  105777  164572          1$:     TSTB    @RKCS           ;IS CONTROL READY SET
1632  014576  100375                          BPL     1$              ;IF NO, BRANCH
1633  014600  004737  016350                  JSR     PC,WRPRO        ;GO TRY OVERWRITE OF IERO'S
1634  014604                          PRTTWO:
1635  014604  104401  014612                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1636  014610  000431                          BR      64$             ;;GET OVER THE ASCIZ
1637                                  ;;65$:  .ASCIZ  <15><12><15><12>/CAUTION! TRY TO OPEN THE DOOR, DO NOT FORCE:/
1638  014674                          64$:
1639  014674  104401  014702                  TYPE    ,67$            ;;TYPE ASCIZ STRING
1640  014700  000414                          BR      66$             ;;GET OVER THE ASCIZ
1641                                  ;;67$:  .ASCIZ  <15><12>/DOOR SHOULD NOT OPEN!/
1642  014732                          66$:
1643  014732  104401  014740                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1644  014736  000420                          BR      68$             ;;GET OVER THE ASCIZ
1645                                  ;;69$:  .ASCIZ  <15><12>/PRESS CONTINUE WHEN FINISHED/
1646  015000                          68$:
1647  015000  000000                          HALT
1648  015002  104401  015010                  TYPE    ,71$            ;;TYPE ASCIZ STRING
1649  015006  000426                          BR      70$             ;;GET OVER THE ASCIZ
1650                                  ;;71$:  .ASCIZ  <15><12><15><12>/PUT DRIVE IN LOAD, WAIT FOR LOAD LIGHT/
1651  015064                          70$:
1652  015064  104401  015072                  TYPE    ,73$            ;;TYPE ASCIZ STRING
1653  015070  000420                          BR      72$             ;;GET OVER THE ASCIZ
1654                                  ;;73$:  .ASCIZ  <15><12>/PRESS CONTINUE WHEN FINISHED/
1655  015132                          72$:
1656  015132  000000                          HALT
1657  015134  013777  001352  164234          MOV     @#DSKTMP,@RKDA  ;SET UP DISK ADDRESS
```

```
 1658  015142  012777  000015  164220          MOV     #15,@RKCS       ;ISSUE A DRIVE RESET
 1659  015150  004737  005600                  JSR     PC,$MTME        ;KILL TIME FOR RK11-C
 1660  015154  105777  164210          1$:     TSTB    @RKCS           ;CONTROL READY SET?
 1661  015160  100375                          BPL     1$              ;IF NO, BRANCH
 1662  015162  032777  100000  164176          BIT     #BIT15,@RKER    ;DRE SET
 1663  015170  001023                          BNE     2$              ;IF YES BRANCH
 1664  015172  104401  015200                  TYPE    ,75$            ;;TYPE ASCIZ STRING
 1665  015176  000420                          BR      74$             ;;GET OVER THE ASCIZ
 1666                                  ;;75$:  .ASCIZ  <15><12>/DRE=BIT15 OF RKER DID NOT SET/
 1667  015240                          74$:
 1668  015240  032777  140000  164122  2$:     BIT     #140000,@RKCS   ;DID HARD ERROR ON ERROR SET
 1669  015246  001015                          BNE     3$              ;BRANCH IF YES
 1670  015250  104401  015256                  TYPE    ,77$            ;;TYPE ASCIZ STRING
 1671  015254  000412                          BR      76$             ;;GET OVER THE ASCIZ
 1672                                  ;;77$:  .ASCIZ  <15><12>/ERROR DID NOT SET/
 1673  015302                          76$:
 1674  015302  012777  000001  164060  3$:     MOV     #1,@RKCS        ;ISSUE A CONTROL RESET
 1675  015310  004737  005600                  JSR     PC,$MTME        ;WAIST TIME
 1676  015314  105777  164050          4$:     TSTB    @RKCS           ;CONTROL READY SET
 1677  015320  100375                          BPL     4$              ;IF NO, BRANCH
 1678  015322  032777  100000  164036          BIT     #BIT15,@RKER    ;'DRE' CLEAR
 1679  015330  001425                          BEQ     5$              ;IF YES BRANCH
 1680  015332  104401  015340                  TYPE    ,79$            ;;TYPE ASCIZ STRING
 1681  015336  000422                          BR      78$             ;;GET OVER THE ASCIZ
 1682                                  ;;79$:  .ASCIZ  <15><12>/CONTROL RESET DID NOT CLEAR 'DRE'/
 1683  015404                          78$:
 1684  015404  032777  140000  163756  5$:     BIT     #140000,@RKCS   ;ERROR BITS CLEAR
 1685  015412  001431                          BEQ     X               ;IF YES BRANCH
 1686  015414  104401  015422                  TYPE    ,81$            ;;TYPE ASCIZ STRING
 1687  015420  000426                          BR      80$             ;;GET OVER THE ASCIZ
 1688                                  ;;81$:  .ASCIZ  <15><12>/CONTROL RESET DID NOT CLEAR 'ERROR', RKCS/
 1689  015476                          80$:
 1690  015476                          X:
 1691  015476  104401  015504                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 1692  015502  000422                          BR      64$             ;;GET OVER THE ASCIZ
 1693                                  ;;65$:  .ASCIZ  <15><12><15><12>/OPEN THE DOOR, PUT DRIVE IN RUN/
 1694  015550                          64$:
 1695  015550  104401  015556                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 1696  015554  000425                          BR      66$             ;;GET OVER THE ASCIZ
 1697                                  ;;67$:  .ASCIZ  <15><12>/CAUTION! IF RUN LIGHT ON ERROR! DEPRESS/
 1698  015630                          66$:
 1699  015630  104401  015636                  TYPE    ,69$            ;;TYPE ASCIZ STRING
 1700  015634  000426                          BR      68$             ;;GET OVER THE ASCIZ
 1701                                  ;;69$:  .ASCIZ  <15><12>/LOAD IMMEDIATELY, CONTINUE WHEN FINISHED/
 1702  015712                          68$:
 1703  015712  000000                  XX:     HALT
 1704  015714  104401  015722                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 1705  015720  000422                          BR      64$             ;;GET OVER THE ASCIZ
 1706                                  ;;65$:  .ASCIZ  <15><12><15><12>/REMOVE THE PACK, CLOSE THE DOOR/
 1707  015766                          64$:
 1708  015766  104401  015774                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 1709  015772  000423                          BR      66$             ;;GET OVER THE ASCIZ
 1710                                  ;;67$:  .ASCIZ  <15><12>/PUT DRIVE IN RUN, DRIVE SHOULD NOT/
 1711  016042                          66$:
 1712  016042  104401  016050                  TYPE    ,69$            ;;TYPE ASCIZ STRING
 1713  016046  000423                          BR      68$             ;;GET OVER THE ASCIZ
```

```
1714                                            ;;69$:  .ASCIZ  <15><12>/RUN...INTERCLOCKS HAVE BEEN CHECKED/
1715  016116                                    68$:
1716  016116  104401  016124                             TYPE    ,71$            ;;TYPE ASCIZ STRING
1717  016122  000404                             BR      70$             ;;GET OVER THE ASCIZ
1718                                            ;;71$:  .ASCIZ  <15><12>/DONE!/
1719  016134                                    70$:
1720  016134  000137  001440                     JMP     @#START
1721  016140  005037  001362                     WRROCK: CLR     @#PATTRN        ;MAKE A PATTERN OF ZERO'S
1722  016144  013777  001352  163224                     MOV     @#DSKTMP,@RKDA  ;SET UP DRIVE ADDRESS
1723  016152  012777  000001  163210                     MOV     #1,@RKCS        ;ISSUE A CONTROL RESET
1724  016160  004737  005600                     JSR     PC,SMTME
1725  016164  105777  163200                     5$:     TSTB    @RKCS           ;CONTROL READY SET
1726  016170  100375                             BPL     5$              ;IF NO BRANCH
1727  016172  013777  001352  163176                     MOV     @#DSKTMP,@RKDA  ;SET UP DRIVE ADDRESS
1728  016200  012777  001362  163166                     MOV     #PATTRN,@RKBA   ;GET BUSS ADDRESS
1729  016206  013777  001344  163156                     MOV     @#SECCNT,@RKWC  ;WORD COUNT 1 SECTOR
1730  016214  013777  001354  163146                     MOV     @#WRITCS,@RKCS  ;IBA + WRITE + GO
1731  016222  004737  005600                     JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1732  016226  105777  163136                     1$:     TSTB    @RKCS           ;CONTROL READY SET
1733  016232  100375                             BPL     1$              ;IF NO, BRANCH
1734  016234  013777  001352  163134                     MOV     @#DSKTMP,@RKDA  ;SET UP RK REGISTERS
1735  016242  012777  007342  163124                     MOV     #RDBUFF,@RKBA   ;TO READ ONE SECTOR
1736  016250  013777  001344  163114                     MOV     @#SECCNT,@RKWC  ;TO THE READ BUFFER
1737  016256  013777  001356  163104                     MOV     @#RDCS,@RKCS
1738  016264  004737  005600                     JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1739  016270  105777  163074                     2$:     TSTB    @RKCS           ;CONTROL READY SET
1740  016274  100375                             BPL     2$              ;IF NO, BRANCH
1741  016276  012704  007342                     MOV     #RDBUFF,R4      ;GET BUFFER TO R4
1742  016302  005005                             CLR     R5              ;SET UP TO COMPARE
1743  016304  020524                     3$:     CMP     R5,(R4)+        ;FOR ZERO'S
1744  016306  001414                             BEQ     4$              ;IF OK, BRANCH
1745  016310  104401  016316                     TYPE    ,65$            ;;TYPE ASCIZ STRING
1746  016314  000410                             BR      64$             ;;GET OVER THE ASCIZ
1747                                            ;;65$:  .ASCIZ  <15><12>/WRITE FAILED/
1748  016336                                    64$:
1749  016336  000700                             BR      WRROCK          ;GO BACK TRY AGAIN
1750  016340  022704  010342                     4$:     CMP     #MANSEL,R4      ;DONE ALL CHECKS
1751  016344  001357                             BNE     3$              ;IF NO, BRANCH
1752  016346  000207                             RTS     PC              ;IF YES, RETURN
1753  016350  032777  000040  163006             WRPRO:  BIT     #BIT5,@RKDS     ;BIT 5 ON
1754  016356  001021                             BNE     1$              ;IF YES, BRANCH
1755  016360  104401  016366                     TYPE    ,65$            ;;TYPE ASCIZ STRING
1756  016364  000416                             BR      64$             ;;GET OVER THE ASCIZ
1757                                            ;;65$:  .ASCIZ  <15><12>/WPS=BIT5 OF RKDS NOT SET/
1758  016422                                    64$:
1759  016422  012737  177777  001362             1$:     MOV     #177777,@#PATTRN ;GO LOAD ALL
1760  016430  013777  001352  162740                     MOV     @#DSKTMP,@RKDA  ;RK REGISTERS
1761  016436  012777  001362  162730                     MOV     #PATTRN,@RKBA   ;TO WRITE
1762  016444  013777  001344  162720                     MOV     @#SECCNT,@RKWC  ;ALL ONES (WITH WRITE LOCK)
1763  016452  013777  001354  162710                     MOV     @#WRITCS,@RKCS  ;OVER THE ZERO'S
1764  016460  004737  005600                     JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1765  016464  105777  162700                     2$:     TSTB    @RKCS           ;CONTROL READY SET?
1766  016470  100375                             BPL     2$              ;IF NO, BRANCH
1767  016472  032777  020000  162666             BIT     #BIT13,@RKER    ;WLO BIT SET
1768  016500  001032                             BNE     3$              ;IF YES BRANCH
1769  016502  104401  016510                     TYPE    ,67$            ;;TYPE ASCIZ STRING
```

```
1770  016506  000427                          BR      66$             ;;GET OVER THE ASCIZ
1771                                  ;;67$:  .ASCIZ  <15><12>/EXPECTED WLO=BIT13 OF RKER BUT DID NOT SET/
1772  016566                          66$:
1773  016566  012777  000001  162574  3$:     MOV     #1,@RKCS        ;DO A CONTROL RESET
1774  016574  004737  005600                  JSR     PC,SMTME        ;KILL TIME FOR RK11-C
1775  016600  105777  162564          4$:     TSTB    @RKCS           ;CONTROL READY SET?
1776  016604  100375                          BPL     4$              ;IF NO BRANCH
1777  016606  032777  020000  162552          BIT     #BIT13,@RKER    ;WLO BIT CLEAR
1778  016614  001431                          BEQ     RDCHK0          ;IF YES, BRANCH
1779  016616  104401  016624                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1780  016622  000426                          BR      68$             ;;GET OVER THE ASCIZ
1781                                  ;;69$:  .ASCIZ  <15><12>/CONTROL RESET DID NOT CLEAR 'WLO' OF RKER/
1782  016700                          68$:
1783  016700  013777  001352  162470  RDCHK0: MOV     @#DSKTMP,@RKDA  ;SET UP RK REGISTERS
1784  016706  012777  007342  162460          MOV     #RDBUFF,@RKBA   ;TO READ SECTOR 0,
1785  016714  013777  001344  162450          MOV     @#  CCNT,@RKWC  ;CYLINDER 0, HEAD 0
1786  016722  013777  001356  162440          MOV     @#HEADCS,@RKCS  ;TO ENSURE NO WRITE TOOK PLACE
1787  016730  004737  005600                  JSR     PC,SMTME        ;KILL TIME
1788  016734  105777  162430          3$:     TSTB    @RKCS
1789  016740  100375                          BPL     3$
1790  016742  012703  000005                  MOV     #5,R3
1791  016746  012704  007342                  MOV     #RDBUFF,R4      ;CHECK TO INSURE NO WRITE
1792  016752  005005                          CLR     R5              ;TOOK PLACE
1793  016754  020524          1$:     CMP     R5,(R4)+        ;WITH WRITE LOCK
1794  016756  001474                          BEQ     2$
1795  016760  005303                          DEC     R3              ;DEC THE ERROR COUNT
1796  016762  001475                          BEQ     4$              ;IF ZERO BRANCH
1797  016764  104401  016772                  TYPE    ,65$            ;;TYPE ASCIZ STRING
1798  016770  000422                          BR      64$             ;;GET OVER THE ASCIZ
1799                                  ;;65$:  .ASCIZ  <15><12>/WRITE OCCURRED WITH WRITE PROTECT/
1800  017036                          64$:
1801  017036  005744                          TST     -(R4)
1802  017040  104401  017046                  TYPE    ,67$            ;;TYPE ASCIZ STRING
1803  017044  000410                          BR      66$             ;;GET OVER THE ASCIZ
1804                                  ;;67$:  .ASCIZ  <15><12>/BUFFER ADDR=/
1805  017066                          66$:
1806  017066  010446                          MOV     R4,-(SP)
1807  017070  104403                          TYPOS
1808  017072     006                          .BYTE   6
1809  017073     001                          .BYTE   1
1810  017074  104401  017102                  TYPE    ,69$            ;;TYPE ASCIZ STRING
1811  017100  000406                          BR      68$             ;;GET OVER THE ASCIZ
1812                                  ;;69$:  .ASCIZ  /    EXPCTD=/
1813  017116                          68$:
1814  017116  010546                          MOV     R5,-(SP)
1815  017120  104404                          TYPON
1816  017122  104401  017130                  TYPE    ,71$            ;;TYPE ASCIZ STRING
1817  017126  000406                          BR      70$             ;;GET OVER THE ASCIZ
1818                                  ;;71$:  .ASCIZ  /    RECVD=/
1819  017144                          70$:
1820  017144  012446                          MOV     (R4)+,-(SP)
1821  017146  104404                          TYPON
1822  017150  022704  010342  2$:     CMP     #MANSEL,R4      ;FINISHED ALL CHECKS
1823  017154  001277                          BNE     1$              ;IF NO, BRANCH
1824  017156  000207          4$:     RTS     PC              ;RETURN
1825
```

```
1826
1827
1828
```

```
                                        ;THE FOLLOWING REVISION WAS MADE BY JIM KAPADIA

                                                .SBTTL  CONTROL PANEL TEST # 2

                                        ;THIS IS THE ENTRY POINT INTO CONTROL PANEL TEST #2.  ALL
                                        ;THE DRIVES THAT ARE PRESENT AND IN 'RDY' CONDITION ARE
                                        ;REPORTED (ON LINE).

017160  000240                  SECT.4: NOP
017162  012777  000001  162200          MOV     #1,@RKCS
017170  105777  162174          3$:     TSTB    @RKCS
017174  100375                          BPL     3$
017176  012700  020474                  MOV     #DRIV0,R0
017202  005001                          CLR     R1
017204  005002                          CLR     R2

017206  010210                  1$:     MOV     R2,(R0)         ;SET UP ADDRESS TABLE
017210  010277  162162                  MOV     R2,@RKDA        ;ADDRESS THE DRIVE
017214  105777  162144                  TSTB    @RKDS           ;IS IT PRESENT?
017220  100021                          BPL     2$              ;NO

017222  104401  020330                  TYPE    ,EM1            ;TYPE 'DRIVE'
017226  010146                          MOV     R1,-(SP)        ;TYPE OUT DRIVE #
017230  104402                          TYPOC
017232  104401  020341                  TYPE    ,EM2            ;TYPE 'ON LINE'
017236  052710  000300                  BIS     #BIT6+BIT7,(R0) ;SET BITS INDICATING THIS
                                                                ;DRIVE PRESENT

017242  012777  000015  162120          MOV     #15,@RKCS       ;ISSUE A DRIVE RESET
017250  004737  005600                  JSR     PC,$MTME        ;ALLOW SOME TIME
017254  032777  000100  162102  4$:     BIT     #RWS,@RKDS      ;WAIT FOR RWS RDY
017262  001774                          BEQ     4$

017264  005720                  2$:     TST     (R0)+
017266  005201                          INC     R1
017270  062702  020000                  ADD     #20000,R2       ;NXT DRIVE
017274  001344                          BNE     1$              ;ALL DONE?

017276  104401  001161                  TYPE    ,$CRLF

                                        ;THIS CODE CHECKS THE CONDITION OF 'DRY' BIT IN RKDS FOR EVERY
                                        ;DRIVE.  IF 'DRY' IS SET DRIVE IS SAID TO BE 'ON LINE', OTHERWISE IT
                                        ;IS OFFLINE.  IF THE 'DRY' BIT HAS CHANGED FROM LAST TIME, THEN
                                        ;IT IS REPORTED.  IF THERE IS NO CHANGE NOTHING IS REPORTED.

017302  012700  020474          BEGCT:  MOV     #DRIV0,R0       ;INITIALIZE POINTERS
017306  005001                          CLR     R1

017310  011077  162062          BEGCT1: MOV     (R0),@RKDA      ;ADDRESS A DRIVE
017314  042777  017777  162054          BIC     #17777,@RKDA    ;MASK OUT NON DR# BITS
017322  105777  162036                  TSTB    @RKDS           ;IS THIS DRIVE ON LINE?
017326  100044                          BPL     1$              ;NO
                                                                ;YES
017330  105710                          TSTB    (R0)            ;WAS IT 'ON LINE' LAST TIME?
017332  100454                          BMI     NXT1            ;YES, NO MESSAGE TO REPORT
017334  052710  000200                  BIS     #BIT7,(R0)      ;IT CHANGED FROM OFF LINE TO ON
```

```
                                        TYPE    ,EM1            ;LINE, REPORT MESSAGE
                                        MOV     R1,-(SP)
                                        TYPOC
                                        TYPE    ,EM2            ;TYPE 'ON LINE'
                                        BIT     #WPS,@RKDS      ;WRITE ENABLED?
                                        BEQ     2$              ;YES, OK
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/EROR,NOT WRT ENABLED/
                                64$:
                                2$:     MOV     #17,@RKCS       ;WRITE PROT THE DISK
                                3$:     TSTB    @RKCS
                                        BPL     3$
                                        BR      NXT1

                                1$:     TSTB    (R0)            ;WAS THIS DRIVE OFF LINE LAST
                                        BPL     NXT1            ;TIME?  BRNCH IF YES
                                        TYPE    ,EM1            ;IF NOT, REPORT THE CHANGE
                                        MOV     R1,-(SP)        ;TYPE DRIVE #
                                        TYPOC
                                        TYPE    ,EM3            ;TYPE 'OFF LINE'
                                        BIC     #BIT7,(R0)      ;CLEAR BIT TO INDICATE THIS
                                                                ;DRIVE 'OFF LINE'

                                ;THIS CODE CHECKS 'WPS' BIT FOR EVERY DRIVE THAT IS IN 'DRY'
                                ;CONDITION (ON LINE).  IT REPORTS ANY CHANGE IN THE CONDITION OF
                                ;THE 'WPS' BIT.  IF THERE WAS NO CHANGE FROM LAST TIME NOTHING
                                ;IS REPORTED.  AT THE TIME OF ENTRY R0 POINTS TO DRIVE FLAG.

                                NXT1:   TSTB    @RKDS           ;IS THIS DRIVE PRESENT?
                                                                ;RKDA CONTAINS THE DRV #
                                        BPL     NXT2            ;NO, SKIP CHECKING

                                        BIT     #WPS,@RKDS      ;WPS BIT SET?
                                        BNE     1$              ;YES
                                                                ;WPS BIT CLEAR
                                        BIT     #BIT2,(R0)      ;WAS IT CLR LAST TIME ALSO?
                                        BEQ     NXT2            ;YES, NOTHING TO REPORT.
                                                                ;WPS CHANGED FROM 'SET'
                                        TYPE    ,EM1            ;TO 'CLR', REPORT IT
                                        MOV     R1,-(SP)        ;TYPE DRIVE #
                                        TYPOC
                                        BIC     #BIT2,(R0)      ;INDICATE THAT 'WPS' IS CLEAR
                                        TYPE    ,EM5            ;TYPE 'WPS CLEAR'
                                        BR      NXT2

                                                                ;WPS BIT IS SET
                                1$:     BIT     #BIT2,(R0)      ;WAS IT SET LAST TIME ALSO?
                                        BNE     NXT2            ;YES, NOTHING TO REPORT.
                                                                ;WPS CHANGED, FROM 'CLR' TO
                                                                ;'SET', REPORT THIS CHANGE
                                        TYPE    ,EM1            ;TYPE 'DRIVE'
                                        MOV     R1,-(SP)        ;TYPE DRIVE #
                                        TYPOC
                                        TYPE    ,EM4            ;TYPE 'WPS SET'
                                        BIS     #BIT2,(R0)      ;SET FLAG BIT INDICATING WPS SET
```

```
1941                                    ;THIS CODE PERFORMS A SEEK FUNCTION ON A DRIVE AND CHECKS IF
1942                                    ;THE 'DPL' BIT SET AS A RESULT, (IF THE POWER WAS CUT OFF
1943                                    ;FROM THE DRIVE).  NOTE THAT ONLY THOSE DRIVES PRE
1944                                    ;CHECKED WHICH WERE FOUND TO BE PRESENT AT BEGINNING (WHEN
1945                                    ;THIS TEST WAS ENTERED).  SEEK IS DONE TO CYLINDER 1.
1946                                    ;AT THE TIME OF ENTRY R0 POINTS TO THE DRIVE FLAG.
1947
1948  017560  032710  000100          NXT2:   BIT     #BIT6,(R0)      ;WAS THIS DRIVE PRESENT AT BEGNG
1949  017564  001403                          BEQ     4$              ;NO
1950  017566  105777  161572                  TSTB    @RKDS           ;IS IT PRESENT NOW?
1951  017572  100402                          BMI     3$              ;YES
1952  017574  000137  020232          4$:     JMP     DN1DRV          ;IF NOT SKIP THIS CHECK
1953
1954  017600  052777  000040  161570  3$:     BIS     #40,@RKDA       ;RKDA ALREADY HAS THE DRV #
1955                                                                  ;SET CYL 1 ADDRESS
1956  017606  012777  000011  161554          MOV     #11,@RKCS       ;SEEK, GO
1957
1958  017614  105777  161550          1$:     TSTB    @RKCS           ;WAIT FOR CONTROL RDY?
1959  017620  100375                          BPL     1$              ;SOMETHING WRONG IF CNTAL RDY
1960                                                                  ;DOES NOT COME BACK
1961  017622  032777  010000  161534          BIT     #DPL,@RKDS      ;DPL BIT SET?
1962  017630  001414                          BEQ     2$              ;NO
1963                                                                  ;YES, DPL SET
1964  017632  032710  000001                  BIT     #BIT0,(R0)      ;WAS 'DPL' SET LAST TIME ALSO?
1965  017636  001167                          BNE     CLRDPL          ;YES, NOTHING TO REPORT.
1966                                                                  ;DPL CHANGED, GOT SET THIS
1967  017640  104401  020330                  TYPE    ,EM1            ;TIME, REPORT IT
1968  017644  010146                          MOV     R1,-(SP)
1969  017646  104402                          TYPOC
1970  017650  104401  020417                  TYPE    ,EM6            ;TYPE 'POWER LO'
1971  017654  052710  000001                  BIS     #BIT0,(R0)      ;SET FLAG BIT INDICATING THAT
1972                                                                  ;DPL SET THIS TIME
1973  017660  000556                          BR      CLRDPL
1974                                                                  ;'DPL' BIT IS CLEAR
1975  017662  032710  000001          2$:     BIT     #BIT0,(R0)      ;WAS 'DPL' CLEAR LAST TIME ALSO?
1976  017666  001410                          BEQ     WATSK           ;YES, NOTHING TO REPORT
1977                                                                  ;REPORT THAT 'DPL' BIT CHANGED,
1978  017670  104401  020330                  TYPE    ,EM1            ;FROM SET TO CLEAR
1979  017674  010146                          MOV     R1,-(SP)
1980  017676  104402                          TYPOC
1981  017700  104401  020440                  TYPE    ,EM7            ;TYPE 'POWER UP'
1982  017704  042710  000001                  BIC     #BIT0,(R0)      ;SET FLAG BIT INDICATING THAT DPL
1983                                                                  ;IS CLEAR THIS TIME
1984
1985                                    ;THIS CODE WAITS FOR THE SEEK (DONE ABOVE) TO FINISH.  WAITING
1986                                    ;TIME IS APPROX. 50 MS (FOR THE WORST CASE).  IF R/W/S RDY
1987                                    ;DOES NOT SET WITHIN 50 MS, THEN IT IS ASSUMED THAT A 'SIN'
1988                                    ;IS POSSIBLE AND THE PROGRAM WAITS FOR 1450 MS MORE, SO THAT
1989                                    ;THE 'SIN' CAN SET.  IF 'SIN' DOES NOT SET WITHIN THIS
1990                                    ;TIME AN ERROR IS REPORTED:
1991                                    ;       SIN DIDN'T OCCUR
1992                                    ;IF R/W/S RDY SETS WITHIN 50 MS THE PROGRAM PROCEEDS TO
1993                                    ;CHECK THE NEXT DRIVE.
1994
1995  017710  012705  164220          WATSK:  MOV     #-6000.,R5      ;SET COUNT TO WAIT FOR
1996                                                                  ;50 MS
```

```
 1997  017714  032777  000100  161442  1$:     BIT     #RWS,@RKDS      ;R/W/S. RDY SET?
 1998  017722  001042                          BNE     3$              ;YES
 1999  017724  005205                          INC     R5              ;WAIT
 2000  017726  001372                          BNE     1$
 2001                                                                  ;50 MS OVER, R/W/S RDY
 2002                                                                  ;DIDN'T SET. WAIT FOR
 2003                                                                  ;SIN TO SET.
 2004  017730  005004                          CLR     R4
 2005  017732  012705  177777                  MOV     #177777,R5      ;SET UP COUNT
 2006  017736  032777  001000  161420  2$:     BIT     #SIN,@RKDS      ;SIN SET?
 2007  017744  001045                          BNE     SINST           ;YES
 2008  017746  005305                          DEC     R5              ;WAIT
 2009  017750  001372                          BNE     2$
 2010  017752  005704                          TST     R4
 2011  017754  001002                          BNE     4$
 2012  017756  005204                          INC     R4
 2013  017760  000766                          BR      2$
 2014                                                                  ;1500 MS ELAPSED, BUT SIN
 2015                                                                  ;DIDN'T SET. ERROR!
 2016  017762                          4$:
 2017  017762  104401  017770                  TYPE    ,65$             ;;TYPE ASCIZ STRING
 2018  017766  000415                          BR      64$              ;;GET OVER THE ASCIZ
 2019                                  ;;65$:  .ASCIZ  <15><12>/SIN DIDN'T OCCUR, DRVE/
 2020  020022                          64$:
 2021  020022  010146                          MOV     R1,-(SP)                ;TYPE DRIVE #
 2022  020024  104402                          TYPOC
 2023  020026  000501                          BR      DN1DRV
 2024  020030  032710  000002          3$:     BIT     #BIT1,(R0)      ;DID R/W/S RDY SET LAST TIME?
 2025  020034  001476                          BEQ     DN1DRV          ;YES
 2026  020036  042710  000002                  BIC     #BIT1,(R0)      ;CLR FLAG INDICATING THAT SEEK IS OK
 2027  020042  104401  020330                  TYPE    ,EM1            ;REPORT THAT SEEK IS OK
 2028  020046  010146                          MOV     R1,-(SP)
 2029  020050  104402                          TYPOC
 2030  020052  104401  020461                  TYPE    ,EM9
 2031  020056  000465                          BR      DN1DRV
 2032
 2033                                  ;IF SIN SET, DO DRIVE RESET AND CLEAR IT
 2034
 2035  020060  032710  000002          SINST:  BIT     #BIT1,(R0)      ;DID 'SIN' SET LAST TIME ALSO?
 2036  020064  001010                          BNE     4$              ;YES, NOTHING TO REPORT
 2037  020066  052710  000002                  BIS     #BIT1,(R0)      ;SET FLAG INDICATING THAT
 2038  020072  104401  020330                  TYPE    ,EM1            ;'SIN' SET, AND REPORT THE CHANGE
 2039  020076  010146                          MOV     R1,-(SP)
 2040  020100  104402                          TYPOC
 2041  020102  104401  020453                  TYPE    ,EM8            ;TYPE 'SIN'
 2042
 2043  020106  017705  161264          4$:     MOV     @RKDA,R5        ;SAVE RKDA
 2044  020112  012777  000001  161250          MOV     #1,@RKCS        ;DO CONTROL RESET
 2045  020120  105777  161244          1$:     TSTB    @RKCS           ;WAIT FOR CONTROL RDY
 2046  020124  100375                          BPL     1$
 2047  020126  010577  161244                  MOV     R5,@RKDA
 2048  020132  012777  000015  161230          MOV     #15,@RKCS       ;DO DRIVE RESET. RKDA
 2049                                                                  ;ALREADY HAS THE DRVE #
 2050  020140  105777  161224          2$:     TSTB    @RKCS           ;WAIT FOR CNTRL RDY
 2051  020144  100375                          BPL     2$
 2052  020146  005005                          CLR     R5
```

```
2053  020150  032777  000100  161206  3$:     BIT     #RWS,@RKDS      ;R/W/S SET?
2054  020156  001025                          BNE     DN1DRV          ;YES
2055  020160  005205                          INC     R5
2056  020162  001372                          BNE     3$              ;WAIT FOR R/W/S RDY
2057                                                                  ;REPORT ERROR. R/W/S RDY CLR
2058  020164  104401  020172                  TYPE    ,65$            ;;TYPE ASCIZ STRING
2059  020170  000411                          BR      64$             ;;GET OVER THE ASCIZ
2060                                  ;;65$:  .ASCIZ  <15><12>/RWS RDY NOT SET/
2061  020214                          64$:
2062
2063  020214  000406                          BR      DN1DRV
2064
2065                                  ;IF DPL SET CLEAR THE ERROR BY DOING CONTROL RESET.
2066
2067  020216  012777  000001  161144  CLRDPL: MOV     #1,@RKCS        ;CONTROL RESET
2068  020224  105777  161140          1$:     TSTB    @RKCS           ;WAIT FOR CNTRL RDY
2069  020230  100375                          BPL     1$
2070                                  ;AT THIS STAGE THE DRIVE (# IN RKDA) HAS BEEN CHECKED
2071                                  ;FOR DRY, WPS, DPL, & SIN. THE POINTERS ARE INCREMENTED
2072                                  ;AND THE SAME CHECKS WILL BE DONE ON THE NEXT
2073                                  ;DRIVE & THEN THE NEXT ONE & SO ON. NOTE THAT
2074                                  ;THIS SUB-PROGRAM KEEPS ON CYCLING THROUGH
2075                                  ;ALL THE DRIVES. AT THE TIME OF ENTRY (HERE)
2076                                  ;R0 POINTS TO THE FLAG FOR THE DRIVE THAT WAS
2077                                  ;JUST CHECKED. BEFORE GOING ON TO THE NEXT
2078                                  ;DRIVE THE HEADS ARE BROUGHT BACK TO CYLINDER
2079                                  ;0 (FOR THE NEXT CYCLE).
2080
2081  020232  011077  161140          DN1DRV: MOV     (R0),@RKDA      ;GET DRIVE #
2082  020236  105777  161122                  TSTB    @RKDS           ;DRIVE PRESENT?
2083  020242  100017                          BPL     3$              ;NO
2084  020244  042777  017777  161124          BIC     #17777,@RKDA    ;CYL ADRES = 0
2085  020252  012777  000011  161110          MOV     #11,@RKCS       ;GO, SEEK
2086
2087  020260  105777  161104          1$:     TSTB    @RKCS           ;WAIT FOR CNTRL RDY
2088  020264  100375                          BPL     1$
2089
2090  020266  004737  005600                  JSR     PC,SMTME
2091  020272  032777  000100  161064  4$:     BIT     #RWS,@RKDS
2092  020300  001774                          BEQ     4$
2093
2094  020302  005720                  3$:     TST     (R0)+           ;INCREMENT POINTERS TO
2095  020304  005201                          INC     R1              ;NEXT DRIVE
2096  020306  020127  000010                  CMP     R1,#10          ;ALL DONE, THIS CYCLE?
2097  020312  001402                          BEQ     2$              ;YES
2098  020314  000137  017310                  JMP     BEGCT1          ;GO DO NEXT DRIVE
2099  020320  000137  017302          2$:     JMP     BEGCT           ;RESTART THE CYCLE OVER
2100                                                                  ;AGAIN
2101
2102
2103
2104  020324  000000                  SHFCNT: .WORD   0
2105  020326  000000                  DRVCNT: .WORD   0
2106                                  ;MESSAGES
2107  020330  005015  051104  053111  EM1:    .ASCIZ  <15><12>/DRIVE /
2108  020336  020105      000
```

```
2109  020341  044C  047440  020116  EM2:    .ASCIZ  /  ON LINE/
2110  020346  044514  042516  000
2111  020353  040  047440  043106  EM3:    .ASCIZ  /  OFF LINE/
2112  020360  046040  047111  000105
2113  020366  020040  051127  020124  EM4:    .ASCIZ  /  WRT PROT/
2114  020374  051120  052117  000
2115  020401  040  053440  052122  EM5:    .ASCIZ  /  WRT ENABLED/
2116  020416  042440  040516  046102
2117  020424  042105  000
2118  020427  040  042040  044522  EM6:    .ASCIZ  /  DRIVE POWER LO/
2119  020444  042526  050040  053517
2120  020452  051105  020040  000117
2121  020457  040  047520  042527  EM7:    .ASCIZ  /  POWER UP/
2122  020466  020122  050125  000
2123  020473  040  051440  047111  EM8:    .ASCIZ  /  SIN/
2124  020500  000
2125  020501  040  051440  042505  EM9:    .ASCIZ  /  SEEK OK/
2126  020506  020113  045517  000
2127
2128                          ;DRIVE FLAGS FOR CONTROL PANEL TEST #2
2129                          ;BITS 15,14,13 GIVE THE DRIVE NO (EX: 0,1,2,---)
2130                          ;BIT 7 IS SET WHEN 'DRY' BIT IS SET FOR THE DRIVE (ON LINE)
2131                          ;BIT 7 IS CLEAR WHEN 'DRY' IS CLEAR (WHEN DRIVE IS IN LOAD/OFF LINE,
2132                          ;DRIVE POWER IS CUT OFF)
2133                          ;BIT 6 IS SET IF A DRIVE IS FOUND TO BE PRESENT (DRY) AT
2134                          ;THE BEGINING. UNLIKE BIT 7 THIS BIT DOES NOT GET SET OR
2135                          ;CLEARED AS THE DRIVE CONDITIONS CHANGE. IT JUST INDICATES
2136                          ;THAT THE DRIVE IS AVAILABLE FOR CHECKING.
2137                          ;BIT 0 IS SET IF 'DPL' BIT GETS SET, IE: DRIVE POWER OFF
2138                          ;BIT 0 IS CLEARED WHEN DRIVE POWER IS ON.
2139                          ;BIT 1 IS SET WHEN SEEK INCOMPLETE 'SIN' OCCURS.
2140                          ;BIT 1 IS CLEAR WHEN SEEK IS OK.
2141                          ;BIT 2 IS SET WHEN WRT PROT IS SET FROM CONSOLE.
2142                          ;BIT 2 IS CLEARED WHEN DRIVE IS WRITE ENABLED.
2143
2144  020474                  .EVEN
2145  020474  000000  DRIV0:  .WORD   0               ;DRIVE FLAGS
2146  020476  000000  DRIV1:  .WORD   0
2147  020500  000000  DRIV2:  .WORD   0
2148  020502  000000  DRIV3:  .WORD   0
2149  020504  000000  DRIV4:  .WORD   0
2150  020506  000000  DRIV5:  .WORD   0
2151  020510  000000  DRIV6:  .WORD   0
2152  020512  000000  DRIV7:  .WORD   0
2153
2154
```

```
                                        .SBTTL  HEAD ALIGNMENT ROUTINE

                                ;HEAD ALIGNMENT ROUTINE
                                ;THIS MAINTAINANCE ROUTINE IS HELPFUL IN HEAD ALIGNMENT. UPON ENTRY
                                ;THE QUESTION - DRIVE? - IS ASKED . THE USER SHOULD REPLY WITH THE
                                ;DRIVE NUMBER THAT IS TO BE ALIGNED. IF THE DRIVE IS AN RK-05F
                                ;THE LETTER 'F' IS ADDED AS A SUFFIX. FOR SELECTING SURFACE 0
                                ;PUT SW0=0, FOR SELECTING SURFACE 1 PUT SW0=1. SET SW1 =1 TO SELECT
                                ;CYLINDER 64.  SET SW1=0 TO SELECT CYLINDER 105.
                                ;IF THE DRIVE IS AN RK-05F, CYLINDER 64 BECOMES CYLINDER 130
                                ;OF THE EVEN DRIVE, AND CYLINDER 105 BECOMES CYLINDER 5 OF THE ODD DRIVE
                                ;THE HEADS ARE PLACED ON THE SELECTED CYLINDER AND DATA IS READ
                                ;CONTINUOUSLY FROM THE CYLINDER (SECTOR 0)
                                ;  THE UPPER OR LOWER HEAD OF ANY CYLINDER CAN BE SELECTED
                                ;DYNAMICALLY, IE. THE PROGRAM DOES NOT HAVE TO BE STOPPED TO SELECT THE
                                ;UPPER OR LOWER HEAD OR CYLINDER.
                                ;IN ORDER TO SELECT ANOTHER DRIVE, PUT ANY SWITCH FROM SW2 TO SW15 UP AND
                                ;THE PROGRAM WILL AGAIN ASK THE QUESTION (DRIVE?).

020514  000240                  SECT.5: NOP
020516  104401  020524                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
020522  000426                          BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/SET SW0=0 FOR SURFACE 0, SW0=1 FOR SUR 1./
020600                          64$:
020600  104401  020606                  TYPE    ,67$                    ;;TYPE ASCIZ STRING
020604  000432                          BR      66$                     ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  <15><12>/SET SW1=0 FOR CYLINDER 105, SW1=1 FOR CYLINDER 64/
020672                          66$:
020672  104401  020700                  TYPE    ,69$                    ;;TYPE ASCIZ STRING
020676  000427                          BR      68$                     ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <15><12>/PUT ANY SW FROM 2-15 HI TO SELECT NEW DRVE/
020756                          68$:
020756  104401  021412          HDALGN: TYPE    ,EM10                   ;ASK FOR DRIVE #
020762  005037  021406                  CLR     FFLAG                   ;FLAG FOR RK-05F
020766  104410                          RDLIN                           ;GET OPR INPUT
020770  012601                          MOV     (SP)+,R1                ;ADDR OF COMMAND STRING
020772  112100                          MOVB    (R1)+,R0                ;FIRST CHAR
020774  162700  000060                  SUB     #60,R0                  ;0 TO 7
021000  002766                          BLT     HDALGN                  ;TOO SMALL
021002  022700  000067                  CMP     #67,R0                  ;MUST BE 7 OR LESS
021006  002763                          BLT     HDALGN                  ;TOO BIG
021010  000241                          CLC
021012  006000                          ROR     R0
021014  006000                          ROR     R0
021016  006000                          ROR     R0
021020  006000                          ROR     R0
021022  010037  020474                  MOV     R0,DRIV0                ;ADDRESS OF DRIVE
021026  112100                          MOVB    (R1)+,R0                ;NEXT INPUT CHAR
021030  001412                          BEQ     5$                      ;ALL DONE IF C.R.
021032  020027  000106                  CMP     R0,#'F                  ;IS IT F?
021036  001347                          BNE     HDALGN                  ;NO, SO ERROR
021040  105711                          TSTB    (R1)                    ;NEXT CHAR MUST BE C.R.
021042  001345                          BNE     HDALGN                  ;ELSE, ERROR
021044  042737  020000  020474          BIC     #BIT13,DRIV0            ;USE EVEN DRIVE IF RK-05F
021052  005237  021406                  INC     FFLAG                   ;SHOW F TYPE DRIVE
021056  013700  020474          5$:     MOV     DRIV0,R0                ;DRIVE ADDR TO R0
```

```
2211  021062  022737  000176  001140          CMP     @#SWREG,SWR             ;SOFTWARE SWITCH REGISTER IN USE?
2212  021070  001002                          BNE     15$                     ;BR IF NOT
2213  021072  104405                          GTSWR                           ;REQUEST NEW CONTENTS FOR SWITCH REG
2214  021074  000401                          BR      16$                     ;CONTINUE
2215  021076  000000                  15$:    HALT                            ;WAIT FOR OPERATOR TO ENTER NEW SWR VALUE
2216  021100  017737  160034  021410  16$:    MOV     @SWR,SWTCH              ;HOLD SWITCHES
2217  021106  042737  177775  021410          BIC     #177775,SWTCH           ;WANT SW1 ONLY
2218  021114  001005                          BNE     7$                      ;SW1 SET, SO LOW CYLINDER
2219  021116  005737  021406                  TST     FFLAG                   ;F DRIVE?
2220  021122  001402                          BEQ     7$                      ;NO
2221  021124  052700  020000                  BIS     #BIT13,R0               ;ODD DRIVE IF HIGH TRACK OF F
2222  021130  010077  160242          7$:     MOV     R0,@RKDA                ;ADDRESS DRIVE
2223  021134  012777  000017  160226          MOV     #17,@RKCS               ;WRITE PROTECT
2224  021142  105777  160222          8$:     TSTB    @RKCS
2225  021146  100375                          BPL     8$                      ;WAIT FOR DRIVE READY
2226  021150  012777  000001  160212          MOV     #1,@RKCS                ;RESET CONTROLLER
2227  021156  105777  160206          9$:     TSTB    @RKCS
2228  021162  100375                          BPL     9$                      ;WAIT FOR READY
2229  021164  005737  021406                  TST     FFLAG                   ;F DRIVE?
2230  021170  001410                          BEQ     13$                     ;NO
2231  021172  012701  000240                  MOV     #5.*40,R1               ;TRACK 5 OF HIGH
2232  021176  005737  021410                  TST     SWTCH                   ;SW1 SET?
2233  021202  001412                          BEQ     10$                     ;YES SO TEST TRACK 8 OF DRIVE HIGH
2234  021204  062701  010100                  ADD     #130.*40,R1             ;TRACK 130. IF SW1 SET
2235  021210  000407                          BR      10$
2236  021212  012701  004000          13$:    MOV     #64.*40,R1              ;CYLINDER 64 IF NOT F
2237  021216  005737  021410                  TST     SWTCH                   ;SW1 SET?
2238  021222  001002                          BNE     10$                     ;YES, SO CYLINDER 64
2239  021224  062701  002440                  ADD     #41.*40,R1              ;CYLINDER 105
2240  021230  005777  160134          10$:    TST     @RKCS                   ;ANY ERROR?
2241  021234  100006                          BPL     11$                     ;NO, CONTINUE
2242  021236  012777  000001  160124          MOV     #1,@RKCS                ;RESET
2243  021244  105777  160120          12$:    TSTB    @RKCS
2244  021250  100375                          BPL     12$                     ;WAIT FOR READY
2245  021252  017702  157662          11$:    MOV     @SWR,R2                 ;SWITCH REG TO R2
2246  021256  042702  177775                  BIC     #177775,R2              ;SW1 ONLY
2247  021262  020237  021410                  CMP     R2,SWTCH                ;ANY CHANGE SINCE LAST?
2248  021266  001273                          BNE     5$                      ;YES, GO SET-UP ADDR AGAIN
2249  021270  010077  160102          6$:     MOV     R0,@RKDA                ;ADDRESS THE DRIVE
2250  021274  012777  000017  160066          MOV     #17,@RKCS               ;WRITE  PROTECT THE DRIVE
2251  021302  105777  160062                  TSTB    @RKCS                   ;WAIT FOR CONTROL RDY
2252  021306  100375                          BPL     .-4
2253  021310  042700  000020          4$:     BIC     #20,R0                  ;CLEAR TRACK ADDR
2254  021314  032777  000001  157616          BIT     #1,@SWR                 ;SW0 SET?
2255  021322  001402                          BEQ     2$                      ;NO TEST TRACK 0
2256  021324  052700  000020                  BIS     #20,R0                  ;TEST TRACK 1
2257  021330  042700  017740          2$:     BIC     #17740,R0               ;CLEAR CYLINDER ADDR
2258  021334  050100                          BIS     R1,R0                   ;PUT CYLINDER ADDR IN ADDR
2259  021336  010077  160034                  MOV     R0,@RKDA                ;ADRES THE DRIVE
2260  021342  012777  177400  160022          MOV     #-400,@RKWC             ;READ 1 SECTOR
2261  021350  012777  007342  160016          MOV     #RDBUFF,@RKBA           ;INTO THIS BUFFER
2262  021356  012777  000005  160004          MOV     #5,@RKCS                ;READ, GO
2263
2264  021364  105777  160000          3$:     TSTB    @RKCS                   ;DONE?
2265  021370  100375                          BPL     3$                      ;NO
2266
```

```
2267  021372  032777  177774  157540          BIT     #177774,@SWR            ;EXIT OUT?
2268  021400  001713                          BEQ     10$                     ;NO, CONTINUE ON THIS DRIVE
2269  021402  000137  020756                  JMP     HOALGN                  ;YES, GET NEW DRIVE
2270
2271  021406  000000                  FFLAG:  0
2272  021410  000000                  SWTCH:  0
2273  021412  005015  051104  053111  EM10:   .ASCIZ  <15><12>/DRIVE?/
2274  021420  037505      000
2275
2276
2277
2278                                          .SBTTL  DISK POWER FAILURE TEST
2279
2280                                  ;(DISK)POWER FAILURE (DURING DISK WRITE) TEST
2281                                  ;THIS TEST CHECKS THAT THE INFORMATION WRITTEN ON THE DISK IS
2282                                  ;NOT DESTROYED WHEN THE DISK SENSES A LOSS OF POWER WHILE DOING A WRITE
2283                                  ;AND RETRACTS THE HEADS. UPON ENTRY THE PROGRAM FINDS OUT THE
2284                                  ;FIRST AVAILABLE DRIVE INDICATES IT (DRIVE XX) AND PROCEEDS TO TEST.
2285                                  ;CYLINDERS 0 TO 15 ARE WRITTEN WITH UNIQUE PATTERNS, THEN THE HEADS
2286                                  ;ARE POSITIONED ON CYLINDER 10 (DECIMAL) AND A MESSAGE (DROP
2287                                  ;POWER) IS GIVEN.  AFTER RECEIVING THIS MESSAGE THE USER SHOULD
2288                                  ;DROP POWER FROM THE DRIVE.  ON SENSING A LOSS OF POWER, THE
2289                                  ;PROGRAM ASKS THE USER TO PUT BACK THE POWER.  THE ERRORS (DPL)
2290                                  ;ARE CLEARED AND A WRITE-CHECK IS PERFORMED TO CHECK IF THE
2291                                  ;UNIQUE PATTERNS ON THE DISK (CYLINDERS 0-9 AND 11-15) ARE STILL
2292                                  ;THERE.  IF NOT A WRITE CHECK ERROR IS REPORTED.
2293          021424                          .EVEN
2294
2295  021424  005000                  SECT.6: CLR     R0                      ;INITIALIZE DRIVE #
2296  021426  005037  020474                  CLR     DRIV0
2297  021432  010077  157740          8$:     MOV     R0,@RKDA                ;IS IT PRESENT?
2298  021436  105777  157722                  TSTB    @RKDS                   ;IF NOT SKIP
2299  021442  100406                          BMI     9$
2300  021444  005237  020474          10$:    INC     DRIV0
2301  021450  062700  020000                  ADD     #20000,R0
2302  021454  001366                          BNE     8$
2303  021456  000762                          BR      SECT.6
2304  021460  104401  020330          9$:     TYPE    ,EM1
2305  021464  013746  020474                  MOV     DRIV0,-(SP)             ;GET DRIVE #
2306  021470  104402                          TYPOC
2307  021472  012777  000001  157670          MOV     #1,@RKCS                ;CONTROL RESET
2308  021500  105777  157664                  TSTB    @RKCS
2309  021504  100375                          BPL     .-4
2310  021506  005005                          CLR     R5                      ;INITIALIZE PATTERN TO BE WRITTEN
2311                                                                          ;0 ON CYL 0, 1 ON CYL 1, ETC
2312  021510  013701  001376                  MOV     RKDA,R1
2313  021514  013702  001372                  MOV     RKWC,R2
2314  021520  013703  001374                  MOV     RKBA,R3
2315  021524  013704  001370                  MOV     RKCS,R4
2316  021530  010011                          MOV     R0,@R1
2317  021532  010537  022070          15:     MOV     R5,BUFR                 ;FILL THE PATTER IN DATA BUFFER.
2318  021536  012713  022070                  MOV     #BUFR,@R3               ;BUS ADRES
2319  021542  012712  164000                  MOV     #-14000,@R2             ;WRITE 1 CYL (256X12X2 WORDS)
2320  021546  012714  004003                  MOV     #4003,@R4               ;WRITE, GO, IBA SET
2321  021552  105714                          TSTB    @R4                     ;WAIT FOR CONTROL READY
2322  021554  100376                          BPL     .-2
```

```
2323
2324  021556  005205                          INC     R5                      ;DONE
2325  021560  020527  000020                  CMP     R5,#20                  ;WRITTEN ALL 15 CYLINDERS?
2326  021564  001362                          BNE     1$                      ;IF NOT, GO BAK
2327
2328  021566  010005                          MOV     R0,R5                   ;DRIVE #
2329  021570  052705  000500                  BIS     #500,R5                 ;CYL 10
2330  021574  012737  000012  022070          MOV     #12,BUFR                ;PATERN TO BE WRITTEN
2331  021602  010511                          MOV     R5,@R1                  ;ADRES THE DISK
2332  021604  012712  164000                  MOV     #-14000,@R2             ;WORD COUNT= 1 CYLINDER
2333  021610  012713  022070                  MOV     #BUFR,@R3               ;BUS ADRES
2334  021614  012714  004003                  MOV     #4003,@R4               ;WRITE, GO, IBA
2335  021620  032777  000100  157536  2$:     BIT     #RWS,@RKDS              ;WAIT FOR THE HEADS TO SETTLE
2336  021626  001774                          BEQ     2$                      ;ON CYL 10
2337  021630  104401  021636                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
2338  021634  000406                          BR      64$                     ;;GET OVER THE ASCIZ
2339                                  ;;65$:  .ASCIZ  /DROP POWER/
2340  021652                          64$:
2341
2342  021652  000407                          BR      5$
2343  021654  010511                  3$:     MOV     R5,@R1                  ;ADRES THE DISK
2344  021656  012712  164000                  MOV     #-14000,@R2             ;WORD COUNT= 1 CYLINDER
2345  021662  012713  022070                  MOV     #BUFR,@R3               ;BUS ADRES
2346  021666  012714  004003                  MOV     #4003,@R4               ;WRITE, GO, IBA
2347  021672  105714                  5$:     TSTB    @R4                     ;WAIT FOR CONTROL READY
2348  021674  100376                          BPL     .-2
2349  021676  005714                          TST     @R4                     ;WAIT FOR DRIVE POWER TO GO DOWN,
2350                                                                          ;OTHERWISE, KEEP ON WRITING ON CYL 10
2351  021700  100365                          BPL     3$
2352                                                                          ;IF DRIVE POWER LOSS WAS SENSED,
2353                                                                          ;ASK TO PUT POWER ON.
2354  021702  104401  021710                  TYPE    ,67$                    ;;TYPE ASCIZ STRING
2355  021706  000406                          BR      66$                     ;;GET OVER THE ASCIZ
2356                                  ;;67$:  .ASCIZ  <15><12>/POWER ON/
2357  021724                          66$:
2358  021724  105777  157434                  TSTB    @RKDS                   ;WAIT FOR DRIVE READY
2359  021730  100375                          BPL     .-4
2360  021732  012714  000001                  MOV     #1,@R4                  ;CONTROL RESET, CLEAR ERROR
2361  021736  105714                          TSTB    @R4
2362  021740  100376                          BPL     .-2
2363  021742  010077  157430                  MOV     R0,@RKDA
2364  021746  005005                          CLR     R5                      ;INITIALIZE PATTERN
2365  021750  010537  022070          6$:     MOV     R5,BUFR
2366  021754  012713  022070                  MOV     #BUFR,@R3
2367  021760  012712  164000                  MOV     #-14000,@R2
2368  021764  012714  004007                  MOV     #4007,@R4               ;WRITE CHECK, GO, IBA
2369  021770  105714                          TSTB    @R4
2370  021772  100376                          BPL     .-2
2371
2372  021774  005714                          TST     @R4                     ;ANY ERROR?
2373  021776  100023                          BPL     7$                      ;NO
2374  022000  104401  022006                  TYPE    ,69$                    ;;TYPE ASCIZ STRING
2375  022004  000416                          BR      68$                     ;;GET OVER THE ASCIZ
2376                                  ;;69$:  .ASCIZ  <15><12>/ERROR, ON PWR-UP,  RKDA=/
2377  022042                          68$:
2378  022042  011146                          MOV     @R1,-(SP)
```

```
                                        TYPOC
                        7$:     INC     R5
                                CMP     R5,#12          ;CYLINDER 10 ?
                                BEQ     7$              ;BR IF YES, INCREMENT AGAIN
                                CMP     R5,#20
                                BNE     6$

                                JMP     10$

                        BUFR:   .WORD   0


                                .SBTTL  TYPE ROUTINE

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

                        $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
                                BPL     1$              ;;BR IF YES
                                HALT                    ;;HALT HERE IF NO TERMINAL
                                BR      3$              ;;LEAVE
                        1$:     MOV     R0,-(SP)        ;;SAVE R0
                                MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
                        2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
                                BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
                                TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
                        60$:    MOV     (SP)+,R0        ;;RESTORE R0
                        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
                                RTI                     ;;RETURN
                        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
                                BEQ     8$
                                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
                                BNE     5$
                                TST     (SP)+           ;;POP  <CR><LF> EQUIV
                                TYPE                    ;;TYPE A CR AND LF
                                $CRLF
                                CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
                                BR      2$              ;;GET NEXT CHARACTER
                        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
                        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
                                BNE     2$              ;;IF NO GO GET NEXT CHAR.
                                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                        ;;AND THE NULL CHAR.
                        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
```

```
2435  022202  002770                          BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
2436  022204  004737  022242                  JSR     PC,$TYPEC               ;;GO TYPE A NULL
2437  022210  105337  022306                  DECB    $CHARCNT                ;;DO NOT COUNT AS A COUNT
2438  022214  000770                          BR      7$                      ;;LOOP
2439
2440                                          ;HORIZONTAL TAB PROCESSOR
2441
2442  022216  112716  000040          8$:     MOVB    #' ,(SP)                ;;REPLACE TAB WITH SPACE
2443  022222  004737  022242          9$:     JSR     PC,$TYPEC               ;;TYPE A SPACE
2444  022226  132737  000007  022306          BITB    #7,$CHARCNT             ;;BRANCH IF NOT AT
2445  022234  001372                          BNE     9$                      ;;TAB STOP
2446  022236  005726                          TST     (SP)+                   ;;POP SPACE OFF STACK
2447  022240  000724                          BR      2$                      ;;GET NEXT CHARACTER
2448  022242  105777  156702          $TYPEC: TSTB    @$TPS                   ;;WAIT UNTIL PRINTER IS READY
2449  022246  100375                          BPL     $TYPEC
2450  022250  116677  000002  156674          MOVB    2(SP),@$TPB             ;;LOAD CHAR TO BE TYPED INTO DATA REG.
2451  022256  122766  000015  000002          CMPB    #CR,2(SP)               ;;IS CHARACTER A CARRIAGE RETURN?
2452  022264  001003                          BNE     1$                      ;;BRANCH IF NO
2453  022266  105037  022306                  CLRB    $CHARCNT                ;;YES--CLEAR CHARACTER COUNT
2454  022272  000406                          BR      $TYPEX                  ;;EXIT
2455  022274  122766  000012  000002  1$:     CMPB    #LF,2(SP)               ;;IS CHARACTER A LINE FEED?
2456  022302  001402                          BEQ     $TYPEX                  ;;BRANCH IF YES
2457  022304  105227                          INCB    (PC)+                   ;;COUNT THE CHARACTER
2458  022306  000000          $CHARCNT:.WORD  0                               ;;CHARACTER COUNT STORAGE
2459  022310  000207          $TYPEX: RTS     PC
2460
2461                                          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
2462
2463                                          ;;**************************************************************
2464                                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
2465                                          ;*OCTAL (ASCII) NUMBER AND TYPE IT.
2466                                          ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
2467                                          ;*CALL:
2468                                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2469                                          ;*      TYPOS                   ;;CALL FOR TYPEOUT
2470                                          ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
2471                                          ;*      .BYTE   M               ;;M=1 OR 0
2472                                          ;*                                      ;;1=TYPE LEADING ZEROS
2473                                          ;*                                      ;;0=SUPPRESS LEADING ZEROS
2474                                          ;*
2475                                          ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
2476                                          ;*$TYPOS OR $TYPOC
2477                                          ;*CALL:
2478                                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2479                                          ;*      TYPON                   ;;CALL FOR TYPEOUT
2480                                          ;*
2481                                          ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
2482                                          ;*CALL:
2483                                          ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
2484                                          ;*      TYPOC                   ;;CALL FOR TYPEOUT
2485
2486  022312  017646  000000          $TYPOS: MOV     @(SP),-(SP)             ;;PICKUP THE MODE
2487  022316  116637  000001  022535          MOVB    1(SP),$OFILL            ;;LOAD ZERO FILL SWITCH
2488  022324  112637  022537                  MOVB    (SP)+,$OMOD+1           ;;NUMBER OF DIGITS TO TYPE
2489  022330  062716  000002                  ADD     #2,(SP)                 ;;ADJUST RETURN ADDRESS
2490  022334  000406                          BR      $TYPON
```

```
2491  022336  112737  000001  022535  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
2492  022344  112737  000006  022537          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
2493  022352  112737  000005  022534  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
2494  022360  010346                          MOV     R3,-(SP)        ;;SAVE R3
2495  022362  010446                          MOV     R4,-(SP)        ;;SAVE R4
2496  022364  010546                          MOV     R5,-(SP)        ;;SAVE R5
2497  022366  113704  022537                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
2498  022372  005404                          NEG     R4
2499  022374  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
2500  022400  110437  022536                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
2501  022404  113704  022535                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
2502  022410  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
2503  022414  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
2504  022416  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
2505  022420  000404                          BR      3$              ;;GO DO MSB
2506  022422  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
2507  022424  006105                          ROL     R5
2508  022426  006105                          ROL     R5
2509  022430  010503                          MOV     R5,R3
2510  022432  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
2511  022434  105337  022536                  DECB    $OMODE          ;;TYPE THIS DIGIT?
2512  022440  100016                          BPL     7$              ;;BR IF NO
2513  022442  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
2514  022446  001002                          BNE     4$              ;;TEST FOR 0
2515  022450  005704                          TST     R4              ;;SUPPRESS THIS 0?
2516  022452  001403                          BEQ     5$              ;;BR IF YES
2517  022454  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
2518  022456  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
2519  022462  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
2520  022466  110337  022532                  MOVB    R3,8$           ;;SAVE FOR TYPING
2521  022472  104401  022532                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
2522  022476  105337  022534          7$:     DECB    $OCNT           ;;COUNT BY 1
2523  022502  003347                          BGT     2$              ;;BR IF MORE TO DO
2524  022504  002402                          BLT     6$              ;;BR IF DONE
2525  022506  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
2526  022510  000744                          BR      2$              ;;GO DO THE LAST DIGIT
2527  022512  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
2528  022514  012604                          MOV     (SP)+,R4        ;;RESTORE R4
2529  022516  012603                          MOV     (SP)+,R3        ;;RESTORE R3
2530  022520  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
2531  022526  012616                          MOV     (SP)+,(SP)
2532  022530  000002                          RTI                     ;;RETURN
2533  022532     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
2534  022533     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
2535  022534     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
2536  022535     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
2537  022536  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
2538                          .SBTTL  TTY INPUT ROUTINE
2539
2540                          ;;**************************************************************
2541                          .ENABL  LSB
2542  022540  000000          $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
2543  022542  000000          $TKQIN: .WORD   0               ;;INPUT POINTER
2544  022544  000000          $TKQOUT: .WORD  0               ;;OUTPUT POINTER
2545  022546  000001          $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
2546          022547          $TKQEND=.
```

```
                        022550                          .EVEN
                                                        ;*TK INITIALIZE ROUTINE
                                                        ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                                        ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                                        ;
                                                        ;*CALL:
                                                        ;*      JSR     PC,$TKINT
                                                        ;*      RETURN
                                                        ;
022550  005037  022540                  $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
022554  012737  022546  022542                  MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
022562  013737  022542  022544                  MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
022570  012737  022620  000060                  MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
022576  012737  000200  000062                  MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
022604  005777  156336                          TST     @$TKB           ;;CLEAR DONE FLAG
022610  012777  000100  156326                  MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
022616  000207                                  RTS     PC              ;;RETURN TO CALLER
                                                        ;*TK SERVICE ROUTINE
                                                        ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                                        ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                                        ;*IT IN THE QUEUE.
                                                        ;
022620  117746  156322                  $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
022624  042716  177600                          BIC     #^C177,(SP)     ;;STRIP THE JUNK
022630  021627  000007                  1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
022634  001004                                  BNE     2$              ;;BRANCH IF NO
022636  022737  000176  001140                  CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
022644  001500                                  BEQ     6$              ;;GO TO SWR CHANGE
                                                        ;
022646                                  2$:
022646  022737  000001  022540                  CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
022654  001004                                  BNE     3$              ;;BRANCH IF NO
022656  104401  023670                          TYPE    ,$BELL          ;;RING THE TTY BELL
022662  005726                                  TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
022664  000451                                  BR      5$              ;;EXIT
022666  021627  000023                  3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
022672  001021                                  BNE     32$             ;;BRANCH IF NO
022674  005077  156244                          CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
022700  005726                                  TST     (SP)+           ;;CLEAN CHAR OFF STACK
022702  105777  156236                  31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
022706  100375                                  BPL     31$             ;;LOOP UNTIL ITS THERE
022710  117746  156232                          MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
022714  042716  177600                          BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
022720  022627  000021                          CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
022724  001366                                  BNE     31$             ;;BRANCH IF NO
022726  012777  000100  156210                  MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
022734  000002                                  RTI                     ;;RETURN
022736  005237  022540                  32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
022742  021627  000140                          CMP     (SP),#140       ;;IS IT UPPER CASE?
022746  002405                                  BLT     4$              ;;BRANCH IF YES
022750  021627  000175                          CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
022754  003002                                  BGT     4$              ;;BRANCH IF YES
022756  042716  000040                          BIC     #40,(SP)        ;;MAKE IT UPPER CASE
022762  112677  177554                  4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
```

```
2603  022766  005237  022542                  INC     $TKQIN                  ;;UPDATE THE POINTER
2604  022772  023727  022542  022547          CMP     $TKQIN,#$TKQEND         ;;GO OFF THE END?
2605  023000  001003                          BNE     5$                      ;;BRANCH IF NO
2606  023002  012737  022546  022542          MOV     #$TKQSRT,$TKQIN         ;;RESET THE POINTER
2607  023010  000002                  5$:     RTI                             ;;RETURN
2608
2609                                          ;;**************************************************************
2610                                          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
2611                                          ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
2612                                          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
2613                                          ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
2614  023012  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR              ;;IS THE SOFT-SWR SELECTED
2615  023020  001104                          BNE     15$                     ;;EXIT IF NOT
2616  023022  105777  156116                  TSTB    @$TKS                   ;;IS A CHAR WAITING?
2617  023026  100101                          BPL     15$                     ;;IF NOT, EXIT
2618  023030  117746  156112                  MOVB    @$TKB,-(SP)             ;;YES
2619  023034  042716  177600                  BIC     #^C177,(SP)             ;;MAKE IT  7-BIT ASCII
2620  023040  021627  000007                  CMP     (SP),#7                 ;;IS IT A CONTROL-G?
2621  023044  001300                          BNE     2$                      ;;IF NOT, PUT IT IN THE TTY QUEUE
2622                                                                          ;;AND EXIT
2623
2624                                          ;;**************************************************************
2625                                          ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
2626                                          ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
2627                                          ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
2628  023046  123727  001134  000001  6$:     CMPB    $AUTOB,#1               ;;ARE WE RUNNING IN AUTO-MODE?
2629  023054  001674                          BEQ     2$                      ;;BRANCH IF YES
2630  023056  005726                          TST     (SP)+                   ;;CLEAR CONTROL-G OFF STACK
2631  023060  004737  022550                  JSR     PC,$TKINT               ;;FLUSH THE TTY INPUT QUEUE
2632  023064  005077  156054                  CLR     @$TKS                   ;;DISABLE TTY KEYBOARD INTERRUPTS
2633  023070  112737  000001  001135          MOVB    #1,$INTAG               ;;SET INTERRUPT MODE INDICATOR
2634
2635  023076  104401  023701                  TYPE    ,$CNTLG                 ;;ECHO THE CONTROL-G (^G)
2636  023102  104401  023706          $GTSWR: TYPE    ,$MSWR                  ;;TYPE CURRENT CONTENTS
2637  023106  013746  000176                  MOV     SWREG,-(SP)             ;;SAVE SWREG FOR TYPEOUT
2638  023112  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
2639  023114  104401  023717                  TYPE    ,$MNEW                  ;;PROMPT FOR NEW SWR
2640  023120  005046                  19$:    CLR     -(SP)                   ;;CLEAR COUNTER
2641  023122  005046                          CLR     -(SP)                   ;;THE NEW SWR
2642  023124  105777  156014          7$:     TSTB    @$TKS                   ;;CHAR THERE?
2643  023130  100375                          BPL     7$                      ;;IF NOT TRY AGAIN
2644
2645  023132  117746  156010                  MOVB    @$TKB,-(SP)             ;;PICK UP CHAR
2646  023136  042716  177600                  BIC     #^C177,(SP)             ;;MAKE IT 7-BIT ASCII
2647
2648
2649
2650  023142  021627  000025          9$:     CMP     (SP),#25                ;;IS IT A CONTROL-U?
2651  023146  001005                          BNE     10$                     ;;BRANCH IF NOT
2652  023150  104401  023674                  TYPE    ,$CNTLU                 ;;YES, ECHO CONTROL-U (^U)
2653  023154  062706  000006          20$:    ADD     #6,SP                   ;;IGNORE PREVIOUS INPUT
2654  023160  000757                          BR      19$                     ;;LET'S TRY IT AGAIN
2655
2656
2657  023162  021627  000015          10$:    CMP     (SP),#15                ;;IS IT A <CR>?
2658  023166  001022                          BNE     16$                     ;;BRANCH IF NO
```

```
2659  023170  005766  000004          TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
2660  023174  001403                  BEQ     11$             ;;BRANCH IF YES
2661  023176  016677  000002  155734  MOV     2(SP),@SWR      ;;SAVE NEW SWR
2662  023204  062706  000006          11$:    ADD     #6,SP   ;;CLEAR UP STACK
2663  023210  104401  001161          14$:    TYPE    ,$CRLF  ;;ECHO <CR> AND <LF>
2664  023214  123727  001135  000001  CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
2665  023222  001003                  BNE     15$             ;;BRANCH IF NOT
2666  023224  012777  000100  155712  MOV     #100,@STKS      ;;RE-ENABLE TTY KBD INTERRUPTS
2667  023232  000002                  15$:    RTI             ;;RETURN
2668  023234  004737  022242          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
2669  023240  021627  000060          CMP     (SP),#60        ;;CHAR < 0?
2670  023244  002420                  BLT     18$             ;;BRANCH IF YES
2671  023246  021627  000067          CMP     (SP),#67        ;;CHAR > 7?
2672  023252  003015                  BGT     18$             ;;BRANCH IF YES
2673  023254  042726  000060          BIC     #60,(SP)+       ;;STRIP-OFF ASCII
2674  023260  005766  000002          TST     2(SP)           ;;IS THIS THE FIRST CHAR
2675  023264  001403                  BEQ     17$             ;;BRANCH IF YES
2676  023266  006316                  ASL     (SP)            ;;NO, SHIFT PRESENT
2677  023270  006316                  ASL     (SP)            ;;   CHAR OVER TO MAKE
2678  023272  006316                  ASL     (SP)            ;;   ROOM FOR NEW ONE.
2679  023274  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
2680  023300  056616  177776          BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
2681  023304  000707                  BR      7$              ;;GET THE NEXT ONE
2682  023306  104401  001160          18$:    TYPE    ,$QUES  ;;TYPE ?<CR><LF>
2683  023312  000720                  BR      20$             ;;SIMULATE CONTROL-U
2684                                  .DSABL  LSB
2685
2686
2687                                  ;;*****************************************************************
2688                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2689                                  ;*CALL:
2690                                  ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
2691                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
2692                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
2693                                  ;
2694
2695  023314  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
2696  023316  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
2697  023324  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
2698  023330  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
2699  023332  012746  023340                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
2700  023336  000002                          RTI                     ;;POP NEW PC AND PS
2701  023340                          64$:
2702  023340  005737  022540          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
2703  023344  001775                          BEQ     1$
2704  023346  005337  022540                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
2705  023352  117766  177166  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
2706  023360  005237  022544                  INC     $TKQOUT         ;;UPDATE THE POINTER
2707  023364  023727  022544  022547          CMP     $TKQOUT,#$TKQEND        ;;DID IT GO OFF OF THE END?
2708  023372  001003                          BNE     2$              ;;BRANCH IF NO
2709  023374  012737  022546  022544          MOV     #$TKQSRT,$TKQOUT        ;;RESET THE POINTER
2710  023402  000002                  2$:     RTI                     ;;RETURN
2711                                  ;;*****************************************************************
2712                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2713                                  ;*CALL:
2714                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
```

```
2715                                    ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2716                                    ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2717
2718  023404  010346                    $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
2719  023406  005046                            CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
2720  023410  012703  023640            1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
2721  023414  022703  023670            2$:     CMP     #$TTYIN+30,R3   ;;BUFFER FULL?
2722  023420  101456                            BLOS    4$              ;;BR IF YES
2723  023422  104407                            RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
2724  023424  112613                            MOVB    (SP)+,(R3)      ;;GET CHARACTER
2725  023426  122713  000177            10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
2726  023432  001022                            BNE     5$              ;;BR IF NO
2727  023434  005716                            TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
2728  023436  001007                            BNE     6$              ;;BR IF NO
2729  023440  112737  000134  023636            MOVB    #'\,9$          ;;TYPE A BACK SLASH
2730  023446  104401  023636                    TYPE    ,9$
2731  023452  012716  177777                    MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
2732  023456  005303                    6$:     DEC     R3              ;;BACKUP BY ONE
2733  023460  020327  023640                    CMP     R3,#$TTYIN      ;;STACK EMPTY?
2734  023464  103434                            BLO     4$              ;;BR IF YES
2735  023466  111337  023636                    MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
2736  023472  104401  023636                    TYPE    ,9$             ;;GO TYPE
2737  023476  000746                            BR      2$              ;;GO READ ANOTHER CHAR.
2738  023500  005716                    5$:     TST     (SP)            ;;RUBOUT KEY SET?
2739  023502  001406                            BEQ     7$              ;;BR IF NO
2740  023504  112737  000134  023636            MOVB    #'\,9$          ;;TYPE A BACK SLASH
2741  023512  104401  023636                    TYPE    ,9$
2742  023516  005016                            CLR     (SP)            ;;CLEAR THE RUBOUT KEY
2743  023520  122713  000025            7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
2744  023524  001003                            BNE     8$              ;;BR IF NO
2745  023526  104401  023674                    TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
2746  023532  000726                            BR      1$              ;;GO START OVER
2747  023534  122713  000022            8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
2748  023540  001011                            BNE     3$              ;;BRANCH IF NO
2749  023542  105013                            CLRB    (R3)            ;;CLEAR THE CHARACTER
2750  023544  104401  001161                    TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
2751  023550  104401  023640                    TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
2752  023554  000717                            BR      2$              ;;GO PICKUP ANOTHER CHACTER
2753  023556  104401  001160            4$:     TYPE    ,$QUES          ;;TYPE A '?'
2754  023562  000712                            BR      1$              ;;CLEAR THE BUFFER AND LOOP
2755  023564  111337  023636            3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
2756  023570  104401  023636                    TYPE    ,9$
2757  023574  122723  000015                    CMPB    #15,(R3)+       ;;CHECK FOR RETURN
2758  023600  001305                            BNE     2$              ;;LOOP IF NOT RETURN
2759  023602  105313  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
2760  023606  104401  001162                    TYPE    ,$LF            ;;TYPE A LINE FEED
2761  023612  005726                            TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
2762  023614  012603                            MOV     (SP)+,R3        ;;RESTORE R3
2763  023616  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
2764  023620  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
2765  023626  012766  023640  000004            MOV     #$TTYIN,4(SP)
2766  023634  000002                            RTI                     ;;RETURN
2767  023636     000                    9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
2768  023637     000                            .BYTE   0               ;;TERMINATOR
2769  023640  000030                    $TTYIN: .BLKB   30              ;;RESERVE 30 BYTES FOR TTY INPUT
2770  023670  177607  000377            $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
```

```
2771  023674  052536  005015      000  $CNTLU: .ASCIZ  /↑U/<15><12>     ;;CONTROL "U"
2772  023701      136  005107  000012  $CNTLG: .ASCIZ  /↑G/<15><12>     ;;CONTROL "G"
2773  023706  005015  053523  020122  $MSWR:  .ASCIZ  <15><12>/SWR = /
2774  023714  020075      000
2775  023717      040  047040  053505  $MNEW:  .ASCIZ  /  NEW = /
2776  023724  036440  000040
2777                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
2778
2779                                  ;;**********************************************************************
2780                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
2781                                  ;*CHANGE IT TO BINARY.
2782                                  ;*CALL:
2783                                  ;*      RDOCT                       ;;READ AN OCTAL NUMBER
2784                                  ;*      RETURN HERE                 ;;LOW ORDER BITS ARE ON TOP OF THE STACK
2785                                  ;*                                  ;;HIGH ORDER BITS ARE IN $HIOCT
2786
2787  023730  011646                  $RDOCT: MOV     (SP),-(SP)          ;;PROVIDE SPACE FOR THE
2788  023732  016666  000004  000002          MOV     4(SP),2(SP)         ;;INPUT NUMBER
2789  023740  010046                          MOV     R0,-(SP)            ;;PUSH R0 ON STACK
2790  023742  010146                          MOV     R1,-(SP)            ;;PUSH R1 ON STACK
2791  023744  010246                          MOV     R2,-(SP)            ;;PUSH R2 ON STACK
2792  023746  104410                  1$:     RDLIN                       ;;READ AN ASCIZ LINE
2793  023750  012600                          MOV     (SP)+,R0            ;;GET ADDRESS OF 1ST CHARACTER
2794  023752  005001                          CLR     R1                  ;;CLEAR DATA WORD
2795  023754  005002                          CLR     R2
2796  023756  112046                  2$:     MOVB    (R0)+,-(SP)         ;;PICKUP THIS CHARACTER
2797  023760  001412                          BEQ     3$                  ;;IF ZERO GET OUT
2798  023762  006301                          ASL     R1                  ;;*2
2799  023764  006102                          ROL     R2
2800  023766  006301                          ASL     R1                  ;;*4
2801  023770  006102                          ROL     R2
2802  023772  006301                          ASL     R1                  ;;*8
2803  023774  006102                          ROL     R2
2804  023776  042716  177770                  BIC     #^C7,(SP)           ;;STRIP THE ASCII JUNK
2805  024002  062601                          ADD     (SP)+,R1            ;;ADD IN THIS DIGIT
2806  024004  000764                          BR      2$                  ;;LOOP
2807  024006  005726                  3$:     TST     (SP)+               ;;CLEAN TERMINATOR FROM STACK
2808  024010  010166  000012                  MOV     R1,12(SP)           ;;SAVE THE RESULT
2809  024014  010237  024030                  MOV     R2,$HIOCT
2810  024020  012602                          MOV     (SP)+,R2            ;;POP STACK INTO R2
2811  024022  012601                          MOV     (SP)+,R1            ;;POP STACK INTO R1
2812  024024  012600                          MOV     (SP)+,R0            ;;POP STACK INTO R0
2813  024026  000002                          RTI                         ;;RETURN
2814  024030  000000                  $HIOCT: .WORD   0                   ;;HIGH ORDER BITS GO HERE
2815                                  .SBTTL  TRAP DECODER
2816
2817                                  ;;**********************************************************************
2818                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
2819                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
2820                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
2821                                  ;*GO TO THAT ROUTINE.
2822
2823  024032  010046                  $TRAP:  MOV     R0,-(SP)            ;;SAVE R0
2824  024034  016600  000002                  MOV     2(SP),R0            ;;GET TRAP ADDRESS
2825  024040  005740                          TST     -(R0)               ;;BACKUP BY 2
2826  024042  111000                          MOVB    (R0),R0             ;;GET RIGHT BYTE OF TRAP
```

```
2827  024044  006300                          ASL     R0                  ;;POSITION FOR INDEXING
2828  024046  016000  024066                  MOV     $TRPAD(R0),R0       ;;INDEX TO TABLE
2829  024052  000200                          RTS     R0                  ;;GO TO ROUTINE
2830
2831
2832                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
2833
2834  024054  011646                  $TRAP2: MOV     (SP),-(SP)          ;;MOVE THE PC DOWN
2835  024056  016666  000004  000002          MOV     4(SP),2(SP)         ;;MOVE THE PSW DOWN
2836  024064  000002                          RTI                         ;;RESTORE THE PSW
2837
2838                                  .SBTTL  TRAP TABLE
2839
2840                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
2841                                  ;*BY THE "TRAP" INSTRUCTION.
2842
2843                                  ;       ROUTINE
2844                                  ;       -------
2845  024066  024054                  $TRPAD: .WORD   $TRAP2
2846  024070  022072                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
2847  024072  022336                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
2848  024074  022312                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
2849  024076  022352                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
2850
2851  024100  023102                          $GTSWR  ;;CALL=GTSWR    TRAP+5(104405)  GET SOFT-SWR SETTING
2852
2853  024102  023012                          $CKSWR  ;;CALL=CKSWR    TRAP+6(104406)  TEST FOR CHANGE IN SOFT-SWR
2854  024104  023314                          $RDCHR  ;;CALL=RDCHR    TRAP+7(104407)  TTY TYPEIN CHARACTER ROUTINE
2855  024106  023404                          $RDLIN  ;;CALL=RDLIN    TRAP+10(104410) TTY TYPEIN STRING ROUTINE
2856  024110  023730                          $RDOCT  ;;CALL=RDOCT    TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
2857                                  .SBTTL  POWER DOWN AND UP ROUTINES
2858
2859                                  ;;**************************************************************
2860                                  ;POWER DOWN ROUTINE
2861  024112  012737  024252  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC    ;;SET FOR FAST UP
2862  024120  012737  000340  000026          MOV     #340,@#PWRVEC+2     ;;PRIO:7
2863  024126  010046                          MOV     R0,-(SP)            ;;PUSH R0 ON STACK
2864  024130  010146                          MOV     R1,-(SP)            ;;PUSH R1 ON STACK
2865  024132  010246                          MOV     R2,-(SP)            ;;PUSH R2 ON STACK
2866  024134  010346                          MOV     R3,-(SP)            ;;PUSH R3 ON STACK
2867  024136  010446                          MOV     R4,-(SP)            ;;PUSH R4 ON STACK
2868  024140  010546                          MOV     R5,-(SP)            ;;PUSH R5 ON STACK
2869  024142  017746  154772                  MOV     @SWR,-(SP)          ;;PUSH @SWR ON STACK
2870  024146  010637  024256                  MOV     SP,$SAVR6           ;;SAVE SP
2871  024152  012737  024164  000024          MOV     #$PWRUP,@#PWRVEC    ;;SET UP VECTOR
2872  024160  000000                          HALT
2873  024162  000776                          BR      .-2                 ;;HANG UP
2874
2875                                  ;;**************************************************************
2876                                  ;POWER UP ROUTINE
2877  024164  012737  024252  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC    ;;SET FOR FAST DOWN
2878  024172  013706  024256                  MOV     $SAVR6,SP           ;;GET SP
2879  024176  005037  024256                  CLR     $SAVR6              ;;WAIT LOOP FOR THE TTY
2880  024202  005237  024256          1$:     INC     $SAVR6              ;;WAIT FOR THE INC
2881  024206  001375                          BNE     1$                  ;;OF  WORD
2882  024210  012677  154724                  MOV     (SP)+,@SWR          ;;POP STACK INTO @SWR
```

```
2883  024214  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
2884  024216  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
2885  024220  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
2886  024222  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
2887  024224  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
2888  024226  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
2889  024230  012737  024112  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
2890  024236  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
2891  024244  104401                          TYPE                    ;;REPORT THE POWER FAILURE
2892  024246  024260                $PWRMG:   .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
2893  024250  000002                          RTI
2894  024252  000000                $ILLUP:   HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
2895  024254  000776                          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
2896  024256  000000                $SAVR6:   0                       ;;PUT THE SP HERE
2897  024260  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
2898  024266  000122
2899                                          .EVEN
2900          000001                          .END
```

```
AUTSL2  002630            440#    657
BA      001406            275#   1363*   1365*   1367*   1369*   1390    1424
BADINP  003260            501     503     510#
BADONE  010344           1142#
BAD.IN  011756           1303#
BAD.ON  013740           1567#   1583    1585
BASE    001266            219#    832
BASINC  005344            837     849#
BEGCT   017302           1874#   2099
BEGCT1  017310           1877#   2098
BEGIN   002606            418     424#
BIT0  = 000001            114#   1075    1395    1412    1429    1964    1971    1975    1982
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#   1393    1428    2024    2026    2035    2037
BIT10 = 002000             94#   1392    1409
BIT11 = 004000             93#   1392    1426
BIT12 = 010000             92#    289
BIT13 = 020000             91#   1767    1777    2208    2221
BIT14 = 040000             90#    505     538     636     670    1415    1544
BIT15 = 100000             89#    606     997     999    1108    1662    1678
BIT2  = 000004            112#   1410    1428    1921    1927    1931    1939
BIT3  = 000010            111#
BIT4  = 000020            110#    797     799     802     934     936     939
BIT5  = 000040            109#    291     751    1753
BIT6  = 000100            108#    290    1353    1358    1468    1854    1948
BIT7  = 000200            107#    587     591    1110    1854    1884    1906
BIT8  = 000400            106#   1426
BIT9  = 001000            105#    292
BPTVEC= 000014            121#
BUFF    013656            275    1558#
BUFR    022070           2317*   2318    2330*   2333    2345    2365*   2366    2388#
BUSADR  001336            253#    793*    864     924*
CHECK1  005434            868#    878     882
CHKCNT  000350            258#   1011*   1052*
CKSWR = 104406           2853#
CLRDPL  020216           1965    1973    2067#
CNTSIN  001322            242#    702*    973*
COM     012304           1348#   1361
COMMON  012300           1347#
COMND   001316            238#    785*    901*    993
CONTIN  011370           1227    1229#   1295
CONTRL  001332            251#    795*    866     927*
CR    = 000015             29#   1378    2451    2461
CRLF  = 000200             30#    363    2422    2461
CYCLE   004160            531     669#
CYCLE2  004204            675#
CYCL2   004174            671     673#
```

```
CYL     013530  1511* 1515* 1519* 1524*
CYLADR  005266   790   832*  919
CYLAD2  005272   833*  853
CYLBAS  001315   237*
CYLCNT  001342   255*  794
CYLOFF  005276   805   834*  942
CYLTBL  001274   226*  818   833   851
DA      001410   276*  1362* 1371* 1372  1388  1405  1422  1437
DOISP = 177570    36*  181   335
DISPLA  001142   181*  335*  343*
DISPRE  000174   136*  343
DN1DRV  020232  1952   2023  2025  2031  2054  2063  2081*
DO2     004234   543   611   684*
DPL   = 010000   289*  1961
DRACTV  001164   194*  530*
DRCNT1  001311   233*  452*  506   651*  931
DRIVE   001314   236*  688*  707   787   1586* 1598
DRIVO   020474  1157  1172  1195  1259  1269  1276  1287  1841  1874  2145* 2201* 2208* 2210
                2296* 2300* 2305
DRIV1   020476  2146*
DRIV2   020500  2147*
DRIV3   020502  2148*
DRIV4   020504  2149*
DRIV5   020506  2150*
DRIV6   020510  2151*
DRIV7   020512  2152*
DRVCNT  020326  1319* 1323* 1324  1327  2105*
DRVO    001206   198*  464   525   536   604   652   682   789   906   917
DSKADR  001334   252*  695*  727   797   799*  802*  846*  847*  863   922*  923*  930*  934
                936*  939*  1036
DSKTMP  001352   259*  694*  922   1338* 1349  1389  1406  1423  1593* 1628  1657  1722  1727
                1734  1760  1783
DSWR  = 177570    35*  180   334
ECNT    001321   241*  701*  960*  975
EMTVEC= 000030   124*
EM1     020330  1329  1850  1885  1902  1924  1935  1967  1978  2027  2038  2107* 2304
EM10    021412  2187  2273*
EM2     020341  1853  1888  2109*
EM3     020353  1905  2111*
EM4     020366  1938  2113*
EM5     020401  1928  2115*
EM6     020417  1970  2118*
EM7     020440  1981  2121*
EM8     020453  2041  2123*
EM9     020461  2030  2125*
ERRCHK  006066   870   952*
ERRFLG  001360   262*  871   873*  957*  967*
ERRF    001424   282*  1340* 1417* 1474* 1475
ERRFC   001426   283*  1341* 1441* 1481* 1484
ERRVEC= 000004   117*  332   333*  344*
ERRWCH  001430   284*  1342* 1449* 1550* 1551
ERRWCS  001432   285*  1343* 1450* 1546* 1547
ERRWF   001422   281*  1339* 1400* 1456* 1457
EXECUT  005366   796   860*  874   888   928
EXIT    003352   526   535*
EXITA   003640   598*  600   638
```

```
EXITB   003646          601#    634
EXITX   003414          537     539     547#
EXIT2   006062          907     946#
EXTFR2  003356          536#    541     546     1132
EXTR    001420          280#    1346*   1394    1411    1427
FAIL    013302          1490#   1548
FAILED  013264          1485    1487#
FFLAG   021406          2188*   2209*   2219    2229    2271#
FIL.DN  003656          604#    607
GD1     012512          1325    1380#
GNS   = ****** U        135     362     374     378     382     386     390     394     398     402     406     411     413
                        447     476     488     495     513     550     560     568     577     616     660     705     713
                        717     741     748     755     775     886     980     986     991     1020    1029    1034    1042
                        1048    1085    1089    1094    1102    1118    1145    1154    1182    1186    1207    1211    1215
                        1219    1239    1306    1311    1377    1462    1493    1570    1575    1596    1604    1608    1612
                        1618    1624    1637    1641    1645    1650    1654    1666    1672    1682    1688    1693    1697
                        1701    1706    1710    1714    1718    1747    1757    1771    1781    1799    1804    1812    1818
                        1893    2019    2060    2177    2181    2185    2339    2356    2376    2846    2847    2848    2849
                        2851    2853    2854    2855    2856
GO      003276          509     520#
GOOT    012466          1374#
GO1     003310          524#    529     534     1001
GO2     003322          522     527#
GO3     003330          528     530#
GTSWR = 104405          357     1190    1226    1315    2213    2851#
HDALGN  020756          2187#   2193    2195    2205    2207    2269
HDRFLG  001320          240#    1009*   1016    1022*
HT    = 000011          27#     2420    2461
IDEX    001324          244#    521*    673     699*
ILEGAL  004126          463     647     657#
INITIL  004510          720     726#    883
INITI2  004522          729#    750     757
INIT.2  010452          1167#
INSYS2  003512          565#    597
IO      012516          1364    1366    1368    1370    1387#   1470    1549
IOTVEC= 000020          122#
KYTEMP  001330          250#    453*    454*    455     457*    480*    481     482     484     498*    499     500*    502
                        504*    505*    508     645     648*    649     650*
LDFLG   004210          676#    1129
LDSEEK  011570          1259#   1294
LF    = 000012          28#     1378    2455    2461
LFLF  = 105212          274#
LOGA    001166          196#    451     508*    520     535     564     676     902     1069    1123
MAINSEL 010342          1054    1136#   1750    1822
MASK    005220          610     675     813#    918
MASKER  004014          593     595     629#
MODE    001312          141*    234#    314*    364
MORE    013044          1437#   1440
MOUNT   004302          696     699#
MSKTBL  001256          210#    674     1065
NEXT    012402          1363#   1373
NG      002600          416     419#
NXT1    017464          1883    1898    1901    1914#
NXT2    017560          1916    1922    1929    1932    1948#
OSPFLG  001326          246#    371*    1203    1221*
PASTBL  001246          208#    1067    1082
```

```
PATTRN  001362          263#   789*   793    800*   803*   917*   937*   940*   1013   1721*  1728   1759*  1761
PIRQ    = 177772         34#
PIRQVE= 000240          128#
PRONUM  001313          235#   491*   516*
PRO1    003272          485    516#
PRO2    003144          483    491#   515
PRTTWO  014604         1634#
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     316*
PSW     = 177776          32#
PWRVEC= 000024          123#   328*   329*  2861*  2862*  2871*  2877*  2889*  2890*
RBA     001414          278#  1407   1435   1436
RBUFF   013660          278   1559#
RDBUFF  007342          924   1012   1134#  1735   1741   1784   1791   2261
RDCHK   006434          929   1007#
RDCHKO  016700         1778   1783#
RDCHR   = 104407         479    497   1577   2723   2854#
RDLIN   = 104410         449   2189   2792   2855#
RDLINK  005616          533    544    613    899#  1130
RDOCT   = 104411         413    579   1120   2856#
RDTBL   001226          207#
RD1     005642          905#   910    921    944
RD2     005654          904    908#
RD3     005662          909    911#
RD4     005724          920    922#   943
RD5     005732          923#   932    938
RD6     005736          924#
RD7     006036          935    939#
READCS  001356          261#   927   1737   1786
RESTRT  006216          962    975#
RESVEC= 000010          118#
RETFR2  007254         1115#
RETRN2  006432          965    974   1002#
RETRN3  005360          852    854#
RETRN4  004022          630    632#
RETRY   013172         1458   1466#  1476   1486   1553
RFCERR  013240         1438   1481#
RFERR   013222         1416   1474#
RKBA    001374          268#   864*  1390*  1407*  1424*  1728*  1735*  1761*  1784*  2261*  2314
RKCS    001370          266#   733*   735    760*   762    866*   868    952    955*   958    966*   969   1168*
                       1170   1174*  1176   1229*  1230   1262*  1264   1279*  1281   1350   1352*  1355  1357*
                       1391*  1392*  1393*  1394*  1395*  1396   1398   1408*  1409*  1410*  1411*  1412*  1413
                       1415   1425*  1426*  1427*  1428*  1429*  1445   1447   1466*  1467*  1482   1544   1629*
                       1631   1658*  1660   1668   1674*  1676   1684   1723*  1725   1730*  1732   1737*  1739
                       1763*  1765   1773*  1775   1786*  1788   1838*  1839   1857*  1895*  1956*  1958
                       2044*  2045   2048*  2050   2067*  2068   2085*  2087   2223*  2224   2226*  2227   2240
                       2242*  2243   2250*  2251   2262*  2264   2307*  2308   2315
RKDA    001376          269#   732*   743*   759*   863*  1159*  1167   1173*  1260*  1261*  1270*  1277*  1278*
                       1288*  1348*  1349*  1388*  1389*  1405*  1406*  1422*  1423*  1487*  1489*  1495   1628*
```

```
                    1657*  1722*  1727*  1734*  1760*  1783*  1846*  1877*  1878*  1954*  2043   2047*  2081*
                    2084*  2222*  2249*  2259*  2297*  2312   2363*
RKDS    001364       264*   744    751    764    963    971   1160   1222   1271   1289   1353   1358   1468
                    1753   1847   1859   1879   1889   1914   1918   1950   1961   1997   2006   2053   2082
                    2091   2298   2335   2358
RKER    001366       265*   1662   1678   1767   1777
RKHC    001372       267*   865*   1387*  1404*  1421*  1729*  1736*  1762*  1785*  2260*  2313
RHC     001416       279*   1344*  1404   1433
RHS   = 000100       290*   1271   1289   1859   1997   2053   2091   2335
RHSRDY  011342      1204    1222*  1223
SEC     013542      1499*   1503*  1526*
SECCNT  001344       256*    926   1729   1736   1762   1785
SECOND  003436       366     557*
SECONE  006736       672    1063*
SECSYS  003034       465     473*   490    655
SECTBL  001305       228*    903    933
SECT.0  013750       428    1572*
SECT.1  011766       427    1308*
SECT.2  010354       426    1151*
SECT.3  002626       425     439*
SECT.4  017160       429    1837*
SECT.5  020514       430    2174*
SECT.6  021424       431    2295*  2303
SEEKI   001432       271*   1250*  1257
SEEKO   00140        272*   1252*  1256
SEKSET  011554      1251    1255*
SHFCNT  020324      1318*   1320   1322*  2104*
SHF1    013574      1504    1508   1538*
SHF2    013572      1537*
SHF3    013570      1500    1512   1516   1536*
SIN   = 001000       292*   2006
SINST   020060      2007    2035*
SMTME   005600       734     761    867    889*   956    968   1169   1175   1263   1280   1630   1659   1675
                    1724    1731   1738   1764   1774   1787   1858   2090
STACK = 001100        22*    324   1000
START   001440       142     315*   552   1380   1465   1532   1720
STARTR  001434       139     314*
STFLG   001325       245*    367    370*
STKLMT= 177774        33*
STOP    013562      1521    1531*
STORE   004056       459     645*
STRT1   001760       371*    424
SUR     013554      1507*   1529*
SWR     001140       180*    322    334*   336    342*   355   1188   1193   1224   1232   1313   1320   2211
                    2216    2245   2254   2267   2575   2614   2661*  2869   2882*
SWREG   000176       137*    342    355   1188   1224   1313   2211   2575   2614   2637
SWTCH   021410      2216*   2217*  2232   2237   2247   2272*
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   013136        1459#   1477    1554
SYSERR  013652        1552    1554#
S10     012160        1321    1327#
S11     012136        1322#   1379
S12     012256        1344#
S13     012126        1320#   1326
TABLTY  002540         369     408#    420
TBITVE= 000014         119#
TBLFUL  004120         653     655#
TIME    005464         869     876#
TIMR    001346         257#    729*    737*    758*    766*    860*    876*    889*    890*
TIMR2   001323         243#    730*    768     770*    862*    879     881*
TKVEC = 000060         126#   2560*   2561*
TPVEC = 000064         127#
TRAPVE= 000034         125#    326*    327*
TRTVEC= 000014         120#
TRYAGN  011352        1224#   1241
TSTSN1  006140         954     961     963#
TYPE  = 104401         351     372     376     380     384     388     392     396     400     404     409     419     441
                       445     474     481     486     493     499     511     548     558     566     575     614     658
                       703     711     715     739     746     753     773     884     978     984     989    1018    1027
                      1032    1040    1046    1083    1087    1092    1100    1116    1143    1152    1180    1184    1205
                      1209    1213    1217    1237    1304    1309    1328    1329    1375    1460    1491    1520    1568
                      1573    1594    1602    1606    1610    1616    1622    1635    1639    1643    1648    1652    1664
                      1670    1680    1686    1691    1695    1699    1704    1708    1712    1716    1745    1755    1769
                      1779    1797    1802    1810    1816    1850    1853    1857    1885    1898    1891    1902    1905
                      1924    1928    1935    1938    1967    1970    1978    1981    2017    2027    2030    2038    2041
                      2058    2175    2179    2183    2187    2304    2337    2354    2374    2425    2521    2581    2635
                      2636    2639    2652    2663    2682    2730    2736    2741    2745    2750    2751    2753    2756
                      2760    2846#   2891
TYPOC = 104402        1331    1852    1887    1904    1926    1937    1969    1980    2022    2029    2040    2306    2379
                      2638    2847#
TYPON = 104404        1045    1051    1815    1821    2849#
TYPOS = 104403         572     619     708    1024    1037    1097    1105    1579    1599    1807    2848#
WATSK   017710        1976    1995#
WC      001412         277#   1345*   1387    1421
WCERR   013602        1452    1544#
WCERRR  013116        1448    1452#
WCHERR  013632        1545    1550#
```

```
WFERR   013122          1399    1456#
WPS   = 000040           291#   1889    1918
WRDCNT  001340           254#    794#    865     926#
WRITCS  001354           260#    795    1730    1763
WRLINK  005070           532     612     785#
WRPRO   016350          1621    1633    1753#
WRRDCK  016140          1615    1627    1721#   1749
WRTNBY  001317           239#    915#   1023
X       015476          1685    1690#
XX      015712          1703#
$AUTOB  001134           177#    359#   2628    2777
$BOADR  001122           172#
$ BOAT  001126           174#
$CALL   023670          2581    2770#
$CHARC  022306          2427#   2437#   2444    2453#   2458#
$CKSWR  023012          2614#   2853
$CMTAG  001100           160#    319     320
$CM3   = 000000           190#
$CNTLG  023701          2635    2772#
$CNTLU  023674          2652    2745    2771#
$CRLF   001161           191#   1328    1867    2426    2461    2663    2750    2770
$ENDAD  001400           149     270#
$ERFLG  001103           163#
$ERMAX  001115           169#
$ERRPC  001116           170#
$ERRTB  001434           307#
$ERTTL  001112           167#
$FILLC  001156           188#   2430    2461
$FILLS  001155           187#   2461
$GDADR  001120           171#
$GDDAT  001124           173#
$GTSWR  023102          2636#   2851
$HD    = 000003            11      12
$HIOCT  024030          2809#   2814#
$ICNT   001104           164#
$ILLUP  024252          2861    2877    2894#
$INTAG  001135           178#   2633#   2664    2777
$ITEMB  001114           168#
$LF     001162           192#   2461    2760    2770
$LPADR  001106           165#
$LPERR  001110           166#
$MAIL = ****** U         346     355    2414
$ MEM   023717          2639    2775#
$ ER    023706          2636    2773#
$NULL   001154           186#   2432    2461
$OCNT   022534          2493#   2522#   2535#
$OMODE  022536          2468#   2492#   2497    2500#   2511#   2537#
$PASS   001100           161#   1126#   1131
$POWER  024260          2892    2897#
$PWRON  024112           328    2861#   2889
$PWRMG  024246          2892#
$PWRUP  024164          2871    2877#
$QUES   001160           190#    419    2461    2682    2753    2770
$ROCHR  023314          2695#   2854
$RODEC= ****** U        2857
$ROLIN  023404          2718#   2855
```

```
$RDOCT  023730          2787#   2856
$RDSZ = 000030          2711#
$R2A  = ****** U        2857
$SAVR5= ****** U        2857
$SAVR6  024256          2870#   2878    2879#   2880#   2896#
$SETUP= 000114           314#    325     326     328     330     351     352    2573    2578    2579    2609    2777
$STUP = 177777           314#
$SVPC = 000222           147#    152
$SWR  = 160000            11      12#     190    2893
STKB    001146           183#   2541    2562    2571    2590    2618    2645
STKCNT  0[illegible]540  2542#   2557#   2579    2596#   2702    2704#
STKINT  0  50             346    2557#   2631
STKQEN= 0  547           2546#   2604    2707
STKQIN  0  542           2543#   2558#   2559    2602#   2603#   2604    2606#
STKQOU  022544           2544#   2559#   2705    2706#   2707    2709#
STKQSR  022546           2545#   2558    2606    2709
STKS    001144           182#   2541    2563#   2586#   2588    2594#   2616    2632#   2642    2666#
STKSRV  022620           2560    2571#
STN   = 000001            11#
STPB    001152           185#   2450#   2461
STPFLG  001157           189#   2408    2461
STPS    001150           184#   2448    2461
STRAP   024032           326    2823#
STRAP2  024054          2834#   2845
STRP  = 000012          2838#   2847#   2848#   2849#   2850#   2851    2852#   2853    2854#   2855#   2856#   2857#
STRPAD  024066          2828    2845#
STSTNM  001102           162#
STTYIN  023640          2720    2721    2733    2751    2765    2769#
STYPBN= ****** U        2850
STYPOS= ****** U        2850
STYPE   022072          2408#   2838    2846
STYPEC  022242          2429    2436    2443    2448#   2449    2668
STYPEX  022310          2454    2456    2459#
STYPOC  022336          2491#   2847
STYPON  022352          2490    2493#   2849
STYPOS  022312          2486#   2848
$OFILL  022535          2487*   2491*   2501    2536#
.     = 024270           131#    135#    140#    147     148#    150#    152#    159#    193     196#    207#    208#    226#
                         248#    323     363#    375#    379#    383#    387#    391#    395#    399#    403#    407#    444#
                         448#    561#    569#    617#    714#    742#    887#   1030#   1049#   1095#   1134#   1155#   1231
                        1240#   1312#   1463#   1494#   15[illegible]#   1559#   1597#   1613#   1638#   1646#   1651#   1655#   1702#
                        1711#   1748#   1758#   1772#   18[illegible]5#   1819#   1894#   2020#   2144#   2186#   2252    2293#   2309
                        2322    2340#   2348    2357#   2359    2362    2370    2377#   2461    2541    2545#   2546    2547#
                        2769#   2770    2777    2873    2895
```

```
COMMEN    129#
ENDCOM    129#
ERROR      23#
ESCAPE    129#
GETPRI    129#
GETSWR      1#    129#    352#
MORETA    153#    194
MULT      129#
NEWTST    129#
POP       129#   2810    2882    2883
PUSH      129#   2789    2863    2869
REPORT    129#
SCOPE      24#
SETPRI    129#   2698
SETTRA   2833#   2847    2848    2849    2851    2853    2854    2855    2856
SETUP     129#    318
SKIP      129#
SLASH     129#
SPACE     129#
STARS     129#    145     155     193     308     312    2393    2463    2540    2609    2624    2687    2711    2779    2817
         2859    2875
SWRSU     129#    330#
TRMTRP   2838#
TYPBIN    129#
TYPDEC    129#
TYPNAM      1#    129#    347
TYPNUM    129#
TYPOCS    129#    707    1598
TYPOCT    129#   2637
TYPTXT    129#    372     376     380     384     388     392     396     400     404     408     440     445     473     486
          493     510     547     557     565     575     614     658     703     711     715     739     746     753     772
          884     978     983     988    1018    1027    1031    1040    1046    1083    1087    1092    1100    1115    1142
         1151    1180    1184    1205    1209    1213    1217    1237    1303    1308    1374    1459    1490    1567    1572
         1594    1602    1606    1610    1616    1622    1634    1639    1643    1648    1652    1664    1670    1680    1686
         1690    1695    1699    1704    1708    1712    1716    1745    1755    1769    1779    1797    1802    1810    1816
         1891    2016    2058    2175    2179    2183    2337    2354    2374
SSCMRE    153#
SSCMTM    153#
SSESCA    129#
SSNEWT    129#
SSSET    2838#   2847    2848    2849    2851    2853    2854    2855    2856
SSSKIP    129#
.EQUAT      1#     19
.HEADE      1#
.SETUP      1#    314
.$ACT1      1#    143
.$CATC      1#    129
.$CMTA      1#    153
.$POWE      1#   2857
.$RDOC      1#   2777
.$READ      1#   2538
.$TRAP      1#   2815
.$TYPE      1#   2391
.$TYPO      1#   2461
```

```
MAINDEC-11-DZRKI-E      MACY11 27(1006)  17-MAR-77  14:57  PAGE 67
DZRKIE.P11    17-MAR-77 14:53           CROSS REFERENCE TABLE -- MACRO NAMES

. ABS.  024270     000


 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DSKZ:DZRKIE.BIN,DSKZ:DZRKIE.LST/CRF/SOL=DSKM:DZRKIE.P11
 RUN-TIME:  13  8  .9 SECONDS
 RUN-TIME RATIO:  82/22=3.5
 CORE USED:  20K  (40 PAGES)
```