

RK05/RK11

UTILITY PACKAGE
MD-11-DZRKI-E

EP-DZRKI-E-DL-C
COPYRIGHT © 1977
FICHE 1 OF 1

JUN 1977
digital
MADE IN USA

157

157

801

EOF1DZBNDJSEQ

00010000

770610

PDP10 411

-THOR1DZRKIESEQ

00010000

770610

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

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

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

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

COPYRIGHT (C) 1974, 1977 BY DIGITAL EQUIPMENT CORPORATION

001

E01

TABLE OF CONTENTS

1.0	ABSTRACT
2.0	REQUIREMENTS
2.1	EQUIPMENT
2.2	STORAGE
2.3	PRELIMINARY PROGRAMS
3.0	LOADING PROCEDURE
4.0	STARTING PROCEDURE
5.0	OPERATING PROCEDURE
6.0	ERRORS
7.0	RESTRICTIONS
8.0	EXECUTION TIME
9.0	PROGRAM DESCRIPTION
9.1	PROGRAM INDEX
9.2	COMPATIBILITY PACKAGE
9.3	OSCILLATING SEEK PACKAGE
9.4	FORMATTER SURFACE VERIFIER
9.5	RK05 CONTROL PANEL TEST
9.6	RK05 CONTROL PANEL TEST # 2
9.7	HEAD ALIGNMENT ROUTINE
9.8	(DISK) POWER FAILURE TEST
9.9	SECTION SPECIAL
9.10	COMPATIBILITY ERROR RECOVERY

F01

G01

1. ABSTRACT

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

2. REQUIREMENTS

2.1 EQUIPMENT
PDP-11 PROCESSOR
8K MEMORY
RK11 OR RKV11 CONTROLLER
1-8 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 SETTINGS 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 ACTION
SEE INDIVIDUAL PACKAGE DESCRIPTION (PARAGRAPH 9)

6. ERRORS

- 6.1 ERROR HALTS AND DESCRIPTION
IF HALTED A MAJOR PROBLEM EXISTS 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 COMPATIBILITY 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

I01

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.

DESCRIPTION: THIS PACKAGE DOES NOT APPLY TO RK-05F DRIVES. 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 NUMBER) MAY BE ON SYSTEM TWO. COMPATIBILITY-A DEFINITION, COMPATIBILITY INFERS MORE THAN THE FACT THAT INFORMATION WHICH WAS WRITTEN ON ONE DRIVE CAN BE

J01

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.
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.....

USE:

EXAMPLE 1

NAME	TYPE
INDEX	0
COMPATIBILITY 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=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 RXXX 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 **K01**
 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 INCOMPATIBILITY.

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 PLES, 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 MUST FOLLOW DIRECTIONS.
ERROR INFO: SEE PARAGRAPH 9.6 SPECIAL SECTION

9.3 SECTION 2 OSCILLATING SEEK PACKAGE

PURPOSE: TO ALLOW THE USER TO MAKE SERVO ADJUSTMENTS
AND/OR SEEK LOGIC CHECKOUT BY PERFORMING
SEEKS BETWEEN USER SPECIFIED ADDRESS

DESCRIPTION: SELECT TYPE 2, THE USER THEN INSERTS
THE DRIVES TO BE TESTED IN SW0 TO SW7
OF THE SWITCH REGISTER. A SWITCH IS SET
FOR EACH DRIVE (E.G. SW2 TO TEST DRIVE 2.
THE USER THEN INSERTS THE
ADDRESS TO SEEK IN THE SWR. IF BOTH ADDRESS
ARE LEGAL, 50 CYCLES (100 SEEKS) WILL BE
MADE BETWEEN THE SPECIFIED ADDRESS THEN
THE PROGRAM WILL LOOK AT THE SWR FOR
POSSIBLE CHANGES THIS SHOULD ALLOW FOR GOOD
STABLE TRACES ON AN OSCILLOSCOPE.

IT SHOULD BE NOTED THAT THE OSCILLATING MO1
 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

ERROR INFO: IF AN ILLEGAL ADDRESS IS SELECTED A MESSAGE IS TYPED
 AND USER NEARLY 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-DSF 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

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

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

PACK GOOD.

RK11 UTILITY PACKAGE

INDEX	NAME	TYPE
	COMPATIBILITY PACKAGE	0
		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 PF EM, 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....

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

INDEX	NAME	TYPE
	COMPATABILITY PACKAGE	0
		1

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

TYPE=

9.6 SECTION 5 RK05 CONTROL PANEL TEST #2

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

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

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

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

THE PROGRAM, THEN STARTS SCANNING ALL DRIVES, ONE AFTER THE OTHER. CHECKS IF THE DRIVE IS ON LINE OR OFF LINE (RDY 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 DRIVE IS 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 AFFECTED⁰⁰²
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 POSITIONED 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 **DD2**
DRIVE:

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

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

ERROR, ON POWER-UP, RKDA=XXXX
XXXX IS THE CONTENTS OF RKDA AT THE TIME OF
ERROR.

THE PROGRAM DOES THE ABOVE POWER FAIL TEST
ON ALL DRIVES THAT ARE PRESENT, ONE AFTER THE
OTHER IN A ROUND BOBBIN FASHION. EXIT IS THROUGH
HALT.

9.9 SECTION SPECIAL

FOR THE BELOW EXAMPLES THE FOLLOWING FORMAT
WILL BE USED.

THE ACTUAL TYPEOUT : COMMENTS ON WHAT
AND RESPONSE : OCCURRED OR WHAT TO DO
#NOTES IF NECESSARY FOR CLARITY

ERROR EXAMPLE 1

FORMATTER-SURFACE VERIFIER	3	
RK05 CONTROL PANEL TEST	4	
TYPE=1		:TYPE 1 SELECTION
DRIVE NUMBERS ON SYTEM 1=0.		:DRIVE #0 SELECTED
IS THERE A SECOND SYSTEM?N		:NO SECOND SYSTEM
MOUNT PACK ON DRIVE #0		:CONTINUE PRESSED BUT
MAKE PACK WRITE ENABLE		:WRITE PROTECT ON
PRESS CONTINUE WHEN DRIVE RDY		:CLEAR WRITE PROTECT SWITCH
DRIVE WRITE PROTECTED.		:NOW RUNS TO FINISH
DRIVE WRITE PROTECTED		:THIS DOES NOT EFFECT
DONE!		:OUTCOME OF TEST

RK11 UTILITY PACKAGE

NAME	TYPE
INDEX	0
COMPATIBILITY PACKAGE	1
OSCILLATING SEEK PACKAGE	2

ERROR EXAMPLE 2

RK11 UTILITY PACKAGE

NAME	TYPE
INDEX	0
COMPATIBILITY PACKAGE	1
OSCILLATING SEEK PACKAGE	2
FORMATTER-SURFACE VERIFIER	3

F02

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

ERROR EXAMPLE 4

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

TYPE=1
DRIVE NUMBERS ON SYTEM 1=0.

IS THERE A SECOND SYSTEM?N ;SAME AS ABOVE BUT FUNCTION
MOUNT PACK ON DRIVE #0 ;WAS A CONTROL RESET
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE R0Y ;ALL COMMENTS ARE THE SAME
CONTROL RESET TIMED OUT ;AS EXAMPLE 3
CONTROL RESET TIMED OUT
CONTROL RESET TIMED OUT
CONTROL RESET TIMED OUT

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

ERROR EXAMPLE 5

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

RK11 UTILITY PACKAGE

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

TYPE=1
DRIVE NUMBERS ON SYTEM 1=0.

IS THERE A SECOND SYSTEM?N
MOUNT PACK ON DRIVE #0
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE R0Y
5 ERRORS OCCURRED DRIVE DECLARED DOWN!! NOT TESTED !
DONE!

RK11 UTILITY PACKAGE

NAME TYPE
INDEX 0

COMPATIBILITY PACKAGE

1

G02

*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
MAKE PACK WRITE ENABLE
PRESS CONTIN WHEN DRIVE RDY
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=00007  EXPCTD=077400  RECVD=077600
  ADDR=00007  EXPCTD=077400  RECVD=037600
  ADDR=00007  EXPCTD=077400  RECVD=037600
  ADDR=00007  EXPCTD=077400  RECVD=037600
  ADDR=00007  EXPCTD=077400  RECVD=037600
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=00004  EXPCTD=077400  RECVD=077600
  ADDR=00004  EXPCTD=077400  RECVD=037600
  ADDR=00004  EXPCTD=077400  RECVD=037600
  ADDR=00004  EXPCTD=077400  RECVD=037600
  ADDR=00004  EXPCTD=077400  RECVD=037600
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=00064  EXPCTD=077400  RECVD=077600
  ADDR=00064  EXPCTD=077400  RECVD=037600
  ADDR=00064  EXPCTD=077400  RECVD=037600
  ADDR=00064  EXPCTD=077400  RECVD=037600
  ADDR=00064  EXPCTD=077400  RECVD=037600
ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
  ADDR=002767 EXPCTD=077400  RECVD=177000
5 ERRORS OCCURRED DRIVE DECLARED DOWN!! NOT TESTED!
DONE!

```

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

9.10 COMPATIBILITY ERROR RECOVERY

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

1. THERE ARE TWO STRATEGICALLY 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 FWER REGISTER BEFORE THE PROGRAM CORRECTS ANY ERRORS WHICH WHICH MAY HAVE OCCURRED.
2. IF PLACED BY CHECKSUM ERRORS AND THE USER WISHES MORE ER OR MAPING THEN HE MAY MODIFY THE MASK WORD AT LOCATION 'ERRCHK+2' TO ONLY RECOGNIZE HARD ERRORS.

- 102
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

K02

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

22	BASIC DEFINITIONS
132	TRAP CATCHER
141	STARTING ADDRESS(ES)
146	ACT11 HOOKS
156	COMMON TAGS
296	ERROR POINTER TABLE
321	INITIALIZE THE COMMON TAGS
350	TYPE PROGRAM NAME
355	GET VALUE FOR SOFTWARE SWITCH REGISTER
437	COMPATIBILITY TEST
1152	OSCILLATING SEEK ROUTINE
1304	FORMATTER-SURFACE VERIFIER
1568	RK05 CONTROL PANEL TEST
1834	CONTROL PANEL TEST # 2
2158	HEAD ALIGNMENT ROUTINE
2281	DISK POWER FAILURE TEST
2394	TYPE ROUTINE
2464	BINARY TO OCTAL (ASCII) AND TYPE
2541	TTY INPUT ROUTINE
2781	READ AN OCTAL NUMBER FROM THE TTY
2819	TRAP DECODER
2842	TRAP TABLE
2861	POWER DOWN AND UP ROUTINES

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56

```

.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.
*
$TN=1
$SWR=160000      ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYP0UT

;#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 ***
STACK= 1100
.EQUIV EMT,ERROR      ;;BASIC DEFINITION OF ERROR CALL
.EQUIV IOT,SCOPE      ;;BASIC DEFINITION OF SCOPE CALL

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

;#GENERAL PURPOSE REGISTER DEFINITIONS
R0= %0                ;;GENERAL REGISTER
R1= %1                ;;GENERAL REGISTER
R2= %2                ;;GENERAL REGISTER
R3= %3                ;;GENERAL REGISTER
R4= %4                ;;GENERAL REGISTER
R5= %5                ;;GENERAL REGISTER
R6= %6                ;;GENERAL REGISTER
R7= %7                ;;GENERAL REGISTER
SP= %6                ;;STACK POINTER
PC= %7                ;;PROGRAM COUNTER

;#PRIORITY LEVEL DEFINITIONS
PR0= 0                ;;PRIORITY LEVEL 0
PR1= 40               ;;PRIORITY LEVEL 1
PR2= 100              ;;PRIORITY LEVEL 2
PR3= 140              ;;PRIORITY LEVEL 3
PR4= 200              ;;PRIORITY LEVEL 4
PR5= 240              ;;PRIORITY LEVEL 5

000001
160000

001100

000011
000012
000015
000200
177776

177774
177772
177570
177570

000000
000001
000002
000003
000004
000005
000006
000007
000006
000007

000000
000040
000100
000140
000200
000240

```



```

113 .EQUIV BIT01,BIT1
114 .EQUIV BIT00,BIT0
115
116 ;#BASIC "CPU" TRAP VECTOR ADDRESSES
117 000004 ERAVEC= 4 ;TIME OUT AND OTHER ERRORS
118 000010 RESVEC= 10 ;RESERVED AND ILLEGAL INSTRUCTIONS
119 000014 TBITVEC=14 ;"T" BIT
120 000014 TRIVEC= 14 ;TRACE TRAP
121 000014 BPTVEC= 14 ;BREAKPOINT TRAP (BPT)
122 000020 IOTVEC= 20 ;INPUT/OUTPUT TRAP (IOT) **SCOPE**
123 000024 PWRVEC= 24 ;POWER FAIL
124 000030 EMTVEC= 30 ;EMULATOR TRAP (EMT) **ERROR**
125 000034 TRAPVEC=34 ;"TRAP" TRAP
126 000060 TKVEC= 60 ;TTY KEYBOARD VECTOR
127 000064 TPVEC= 64 ;TTY PRINTER VECTOR
128 000240 PIRAVEC=240 ;PROGRAM INTERRUPT REQUEST VECTOR
129 .SBTTL TRAP CATCHER
130
131 000000 .=0
132 ;#ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,MALT"
133 ;#SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
134 ;#LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
135 .=174
136 000174 000000 DISPREG: .WORD 0 ;SOFTWARE DISPLAY REGISTER
137 000176 000000 SWREG: .WORD 0 ;SOFTWARE SWITCH REGISTER
138 .SBTTL STARTING ADDRESS(ES)
139 000200 000137 001434 JMP @#STARTR ;JUMP TO STARTING ADDRESS OF PROGRAM
140 .=210
141 000210 112737 000377 001312 MOVB #377,@#MODE
142 000216 000137 001440 JMP @#START
143 .SBTTL ACT11 HOOKS
144
145 ;*****
146 ;HOOKS REQUIRED BY ACT11
147 000222 $SVPC=. ;SAVE PC
148 000046 .=46
149 000046 001400 SENDAD ;1)SET LOC.46 TO ADDRESS OF SENDAD IN .SEOP
150 000052 .=52
151 000052 000000 .WORD 0 ;2)SET LOC.52 TO ZERO
152 000222 .=$SVPC ;RESTORE PC

```

.SBTTL COMMON TAGS

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

153									
154									
155									
156									
157									
158									
159		001100							
160	001100								
161	001100	000000							
162	001102	000							
163	001103	000							
164	001104	000000							
165	001106	000000							
166	001110	000000							
167	001112	000000							
168	001114	000							
169	001115	001							
170	001116	000000							
171	001120	000000							
172	001122	000000							
173	001124	000000							
174	001126	000000							
175	001130	000000							
176	001132	000000							
177	001134	000							
178	001135	000							
179	001136	000000							
180	001140	177570							
181	001142	177570							
182	001144	177560							
183	001146	177562							
184	001150	177564							
185	001152	177566							
186	001154	000							
187	001155	002							
188	001156	012							
189	001157	000							
190	001160	077							
191	001161	015							
192	001162	000012							
193									
194	001164	000000							
195									
196	001166	000010							
197									
198	001206	152525							
199	001210	077777							
200	001212	000000							
201	001214	012345							
202	001216	125252							
203	001220	000001							
204	001222	177777							
205	001224	154320							
206									
207	001226	000010							
208	001246	000004							

```

.=1100
SCMTAG: .WORD 0 ; START OF COMMON TAGS
$PASS: .WORD 0 ; CONTAINS PASS COUNT
$STNM: .BYTE 0 ; CONTAINS THE TEST NUMBER
$ERFLG: .BYTE 0 ; CONTAINS ERROR FLAG
$ICNT: .WORD 0 ; CONTAINS SUBTEST ITERATION COUNT
$LPADR: .WORD 0 ; CONTAINS SCOPE LOOP ADDRESS
$LPERR: .WORD 0 ; CONTAINS SCOPE RETURN FOR ERRORS
$ERTTL: .WORD 0 ; CONTAINS TOTAL ERRORS DETECTED
$ITEMB: .BYTE 0 ; CONTAINS ITEM CONTROL BYTE
$ERMAX: .BYTE 1 ; CONTAINS MAX. ERRORS PER TEST
$ERRPC: .WORD 0 ; CONTAINS PC OF LAST ERROR INSTRUCTION
$GDADR: .WORD 0 ; CONTAINS ADDRESS OF 'GOOD' DATA
$BDADR: .WORD 0 ; CONTAINS ADDRESS OF 'BAD' DATA
$GDAT: .WORD 0 ; CONTAINS 'GOOD' DATA
$BDAT: .WORD 0 ; CONTAINS 'BAD' DATA
;RESERVED--NOT TO BE USED

$AUTOB: .BYTE 0 ; AUTOMATIC MODE INDICATOR
$INTAG: .BYTE 0 ; INTERRUPT MODE INDICATOR

$SWR: .WORD DSWR ; ADDRESS OF SWITCH REGISTER
$DISPLAY: .WORD DOISP ; ADDRESS OF DISPLAY REGISTER
$TKS: 177560 ; TTY KBD STATUS
$TKB: 177562 ; TTY KBD BUFFER
$TPS: 177564 ; TTY PRINTER STATUS REG. ADDRESS
$TPB: 177566 ; TTY PRINTER BUFFER REG. ADDRESS
$NULL: .BYTE 0 ; CONTAINS NULL CHARACTER FOR FILLS
$FILLS: .BYTE 2 ; CONTAINS # OF FILLER CHARACTERS REQUIRED
$FILLC: .BYTE 12 ; INSERT FILL CHARS. AFTER A "LINE FEED"
$TPFLG: .BYTE 0 ; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
$QUES: .ASCII /?/ ; QUESTION MARK
$CRLF: .ASCII <15> ; CARRIAGE RETURN
$LF: .ASCII <12> ; LINE FEED
*****
DRACTV: .WORD 0 ; ACTIVE DRIVE WORD

LOGA: .BLKW 10 ; TABLE OF ACTIVE DRIVE WORDS

DRVO: .WORD 152525 ; TABLE OF PATTERN = TO DRIVE #'S
.WORD 077777
.WORD 000000
.WORD 012345
.WORD 125252
.WORD 000001
.WORD 177777
.WORD 154320

RDTBL: .BLKW 10 ; TABLE OF READ ADDRESS
PASTBL: .BLKW 4 ; TABLE OF PARAMETERS FOR SYSTEM #2

```

209						
210	001256	377	MSKTBL:	.BYTE	377	;TABLE OF CYLINDER BASE FOR AUTO MODE
211	001257	177		.BYTE	177	
212	001260	077		.BYTE	077	
213	001261	037		.BYTE	037	
214	001262	017		.BYTE	017	
215	001263	007		.BYTE	007	
216	001264	003		.BYTE	003	
217	001265	001		.BYTE	001	
218						
219	001266	000	BASE:	.BYTE	0	;CYL 0 BASE CYLINDER ADDRESS
220	001267	050		.BYTE	50	;CYL 40 BASE CYLINDER ADDRESS
221	001270	120		.BYTE	120	;CYL 80 BASE CYLINDER ADDRESS
222	001271	170		.BYTE	170	;CYL 120 BASE CYLINDER ADDRESS
223	001272	240		.BYTE	240	;CYL 160 BASE CYLINDER ADDRESS
224	001273	303		.BYTE	303	;CYL 195 BASE CYLINDER ADDRESS
225						
226	001274	000011	CYLTBL:	.BLKB	11	;TABLE OF SELECTED BASES
227						
228	001305	000	SECTBL:	.BYTE	0	;SECTOR 0
229	001306	004		.BYTE	4	;SECTOR 4
230	001307	007		.BYTE	7	;SECTOR 7
231	001310	013		.BYTE	13	;SECTOR 12
232						
233	001311	000	DRCNT1:	.BYTE	0	;COUNT OF NUMBER OF DRIVES ON SYS. 1
234	001312	000	MODE:	.BYTE	0	;IF -1 START 210 SELECTED
235	001313	000	PROMUM:	.BYTE	0	;IF 0 1 PROCESSOR SELECTED
236	001314	000	DRIVE:	.BYTE	0	;DRIVE # UP _R TEST (MAN+AUTO MODE)
237	001315	000	CYLBAS:	.BYTE	0	;BASE SELECTED (MANUAL MODE)
238	001316	000	COMND:	.BYTE	0	;IF 0 WRITE COMMAND
239	001317	000	WRTNBY:	.BYTE	0	;DRIVE WHICH DID WRITE (READ OPERATION)
240	001320	000	HDRFLG:	.BYTE	0	;FLAG FOR ONE HEADER PRINTOUT
241	001321	000	ECNT:	.BYTE	0	;ERROR COUNTER
242	001322	000	CNTSIN:	.BYTE	0	;SEEK INCOM. COUNTER
243	001323	000	TMR2:	.BYTE	0	;SECOND PASS TIMER
244	001324	000	IDEX:	.BYTE	0	;CURRENT INDEX #
245	001325	000	STFLG:	.BYTE	0	
246	001326	000	OSPFLG:	.BYTE	0	
247						
248		001330		.EVEN		
249						
250	001330	000000	KYTEMP:	.WORD	0	;TEMP. KEYBOARD BUFFER
251	001332	000000	CONTRL:	.WORD	0	;TEMP. CONTROL+STATUS WORD
252	001334	000000	DSKADR:	.WORD	0	;TEMP. DISK ADDRESS WORD
253	001336	000000	BUSADR:	.WORD	0	;TEMP. BUS ADDRESS WORD
254	001340	000000	WRDCNT:	.WORD	0	;TEMP. WORD COUNT
255	001342	172000	CYLCNT:	.WORD	-6000	;WORD COUNT OF 1 CYLINDER
256	001344	177400	SECCNT:	.WORD	-400	;WORD COUNT OF 1 SECTOR
257	001346	000000	TMR:	.WORD	0	;TIMER FOR OPERATIONS
258	001350	000000	CHKCNT:	.WORD	0	;NUMBER OF ERROR PRINTOUTS
259	001352	000000	DSKTMP:	.WORD	0	;SAVE OF CURRENT DISK #
260	001354	004003	WRITCS:	.WORD	4003	;IBA+WRITE+GO
261	001356	000005	READCS:	.WORD	5	;READ+GO
262	001360	000000	ERRFLG:	.WORD	0	;ERROR FLAG INHIBIT ADDRESS CHANGE
263	001362	000000	PATRN:	.WORD	0	;DATA PATTERN
264	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=	BITS	
292		001000	SIN=	BIT9	

29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113

001434

```

.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

SERRTB:
; ;*****
; ;
; THE ERROR TABLE IS UNUSED IN THIS PROGRAM
; ;*****

```

```

314 001434 105037 001312 STARTR: CLRB @MODE
315 001440 000005 START: RESET ; CLEAR THE BUS
316 001442 005037 177776 CLR PS ; LOWER PROCESSOR PRIORITY TO
; ALLOW TTY KYBD INTERRUPTS
317
318 .SBTTL INITIALIZE THE COMMON TAGS
319 ;; CLEAR THE COMMON TAGS (SCMTAG) AREA
320 001446 012706 001100 MOV @SCMTAG,R6 ; FIRST LOCATION TO BE CLEARED
321 00 32 005026 CLR (R6)+ ; CLEAR MEMORY LOCATION
322 001454 022706 001140 CMP @SWR,R6 ;; DONE?
323 001460 001374 BNE -6 ; LOOP BACK IF NO
324 001462 012706 001100 MOV @STACK,SP ; SETUP THE STACK POINTER
325 ;; INITIALIZE A FEW VECTORS
326 001466 012737 024032 000034 MOV @STRAP,@TRAPVEC ; TRAP VECTOR FOR TRAP CALLS
327 001474 012737 000340 000036 MOV @340,@TRAPVEC+2 ; LEVEL 7
328 001502 012737 024112 000024 MOV @SPWRON,@SWR.VEC ; POWER FAILURE VECTOR
329 001510 012737 000340 000026 MOV @340,@SWRVEC+2 ; LEVEL 7
330 ;; SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
331 ;; EQUAL TO A "-1" SETUP FOR A SOFTWARE SWITCH REGISTER.
332 001516 013746 000004 MOV @ERRVEC,-(SP) ; SAVE ERROR VECTOR
333 001522 012737 001556 000004 MOV @64S,@ERRVEC ; SET UP ERROR VECTOR
334 001530 012737 177570 001140 MOV @DSWR,SWR ; SETUP FOR A HARDWARE SWICH REGISTER
335 001536 012737 177570 001142 MOV @DISP,DISPLAY ; AND A HARDWARE DISPLAY REGISTER
336 001544 022777 177777 177366 CMP #-1,@SWR ; TRY TO REFERENCE HARDWARE SWR
337 001552 001012 BNE 66S ; BRANCH IF NO TIMEOUT TRAP OCCURRED
338 ; AND THE HARDWARE SWR IS NOT = -1
339 001554 000403 BR 65S ; BRANCH IF NO TIMEOUT
340 001556 012716 001564 64S: MOV @65S,(SP) ; SET UP FOR TRAP RETURN
341 001562 000002 RTI
342 001564 012737 000176 001140 65S: MOV @SWREG,SWR ; POINT TO SOFTWARE SWR
343 001572 012737 000174 001142 MOV @DISPREG,DISPLAY
344 001600 012637 000004 66S: MOV (SP)+,@ERRVEC ; RESTORE ERROR VECTOR
345
346 001604 004737 022550 JSR PC,STKINT ; INITIALIZE THE TTY INTERRUPT HANDLER
347
348 .SBTTL TYPE PROGRAM NAME
349 ;; TYPE THE NAME OF THE PROGRAM IF FIRST PASS
350 001610 005227 177777 INC #-1 ; FIRST TIME?
351 001614 001045 BNE 67S ; BRANCH IF NO
352 001616 104401 001654 TYPE 68S ; TYPE ASCIZ STRING
353 .SBTTL GET VALUE FOR SOFTWARE SWITCH REGISTER
354 001622 005737 000042 TST @42 ; ARE WE RUNNING UNDER XXDP/ACT?
355 001626 001006 BNE 69S ; BRANCH IF YES
356 001630 023727 001140 000176 CMP SWR,@SWREG ; SOFTWARE SWITCH REG SELECTED?
357 001636 001005 BNE 70S ; BRANCH IF NO
358 001640 104405 GTSWR ; GET SOFT-SWR SETTINGS
359 001642 000403 BR 70S
360 001644 112737 000001 001134 69S: MOV @1,SAUTOB ; SET AUTO-MODE INDICATOR
361 001652 000426 70S: BR 67S ; GET OVER THE ASCIZ
362 ;; 68S: .ASCIZ <CRLF>/RK11 UTILITY PACKAGE/<15><12>/MAINDEC-11-DZRKI-E/<CRLF>
363 001730 67S:
364 001730 105737 001312 TSTB @MODE
365 001734 100002 BPL 1S
366 001736 000137 003436 JMP @SECOND
367 001742 105737 001325 1S: TSTB STFLG ; PRINT ONLY THE FIRST TIME
368 001746 001402 BEQ 10S
369 001750 000137 002540 JMP TABLTY

```

H03

MAINDEC-11-DZRKI-E MACY11 27(1006) 17-MAR-77 14:57 PAGE 9
 DZRKIE.P11 17-MAR-77 14:53 GET VALUE FOR SOFTWARE SWITCH REGISTER

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

;TEST ENTRY DISPATCH TABLE

426 002612 010354
 427 002614 011766
 428 002616 013750
 429 002620 017160
 430 002622 020514
 431 002624 021424

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.

438 002626 000240
 439 002630
 440 002630 104401 002636
 441 002634 000415
 442
 443 002670
 444 002670 104401 002676
 445 002674 000417
 446
 447 002734
 448 002734 104410
 449 002736 012600
 450 002740 012701 001166
 451 002744 105037 001311
 452 002750 005037 001330
 453 002754 112037 001330
 454 002760 122737 000054 001330
 455 002766 001770
 456 002770 162737 000060 001330
 457 002776 100403
 458 003000 004737 004056
 459 003004 000761
 460 003006 122740 000056
 461 003012 001402
 462 003014 004737 004126
 463 003020 022701 001206
 464 003024 001403
 465 003026 012721 100000
 466 003032 000772

SECT.3: NOP ; NO-OP
 AUTSL2:
 TYPE 65\$; TYPE ASCIZ STRING
 BR 64\$; GET OVER THE ASCIZ
 ; 65\$: .ASCIZ <15><12>/TERMINATE WITH '<CR>'/
 64\$:
 TYPE 67\$; TYPE ASCIZ STRING
 BR 66\$; GET OVER THE ASCIZ
 ; 67\$: .ASCIZ <15><12>/DRIVE NUMBERS ON SYSTEM 1=
 66\$:
 RDLIN
 MOV (SP)+,R0 ; PICK UP THE ADDRESS OF THE INPUT BUFFER
 MOV #LOGA,R1 ; GET THE ADDRESS OF THE LOGICAL UNIT TBL.
 CLRB @#DRCNT1 ; CLEAR THE DRIVE COUNTER
 CLR @#KYTEMP ; CLEAR TEMP
 1\$: MOVB (R0)+,@#KYTEMP ; GET THE FIRST DRIVE #
 CMPB #54,@#KYTEMP ; IS IT A COMMA THAT WAS TYPED?
 BEQ 1\$; IF YES GO BACK
 SUB #60,@#KYTEMP ; MAKE ASCII A DRIVE #
 BMI 2\$; IF RESULT NEGATIVE BRANCH
 JSR PC,STORE ; IF RESULT POSITIVE JUMP
 BR 1\$; AFTER STORING GET NEXT #
 2\$: CMPB #56,-(R0) ; WAS NEGATIVE RESULT A TERMINATOR?
 BEQ 3\$; IF YES BRANCH
 JSR PC,ILEGAL ; IF NO BAD CHARACTER JUMP
 3\$: CMP #0AVD,R1 ; IS THE TABLE FULL
 BEQ SECSYS ; IF YES BRANCH
 MOV #100000,(R1)+ ; IF NO FILL TABLE WITH DOWN INDICATOR.
 BR 3\$; GO BACK AND CHECK

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

473 003034
 474 003034 104401 003042
 475 003040 000416
 476
 477 003076
 478 003076 000240
 479 003100 104407
 480 003102 012637 001330
 481 003106 104401 001330

SECSYS:
 TYPE 65\$; TYPE ASCIZ STRING
 BR 64\$; GET OVER THE ASCIZ
 ; 65\$: .ASCIZ <15><12>/IS THERE A SECOND SYSTEM?/
 64\$:
 NOP ; ***
 RDCHR ; READ A CHARACTER
 MOV (SP)+,@#KYTEMP ; GET THE RESPONSE
 TYPE ,KYTEMP ; ECHO

```

482 003112 022737 000131 001330      CMP      #131,@#KYTEMP      ; WAS IT A "Y" (FOR YES)?
483 003120 001411                    BEQ      PR02              ; IF YES BRANCH TO PROCESSOR 2
484 003122 022737 000116 001330      CMP      #116,@#KYTEMP      ; WAS RESPONSE LEGAL (N FOR NO)?
485 003130 001460                    BEQ      PR01              ; IF LEGAL BRANCH
486 003132 104401 003140      TYPE    ,67$              ; TYPE ASCIZ STRING
487 003136 000401                    BR       66$              ; GET OVER THE ASCIZ
488                                     ;:67$: .ASCIZ  /?/
489                                     66$:
490 003142 000734                    BR       SECSYS           ; GO BACK ASK AGAIN
491 003144 152737 000377 001313 PR02: 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$: .ASCIZ <15><12>/DRIVE # =/
496                                     64$:
497 003176 104407                    RDCHR
498 003200 012637 001330      MOV      (SP)+,@#KYTEMP     ; READ A CHARACTER
499 003204 104401 001330      TYPE    ,KYTEMP           ; PICK UP THE RESPONSE
500 003210 162737 000060 001330      SUB      #60,@#KYTEMP      ; ECHO
501 003216 100420                    BMI     BADINP            ; MAKE IT A NUMBER
502 003230 022737 000010 001330      CMP      #10,@#KYTEMP      ; IF NOT A NUMBER, BRANCH
503 003236 003414                    BLE     BADINP            ; IS IT A LEGAL #?
504 003230 000337 001330      SWAB    @#KYTEMP          ; IF NO BRANCH
505 003234 052737 040000 001330      BIS     #BIT14,@#KYTEMP    ; GET THE DRIVE # TO THE HIGH BYTE
506 003242 113705 001311      MOV     @#ORCNT1,R5       ; SET THE SECOND SYSTEM BIT
507 003246 006105                    ROL     R5                ; GET THE DRIVE COUNT
508 003250 013765 001330 001166      MOV     @#KYTEMP,LOGA(R5)  ; MAKE IT AN INDEX
509 003256 000407                    BR      GO                ; STORE THE SYSTEM #2 WORD
510                                     ; GO DO THE TEST
511                                     BADINP:
512 003260 104401 003266      TYPE    ,65$              ; TYPE ASCIZ STRING
513 003264 000.J1                    BR       64$              ; GET OVER THE ASCIZ
514                                     ;:65$: .ASCIZ  /?/
515                                     64$:
516 003270 000725                    BR       PR02              ; GO BACK ASK AGAIN
517 003272 105037 001313 PR01: CLRB   @#PRONUM      ; CLEAR THE FLAG ONE PROCESSOR
518                                     ; THIS IS THE ACTUAL PROGRAM
519
520 003276 012700 001166      GO:     MOV     #LOGA,R0    ; GET THE TABLE ADDRESS TO R0
521 003302 105037 001324      CLRB   @#INDEX           ; CLR THE INDEX
522 003306 000405                    BR      GO2              ; BRANCH AROUND INCREMENT ROUTINE
523
524 003310 062700 000002      G01:   ADD     #2,R0       ; INDEX THRU THE TABLE
525 003314 022700 001206      CMP     #DRVD,R0         ; DONE?
526 003320 001414                    BEQ     EXIT             ; IF YES GET OUT
527 003322 005710      G02:   TST     (R0)        ; IS THE DRIVE ACTIVE
528 003324 100001                    BPL     G03              ; IF YES BRANCH
529 003326 000770                    BR      G01              ; NO TRY THE NEXT ONE
530 003330 011037 001164      G03:   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      G01              ; GO GET NEXT DRIVE
535 003352 012700 001166      EXIT:  MOV     #LOGA,R0
536 003356 022700 001206      EXITR2: CMP    #DRVD,R0
537 003362 001414                    BEQ     EXIT

```

```

538 003764 032710 040000
539 003764 001011
540 003764 001720
541 003764 100770
542 003764 014001
543 003764 004737 004234
544 003764 004737 005616
545 003764 005720
546 003764 000761
547 003764
548 003764 104401 003422
549 003764 000404
550
551 003432
552 003432 000137 001440
553
554
555
556
557 003436
558 003436 104401 003444
559 003442 000415
560
561 003476
562 003476 012704 000200
563 003476 012703 000001
564 003476 012702 001166
565
566 003512 104401 003520
567 003516 000405
568
569 003532
570 003532 000240
571 003532 010346
572 003532 104403
573 003532 0006
574 003532 0000
575 003542 104401 003550
576 003546 000401
577
578 003552
579 003552 104411
580 003552 012600
581 003552 110001
582 003552 000300
583 003552 042700 177400
584 003552 042701 177400
585 003552 006000
586 003552 103003
587 003552 052700 000200
588 003552 000241
589 003552 006001
590 003552 103002
591 003552 052701 000200
592 003552 110162 000001
593 003552 004737 004014

```

```

BIT #BIT14,(R0)
BNE EXITX
TST (R0)+
BMI EXTFR2
MOV -(R0),R1
JSR PC,D02
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
MOV 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
ROR R1 ;NOW DO THE SAME FOR R1
BCC 3$ ;IF NO ERROR, BRANCH
BIS #BIT7,R1 ;IF ERROR SET THE BIT
MOV R1,1(R2) ;SET DRIVE FIRST IN TABLE
JSR PC,MASKER ;CALL THE MASK CONTROL SUBROUTINE

```

```

594 003624 110062 000001      MOVB   R0,1(R2)      ;SET DRIVE SECOND IN TABLE (NEXT WORD)
595 003630 004737 004014      JSR    PC,MASKER    ;CALL THE MASK CONTROL SUBROUTINE
596 003634 005203          INC    R3           ;INCREMENT THE WORD COUNTER
597 003636 000725          BR     INSYS2      ;GO BACK AND GET NEXT WORD
598 003640 050412          EXITA: BIS   R4,(R2) ;FILL THE TABLE (LAST WORD)
599 003642 006004          ROR   R4           ;WITH ALL BITS SET
600 003644 103375          BCC   EXITA       ;GO BACK IF NOT DONE
601 003646 042712 174000          EXITB: BIC   #174000,(R2) ;CLEAR DOWN AND SECOND SYSTEM BITS AT TABLE
602 003652 010200          MOV   R2,R0       ;GET ADDR =SS TO R0 (CURRENT TABLE)
603 003654 005722          TST   (R2)+       ;ADD 2 TO THE POINTER
604 003656 022702 001206          FIL.DN: CMP  #DRVD,R2 ;TABLE FULL?
605 003658 001403          BEQ  2$          ;IF YES BRANCH
606 003664 012722 100000          MOV   #BIT15,(R2)+ ;FILL THE TABLE
607 003670 000772          BR   FIL.DN      ;GO BACK TRY AGAIN
608 003672 011001          2$:  MOV   (R0),R1   ;GET THE WORD TO R1
609 003674 110102          MOVB  R1,R2
610 003676 004737 005220          JSR   PC,MASK     ;FORM DRIVE # FOR MOUNT
611 003702 004737 004234          JSR   PC,D02      ;WRITE NEW INFO
612 003706 004737 005070          JSR   PC,RDLINK   ;READ SAMPLE (ALL DRIVES)
613 003712 004737 005616          JSR   PC,RDLINK   ;TYPE ASCIZ STRING
614 003716 104401 003724          TYPE  65$        ;GET OVER THE ASCIZ
615 003722 000427          BR   64$
616          ;:65$: .ASCIZ <15><12>/DONE SYSTEM 2 RESTART SYSTEM 1, TYPE WORD /
617          64$:
618 004002 011046          MOV   (R0),-(SP)  ;GET WORD FOR SYSTEM 1 AND TYPE
619 004004 104403          TYPOS
620 004006 006          .BYTE 6
621 004007 001          .BYTE 1
622 004010 000000          1$:  HALT          ;HALT (FINISHED SYSTEM #2)
623 004012 000776          BR   1$          ;GO BACK TO HALT
624
625          ;THIS ROUTINE SETS UP THE MASK BITS IN THE LOG TABLE FOR
626          ;SYSTEM #2, IF A DRIVE IS DOWN NO BIT IS SET BUT THE MASK
627          ;IS SHIFTED.
628
629 004014 005712          MASKER: TST   (R2)      ;IS THE DRIVE UP
630 004016 100401          BMI  RETRM4      ;IF NO, BRANCH
631 004020 110412          MOVB  R4,(R2)    ;MOVE THE MASK BIT IN THE TABLE
632 004022 006004          RETRM4: ROR   R4   ;ROTATE THE MASK
633 004024 103003          BCC  1$          ;DONE? IF NO BRANCH
634 004026 012716 003646          MOV   #EXITB,(SP) ;SET UP FOR RETURN
635 004032 000207          RTS   PC
636 004034 032712 040000          1$:  BIT   #BIT14,(R2) ;IS THIS SYSTEM # 2'S DRIVE?
637 004040 001403          BEQ  2$          ;IF NO, BRANCH
638 004042 012716 003640          MOV   #EXITA,(SP) ;SET UP FOR RETURN
639 004046 000207          RTS   PC
640 004050 062702 000002          2$:  ADD   #2,R2     ;INDEX THRU LOGA TABLE
641 004054 000207          RTS   PC          ;RETURN
642
643          ;ROUTINE TO BUILD THE ACTIVE LOGICAL UNIT TABLE
644
645 004056 022737 000010 001330          STORE: CMP  #10,#KYTEMP ;IS INPUT A LEGAL NUMBER?
646 004064 000240          NOP
647 004066 003417          BLE  ILEGAL      ;***
648 004070 000337 001330          SWAB  #KYTEMP     ;ALIGN DRIVE # FOR TABLE
649 004074 013721 001330          MOV   #KYTEMP,(R1)+ ;PUT THE WORD IN THE TABLE
    
```

```

650 004100 005037 001330          CLR      2#KYTEMP      ;CLEAR THE TEMP WORD
651 004104 10 237 001311          INCB     2#DRCNT1     ;INCREMENT THE COUNTER
652 004110 022701 001206          CMP      2#DRVD,R1   ;IS THE TABLE FULL?
653 004114 001401                    BEQ      TBLFUL       ;IF YES BRANCH
654 004116 000207                    RTS      PC           ;IF NO RETURN
655 004120 012716 003034          TBLFUL: MOV     2#SECSYS,(SP) ;IF TABLE FULL SET UP FOR SYSTEM #2
656 004124 000207                    RTS      PC           ;RETURN
657 004126 012716 002630          ILEGAL: MOV     2#AUTSL2,(SP) ;IF ILLEGAL RESPONSE, GO BACK
658 004132 104401 004140          TYPE    65$         ;TYPE ASCIZ STRING
659 004136 000407                    BR       64$         ;GET OVER THE ASCIZ
660
661 004156
662 004156 000207
663
664
665
666
667
668
669 004160 011001                    CYCLE:  MOV     (R0),R1 ;GET THE LOGICAL UNIT ACTIVE WORD TO R1
670 004162 032701 040000          BIT      2#BIT14,R1 ;IS IT ON SYSTEM #2
671 004166 001402                    BEQ      CYCL2       ;IF NO BRANCH
672 004170 000137 006736          JMP      2#SECONE    ;GO TO SYSTEM #2
673 004174 113703 001324          CYCL2:  MOV     2#IDEX,R3 ;GET THE INDEX VALUE
674 004200 116302 001256          MOV     MSKTBL(R3),R2 ;GET THE MASK TO R2
675 004204 004737 005220          CYCLE2: JSR     PC,MASK ;CALL THE MASK SUBROUTINE
676 004210 012703 001166          LDFLG:  MOV     2#LOGA,R3 ;GET TABLE ADDRESS TO R3
677 004214 020003                    2$:      CMP     R0,R3   ;IS ACTIVE ADDRESS=TO FIRST ADDRESS
678 004216 001002                    BNE     3$          ;IF NO BRANCH
679 004220 050223                    BIS     R2,(R3)+    ;IF YES SET BITS TO SHOW WRITE
680 004222 000401                    BR      4$          ;BRANCH TO SEE IF DONE
681 004224 050223                    3$:     BIC     R2,(R3)+ ;CLEAR BITS TO SHOW OVER-WRITE
682 004226 022703 001206          4$:     CMP     2#DRVD,R3 ;DONE
683 004228 001370                    BNE     2$          ;IF NO GO BACK
684 004230 000301                    002:    SWAB    R1     ;GET DRIVE # TO LOW BYTE
685 004236 000240                    NOP
686
687 004240 110102                    MOV     R1,R2       ;GET IT TO R2
688 004242 042702 000370          BIC     2#370,R2    ;CLEAR THE UNUSED BITS
689 004246 110237 001314          MOV     R2,2#DRIVE ;GET THE DRIVE #
690
691 004248 006102                    ROL    R2           ;SHIFT THE DRIVE # TO
692 004250 016102                    ROL    R2           ;ALIGN IT FOR THE DRIVE ADDR.
693 004252 016102                    ROL    R2           ;KEEP IT MOVING!
694 004254 016102                    ROL    R2           ;A LITTLE MORE!
695 004256 016102                    ROL    R2           ;THERE IT IS
696 004258 006102
697 004260 006102
698 004262 006102
699 004264 110237 001353          MOV     R2,2#DSKTMP+1 ;GET IT TO DISK ADDR. TEMP.
700 004266 110237 001335          MOV     R2,2#DSKADR+1 ;CALL MOUNT
701 004268 004737 004302          JSR     PC,MOUNT    ;RETURN
702 004270 000207                    RTS      PC
703
704 004302 105237 001324          MOUNT:  INCB     2#IDEX ;INCREMENT THE INDEX
705 004306 000240                    NOP
706 004310 112737 000005 001321          MOV     2#5,2#ECNT  ;SET ERROR CNTR TO 5
707 004314 112737 000003 001322          MOV     2#3,2#CNTSIN ;SET SIN CNTR TO 3
708 004318 104401 004332          TYPE    65$         ;TYPE ASCIZ STRING
709 004322 000414                    BR       64$         ;GET OVER THE ASCIZ
710
711 ;;65$: .ASCIZ <15><12>/MOUNT PACK ON DRIVE #/

```

```

706 004362
707 004362 013746 001314
708 004366 104403
709 004370 001
710 004371 000
711 004372 104401 004400
712 004376 000415
713
714 004432
715 004432 104401 004440
716 004436 000420
717
718 004500
719 004 0 000000
720 004 2 004737 004510
721 004506 000207
722
723
724
725
726 004510 010046
727 004512 013700 001334
728 004516 042700 001777
729 004522 005037 001346
730 004526 104037 001323
731 004532 000240
732 004534 010077 174636
733 004540 012777 000001 174622
734 004546 004737 000600
735 004552 105777 174612
736 004556 107423
737 004560 000237 001346
738 004564 001372
739 004566 104401 004574
740 004572 000415
741
742 004626
743 004626 015077 174544
744 004632 105777 174526
745 004636 100415
746 004640 104401 004646
747 004644 000411
748
749 004670
750 004670 000714
751 004672 032777 000040 174464
752 004700 001420
753 004702 104401 004710
754 004706 000414
755
756 004740
757 004740 000670
758 004742 005037 001346
759 004746 010077 174424
760 004752 012777 000015 174410
761 004760 004737 005600

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

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

INITIL: MOV RO,-(SP) ;SAVE RO
MOV #0,SKADR,RO ;GET THE DRIVE # TO RO
BIC #1777,RO ;CLEAR THE UNUSED BITS
INITI2: CLR #0,TIMR ;CLEAR THE TIMER
CLR #0,TIMR2
NOP
MOV RO,DRKDA ;***
;GET DRIVE # TO 'DA' REGISTER
MOV #1,DRKCS ;ISSUE CONTROL RESET + GO
JSR PC,SMTME
1S: TSTB DRKCS ;DID CONTROL READY SET
BMI 2S ;IF YES BRANCH
INC #0,TIMR ;IF NO INCREMENT THE TIMER
BNE 1S ;IF TIMER NOT ZERO BRANCH
TYPE 65S ;TYPE ASCIZ STRING
BR 64S ;GET OVER THE ASCIZ
;;65S: .ASCIZ <15><12>/CONTROL RESET TIME OUT/
64S:
2S: MOV RO,DRKDA ;DRIVE NUMBER TO 'DA' REG.
TSTB DRKCS ;IS DRIVE RDY
BMI 3S ;IF YES BRANCH
TYPE 67S ;TYPE ASCIZ STRING
BR 66S ;GET OVER THE ASCIZ
;;67S: .ASCIZ <15><12>/DRIVE NOT READY/
66S:
BR INITI2 ;GO BACK TRY AGAIN
3S: BIT #BIT5,DRKDS ;IS DRIVE WRITE LOCKED?
BEQ 4S ;IF NO, BRANCH
TYPE 69S ;TYPE ASCIZ STRING
BR 68S ;GET OVER THE ASCIZ
;;69S: .ASCIZ <15><12>/DRIVE WRITE PROTECTED/
68S:
4S: BR INITI2 ;YES, GO BACK TRY AGAIN
CLR #0,TIMR ;CLEAR THE TIMER
MOV RO,DRKDA ;GET THE DRIVE # TO 'DA' REGISTER
MOV #15,DRKCS ;ISSUE DRIVE RESET + GO
JSR PC,SMTME
    
```

```

762 004764 105777 174400 5S: TSTB 2RKCS
763 004770 100375 BPL 5S
764 004772 032777 000100 174364 BIT 2100,2RKDS ;READ/WRITE/SEEK READY BIT SET?
765 005000 001031 BNE 6S ;IF YES, BRANCH
766 005002 005237 001346 INC 28TIMR ;NO, INCREMENT THE TIMER
767 005006 001366 BNE 5S ;GO BACK AND CHECK IF TIMER NOT 0
768 005010 105737 001323 TSTB 28TIMR2
769 005014 001003 BNE 7S
770 005016 105237 001323 INCB 28TIMR2
771 005022 000760 BR 5S
772 005024
773 005024 104401 005032 7S: TYPE 71S ;TYPE ASCIZ STRING
774 005030 000414 BR 70S ;GET OVER THE ASCIZ
775 ;:71S: .ASCIZ <15><12>/DRIVE RESET TIMED OUT/
776 005062 70S: BR 4S ;GO BACK, TRY AGAIN
777 005062 000727 6S: MOV (SP)+,R0 ;RESTORE R0
778 005064 012600 RTS PC ;RETURN TO CALLER
779 005066 000207
780
781 ;THIS ROUTINE TAKES CARE OF ALL LINKAGES FOR THE
782 ;EXECUTE ROUTINE. IT FORMS THE ADDRESS AND SETS UP ALL THE
783 ;REGISTERS FOR THE WRITE OPERATION
784
785 005070 105037 001316 WRLINK: CLRB 2#COMND ;INDICATE WRITE OPERATION
786 005074 000240 NOP ;***
787 005076 113701 001314 MOVB 2#DRIVE,R1 ;PICK UP THE DRIVE #
788 005102 006101 ROL R1 ;MAKE IT A WORD INDEX
789 005104 016137 001206 001362 1S: MOV DRVD(R1),2#PATRN ;PICK UP THE DATA PATTERN
790 005112 004737 005266 JSR PC,CYLADR ;CALL CYLINDER ADDRESS
791 005116 000401 BR 2S ;RETURN HERE IF NOT LAST BASE
792 005120 000207 RTS PC ;RETURN HERE IF LAST BASE
793 005122 012737 001362 001336 2S: MOV 2#PATRN,2#BUSADR ;GET THE ADDRESS OF THE OUTPUT
794 005130 013737 001342 001340 MOV 2#CYLCNT,2#WRCNT ;GET THE WORD COUNT
795 005136 013737 001354 001332 MOV 2#WRTCS,2#CONTRL ;GET THE CONTROL + STATUS WORD
796 005144 004737 005366 JSR PC,EXECUT ;CALL EXECUTE
797 005150 032737 000020 001334 BIT 2#BIT4,2#DSKADR ;WAS THIS WRITE SURFACE "1"?
798 005156 001006 BNE 3S ;IF YES BRANCH
799 005160 052737 000020 001334 BIS 2#BIT4,2#DSKADR ;SET SURFACE ONE BIT
800 005166 105137 001362 COMB 2#PATRN ;MAKE IT SURFACE ONE DATA
801 005172 000753 BR 2S ;RELOAD REGISTERS AND EXECUTE
802 005174 0-2737 000020 001334 3S: BIC 2#BIT4,2#DSKADR ;SET UP FOR SURFACE "0"
803 005202 105137 001362 COMB 2#PATRN ;MAKE IT SURFACE 0 DATA
804 005206 005202 INC R2 ;INC. THRU SELECTED CYL. OFFSET TABLE
805 005210 004737 005276 JSR PC,CYLOFF ;GET THE CYLINDER VALUE
806 005214 000742 BR 2S ;RETURN HERE IF MORE TO READ
807 005216 000207 RTS PC ;RETURN HERE IF FINISHED
808
809 ;THIS ROUTINE EXPECTS TO FIND A MASK IN R2, AND FROM THIS MASK
810 ;BUILDS A TABLE (AT CYL1BL) OF CYLINDER ADDRESS OFFSETS; THE TABLE
811 ;IS TERMINATED BY A #377
812
813 005220 010546 MASK: MOV R5, -(SP) ;SAVE R5
814 005222 042702 177400 BIC 2177400,R2 ;CLR THE UNUSED BITS OF THE MASK
815 005226 000240 NOP ;***
816 005230 012703 000200 MOV 200,R3 ;SET UP THE COMPARE MASK
817 005234 005004 CLR R4 ;CLR THE INDEX COUNTER
    
```

818 005236 012705 001274
819 005242 030203
820 005244 001401
821 005246 110425
822 005250 105204
823 005252 006003
824 005254 103372
825 005256 112715 000377
826 005262 012605
827 005264 000207

15: MOV #CYLTBL,R5 ;GET THE CYLINDER TABLE ADDRESS
BIT R2,R3 ;IS THE MASK BIT SELECTED IN BASE
BEQ 25 ;IF NO BRANCH
25: MOVB R4,(R5)+ ;MOVE THE CYLINDER BASE TO THE TABLE
INCB R4 ;INCREMENT THE BASE
ROR R3 ;ROTATE THE COMPARE MASK
BCC 15 ;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

832 005266 012703 001266
833 005272 012702 001274
834 005276 111204
835 005300 000240
836 005302 122704 000377
837 005306 001416
838 005310 005046
839 005312 111316
840 005314 062604
841 005316 006104
842 005320 006104
843 005322 006104
844 005324 006104
845 005326 006104
846 005330 042737 017777 001334
847 005336 050437 001334
848 005342 000207
849 005344 005203
850 005346 000240
851 005350 022703 001274
852 005354 001401
853 005356 000745
854 005360 062716 000002
855 005364 000207

CYLA0R: MOV #BASE,R3 ;GET THE CYLINDER TABLE ADDRESS
CYLA02: 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 R4 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 CYLA02 ;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

860 005366 005037 001346
861 005372 000240
862 005374 105037 001323
863 005400 013777 001334 173770
864 005406 013777 001336 173760
865 005414 013777 001340 173750
866 005422 013777 001332 173740
867 005430 004737 005600
868 005434 105777 173730
869 005440 100011
870 005442 004737 006066
871 005446 005737 001360
872 005452 001403
873 005454 005037 001360

EXECUT: CLR @TIMER ;CLEAR THE TIMER
NOP ;HALT HERE TO CHECK COMMAND ABOUT TO BE EXECUTED
CLRB @TIMER2 ;CLEAR SECOND TIMER
MOV @DSKADR,@RKDA ;LOAD THE DISK ADDRESS REGISTER
MOV @BUSADR,@RKBA ;LOAD THE BUS ADDRESS REGISTER
MOV @WORDCNT,@RKWC ;LOAD THE WORD COUNT REGISTER
MOV @CONTR,@RKCS ;LOAD THE CONTROL REGISTER
JSR PC,SMTIME ;KILL TIME FOR RK11-C
CHECK1: TSTB @RKCS ;IS CONTROL READY SET
BPL TIME ;IF NO BRANCH
JSR PC,ERRCHK ;ERROR?
TST @ERRFLG ;IF NO BRANCH
BEQ 15 ;CLEAR THE FLAG
CLR @ERRFLG

```

874 005460 000742 BR EXECUT ;TRY AGAIN
875 005462 000207 IS: 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 001073 BNE IS ;IF YES BRANCH
881 005502 105237 001323 INCB @#TIMR2 ;INDICATE SECOND TIMEOUT
882 005506 000752 BR CHECK1 ;GO BACK
883 005510 004737 004510 IS: JSR PC,INITIL
884 005514 104401 005522 TYPE 65$ ;:TYPE ASCIZ STRING
885 005520 000426 BR 64$ ;:GET OVER THE ASCIZ
886 ;:65$: .ASCIZ <15><12>/TIMED OUT ON OPERATION RETRY IN PROGRESS/
887 ;:64$:
888 005576 000673 BR EXECUT
889 005600 012737 000500 001346 SMTME: MOV #500,@#TIMR
890 005606 005337 001346 IS: DEC @#TIMR
891 005612 001375 BNE IS
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 RDLINK: MOV RO,-(SP) ;SAVE RO
900 NOP ;***
901 BISB #377,@#COMND ;INDICATE READ OPERATION
902 MOV #LOGA,R1 ;GET THE TABLE ADDRESS TO R1
903 MOV #SECTBL,R5 ;GET THE SECTOR TABLE ADDRESS
904 BR R02 ;SKIP OVER THE INDEX, FIRST PASS
905 R01: AM #2,R1 ;ADD 2 TO THE ADDRESS
906 CMR #DRVO,R1 ;ARE YOU THRU THE ENTIRE TABLE
907 BEQ EXIT2 ;IF YES EXIT
908 R02: TST (R1) ;IS THE DRIVE ACTIVE
909 BPL R03 ;IF YES BRANCH
910 BR R01 ;IF NO GET NEXT WORD
911 R03: MOV (R1),RO ;GET THE ACTIVE WORD TO RO
912 MOV RO,R2 ;COPY THE WORD
913 SWAB RO ;GET THE DRIVE # TO THE LOW BYTE
914 BIC #177770,RO ;CLR ALL BUT THE DRIVE #
915 MOVB RO,@#WRTNBY ;SAVE THE DRIVE #
916 RUL RO ;MAKE IT AN INDEX
917 MOV DRVO(RO),@#PATTRN ;PICK UP THE DATA PATTERN
918 JSR PC,MASK ;CALL THE MASK SUBROUTINE
919 JSR PC,CYLADR ;GO FORM A CYLINDER ADDRESS
920 BR R04 ;RETURN HERE IF NOT LAST BASE ADDR.
921 BR R01 ;IF LAST BASE ADDRESS, RETURN HERE
922 R04: BIS @#OSKTMP,@#OSKADR ;SET THE DRIVE # BITS IN DISK ADDR.
923 R05: BISB (R5)+,@#OSKADR ;SET THE SECTOR BITS IN DISK ADDR.
924 R06: MOV #JBUFF,@#BUSADR ;GET THE BUFFER ADDR.
925 NOP ;***
926 MOV @#SECCNT,@#WORDCNT ;GET THE WORD COUNT
927 MOV @#READCS,@#CONTRL ;GET THE READ CONTROL WORD
928 JSR PC,EXECUT ;DO THE READ
929 JSR PC,RDCHK ;CHECK THE DATA

```

```

930 005772 042737 000017 001334 BIC #17,2#OSKADR ;CLR THE SECTOR BITS IN DISK ADDR.
931 006000 022705 001311 CMP #DRCNT1,RS ;WAS THIS THE LAST SECTOR?
932 006004 001352 BNE RDS ;IF NO GO BACK
933 006006 012705 001305 MOV #SECTBL,RS ;IF YES RESET SECTOR POINTER
934 006012 032737 000020 001334 BIT #BIT4,2#OSKADR ;WAS IT SURFACE "1" THAT WAS READ?
935 006020 001006 BNE R07 ;IF YES, BRANCH
936 006022 052737 000020 001334 BIS #BIT4,2#OSKADR ;NO, SET SURFACE "1" BIT
937 006030 105137 001362 COMB 2#PATTRN ;MAKE HEAD "1" PATTERN
938 006034 000736 BR RDS ;GO BACK AND EXECUTE
939 006036 042737 000020 001334 RD7: BIC #BIT4,2#OSKADR ;CLEAR THE SURFACE BIT
940 006044 105137 001362 COMB 2#PATTRN ;MAKE IT SURFACE 0 DATA
941 006050 005202 INC R2 ;INCREMENT THE SELECTED CYL TABLE POINTER
942 006052 004737 005276 JSR PC,CYLOFF ;GO FORM NEXT ADDRESS
943 006056 000722 BR R04 ;IF HERE IT IS NOT THE LAST BASE ADDR
944 006060 000670 BR R01 ;IF HERE GET NEXT WORD-DRIVE FINISHED
945
946 006062 012600 EXIT2: MOV (SP)+,R0 ;RESTORE R0
947 006064 000207 RTS PC ;RETURN TO MAIN LINE CODE
948
949 ;THE ERROR CHECK ROUTINE CHECKS IF AN ERROR OCCURRED
950 ;ON WRITING OR READING.
951
952 006066 032777 140000 173274 ERRCHK: BIT #140000,2#KCS ;WAS ERROR OR ERROR SET?
953 006074 000240 NOP ;HALT HERE TO EXAMINE ERROR REG., ECT.
954 006076 001420 BEQ TSTSN1 ;IF NO, GO TEST 'SIN' BIT
955 006100 012777 000001 173262 MOV #1,2#KCS ;IF YES, ISSUE CNTRL RESET + GO
956 006106 004737 005600 JSR PC,SMTHE
957 006112 012777 177777 173240 MOV #1,2#ERRFLG ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)
958 006120 105777 173244 1S: TSTB 2#KCS ;CNTRL READY BIT SET (FROM CNTRL RESET)
959 006124 100375 BPL 1S ;IF NO WAIT. (IF HUNG HERE RUN STATIC)
960 006126 105337 001321 DECB 2#ECNT ;DECREMENT ERROR COUNTER
961 006132 001002 BNE TSTSN1 ;HAVE ERROR BITS SET 5 TIMES?
962 006134 000137 006216 JMP 2#RESTRT ;IF HERE 5 ERRORS HAVE OCCURRED
963 006140 032777 001000 173216 TSTSN1: BIT #1000,2#KDS ;SEEK INCOMPLETE SET?
964 006146 000240 NOP ;***
965 006150 001530 BEQ RETRN2 ;BRANCH IF NO
966 006152 012777 000015 173210 MOV #15,2#KCS ;IF YES, ISSUE DRIVE RESET, GO
967 006160 012777 177777 173172 MOV #1,2#ERRFLG ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)
968 006166 004737 005600 JSR PC,SMTHE
969 006172 105777 173172 2S: TSTB 2#KCS
970 006176 100375 BPL 2S
971 006200 032777 000100 173156 BIT #100,2#KDS ;"R/W/S READY" BIT SET?
972 006206 001771 BEQ 2S ;IF NO WAIT! (IF HUNG HERE RUN STATIC)
973 006210 105337 001322 DECB 2#CNTSIN ;DECREMENT SEEK INCOMPLETE COUNTER
974 006214 001106 BNE RETRN2 ;IF 3 'SIN' ERRORS FALL THROUGH
975 006216 105737 001321 RESTRT: TSTB 2#ECNT
976 006222 000240 NOP ;***
977 006224 001421 BEQ 1S
978 006226 104401 006234 TYPE ,65S ;:TYPE ASCIZ STRING
979 006232 000415 BR ,64S ;:GET OVER THE ASCIZ
980 ;:65S: .ASCIZ <15><12>/3 'SIN' ERRORS OCCURRED/
981 ;:64S:
982 BR 2S
983 1S:
984 TYPE ,67S ;:TYPE ASCIZ STRING
985 BR ,66S ;:GET OVER THE ASCIZ
  
```

```

986
987 006322
988 006322
989 006322 104401 006330
990 006326 000422
991
992 006374
993 006374 105737 001316
994 006400 100006
995 006402 062706 000004
996 006406 012600
997 006410 052710 100000
998 006414 000207
999 006416 052710 100000
1000 006422 012706 001100
1001 006426 000137 003310
1002 006432 000207
1003
1004
1005
1006
1007 006434 010446
1008 006436 010546
1009 006440 105037 001320
1010 006444 000240
1011 006446 012737 000005 001350
1012 006454 012704 007342
1013 006460 013705 001362
1014 006464 020524
1015 006466 001515
1016 006470 105737 001320
1017 006474 001046
1018 006476 104401 006504
1019 006502 000420
1020
1021 006544
1022 006544 152737 000377 001320
1023 006552 113746 001317
1024 006556 104403
1025 006560 001
1026 006561 000
1027 006562 104401 006570
1028 006566 000411
1029
1030 006612
1031 006612
1032 006612 104401 006620
1033 006616 000405
1034
1035 006632
1036 006632 013746 001334
1037 006636 104403
1038 006640 006
1039 006641 001
1040 006642 104401 006650
1041 006646 000405

```

```

;;67$: .ASCIZ <15><12>/5 ERRORS OCCURRED/
66$:
2$:
TYPE 69$ ;;TYPE ASCIZ STRING
BR 68$ ;;GET OVER THE ASCIZ
;;69$: .ASCIZ / DRIVE DECLARED DOWN!! NOT TESTED !/
68$:
TSTB 2#COMND ;TEST THE COMMAND
BPL 3$ ;IF WRITE, BRANCH
ADD #4,SP ;POINT TO SAVE RD
MOV (SP)+,RO ;GET RD BACK
BIS #BIT15,(RO) ;SET THE DOWN BIT
RTS PC ;RETURN TO MAIN CODE
3$: BIS #BIT15,(RO) ;SET THE DOWN BIT
MOV #STACK,SP ;RESTORE THE STACK
JMP 2#G01
RETRN2: RTS PC

;THIS ROUTINE CHECKS A SECTORS WORTH OF DATA ON A READ OPERATION
;IT ALLOWS 5 ERROR PRINTOUTS PER SECTOR
RDCHK: MOV R4,-(SP) ;SAVE R4
MOV R5,-(SP) ;SAVE R5 FOR RDLINK
CLRB 2#HDRFLG ;CLEAR THE PRINT HEADER FLAG
NOP ***
MOV #5,2#CHKCNT ;PUT ERROR COUNT IN CHECK COUNT
MOV #ROBUFF,R4 ;GET THE TABLE ADDRESS TO R4
MOV 2#PATTRN,R5 ;GET THE EXPECTED DATA TO R5
1$: CMP R5,(R4)+ ;ARE THEY THE SAME
BEQ 7$ ;IF YES BRANCH
TSTB 2#HDRFLG ;IS THE HEADER FLAG CLEAR
BNE 2$ ;IF NO BRANCH
TYPE 65$ ;;TYPE ASCIZ STRING
BR 64$ ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <15><12>/ERROR! DATA WRITTEN BY DRIVE /
64$:
BISB #377,2#HDRFLG ;SET THE HEADER FLAG
MOVB 2#WRTNBY,-(SP) ;PICK UP THE DRIVE # THAT WROTE
TYPOS
.BYTE 1
.BYTE 0
TYPE 67$ ;;TYPE ASCIZ STRING
BR 66$ ;;GET OVER THE ASCIZ
;;67$: .ASCIZ / CANNOT BE READ./
66$:
2$:
TYPE 69$ ;;TYPE ASCIZ STRING
BR 68$ ;;GET OVER THE ASCIZ
;;69$: .ASCIZ <15><12>/ ADDR=/
68$:
MOV 2#DSKADR,-(SP) ;PICK UP THE ADDRESS THAT FAILED
TYPOS
.BYTE 6
.BYTE 1
TYPE 71$ ;;TYPE ASCIZ STRING
BR 70$ ;;GET OVER THE ASCIZ

```

G04

```

1042      ;:71$: .ASCIZ / EXPCTD=/
1043      70$:
1044      MOV     RS,-(SP)           ;PICK UP THE EXPECTED DATA (GOOD)
1045      TYPON
1046      TYPE   ,73$              ;;TYPE ASCIZ STRING
1047      BR     ,72$              ;;GET OVER THE ASCIZ
1048      ;:73$: .ASCIZ / RECVD=/
1049      72$:
1050      MOV     -2(R4),-(SP)       ;PICK UP THE RECEIVED DATA (BAD)
1051      TYPON
1052      DEC     @#CHKCNT           ;DECREMENT THE CHECK COUNT
1053      BEQ    ,4$                ;IF ZERO, BRANCH
1054      3$:  CMP     #MANSSEL,R4    ;DONE ALL CHECKS?
1055      BNE    ,1$                ;IF NO, GO BACK
1056      4$:  MOV     (SP)+,RS       ;RESTORE RS
1057      MOV     (SP)+,R4          ;RESTORE R4
1058      RTS     PC                ;RETURN TO CALLER
1059
1060      ; THIS ROUTINE BUILDS A TABLE OF PARAMETERS TO PASS TO THE SECOND SYSTEM
1061      ; IT PACKS THE INFO FOR TWO DRIVES INTO ONE WORD
1062
1063      SECCNE: CLR     R3           ;CLEAR THE WORD COUNTER
1064      NOP
1065      MOV     #MSKTBL,R2         ;***
1066      6$:  CLR     -(R2)
1067      CMP     #PASTBL,R2
1068      BNE    ,6$
1069      MOV     #LOGA,R0          ;GET THE ACTIVE TABLE ADDRESS
1070      1$:  MOV     (R0)+,R1       ;PICK UP THE WORD
1071      SWAB   R1                ;GET THE DRIVE # TO THE LOW BYTE
1072      BIC    #177400,R1        ;CLEAR THE UNWANTED BITS
1073      ROLB  R1                ;ROTATE THE BYTE, WAS DOWN SET?
1074      BCC   ,2$                ;IF NO BRANCH
1075      BIS    #BIT0,R1          ;SHOW THE DRIVE AS DOWN IN THE TABLE
1076      2$:  TSTB  R1             ;IS THIS THE SYSTEM #2 DRIVE?
1077      BMI   ,3$                ;BRANCH IF YES
1078      BICB  #360,R1           ;CLEAR THE UNUSED BITS
1079      MOVB  R1,(R2)+          ;GET THIS # TO THE PASS TABLE
1080      BR    ,1$                ;GET THE NEXT WORD FROM ACTIVE TABLE
1081      3$:  MOVB  R1,(R2)        ;GET THE LAST DRIVE TO THE PASS TABLE
1082      MOV   #PASTBL,R2       ;RESTORE THE TABLE POINTER
1083      TYPE  ,65$              ;TYPE ASCIZ STRING
1084      BR    ,64$              ;GET OVER THE ASCIZ
1085      ;:65$: .ASCIZ <15><12>/LOAD AND START ADDRESS 210 ON SYSTEM #2/
1086      64$:
1087      TYPE  ,67$              ;;TYPE ASCIZ STRING
1088      BR    ,66$              ;;GET OVER THE ASCIZ
1089      ;:67$: .ASCIZ <15><12>/AND TYPE THE BELOW WHEN ASKED FOR IT./
1090      66$:
1091      4$:  INC     R3            ;INCREMENT THE WORD COUNTER
1092      TYPE  ,69$              ;;TYPE ASCIZ STRING
1093      BR    ,68$              ;;GET OVER THE ASCIZ
1094      ;:69$: .ASCIZ <15><12>/WORD /
1095      68$:
1096      MOV   R3,-(SP)          ;GET THE WORD COUNT ON THE STACK
1097      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$: .ASCIZ /=/
1103 70$:
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$: .ASCIZ <15><12>/WORD=/
1119 64$:
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,#$PASS
1127 007316 014401 MOV -(R4),R1
1128 007320 000240 NOP ;***
1129 007322 004737 004210 JSR PC,LDFLG ;CALL D02+
1130 007326 004737 005616 JSR PC,ROLINK ;CALL READ CHECK
1131 007332 013700 001100 MOV #,$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      #DRIVO,R0      ;FIND OUT WHICH DRIVES ARE
1158 010422 005001      CLR      R1              ;PRESENT AND PUT THE
1159 010424 010177 170746 1$:  MOV      R1,DRKDA      ;DRIVE #'S IN A TABLE STARTING
1160 010430 105777 170730      TSTB    DRKDS           ;AT 'DRIVO'. BITS 15-13 CONTAINS
1161 010434 100001      BPL      2$             ;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,DRKCS      ;ISSUE CONTROL RESET + GO !
1169 010464 004737 005600      JSR     PC,SMTME
1170 010470 105777 170674 1$:  TSTB    DRKCS           ;DID CONTROL READY SET?
1171 010474 100375      BPL      1$             ;IF NO WAIT! (IF HUNG RUN STATIC)
1172 010476 012700 020474      MOV      #DRIVO,R0
1173 010502 012077 170670 3$:  MOV      (R0)+,DRKDA
1174 010506 012777 000015 170654 MOV      #15,DRKCS     ;ISSUE DRIVE RESET + GO!
1175 010514 004737 005600      JSR     PC,SMTME
1176 010520 105777 170644 2$:  TSTB    DRKCS
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
1191 010722 000401      BR      8$             ;REQUEST NEW CONTENTS FOR SWITCH REG
1192 010724 000000      HALT
1193 010726 117704 170206 8$:  MOVB    2SWR,R4        ;CONTINUE
1194 010732 001413      BEQ     4$             ;WAIT FOR OPERATOR TO ENTER NEW SWR VALUE
1195 010734 012700 020474      MOV      #DRIVO,R0     ;SW0 TO SW7 TO R4
1196 010740 005001      CLR      R1            ;NONE SET, SO TEST ALL
1197 010742 006004 6$:  ROR     R4             ;TABLE TO STORE DRIVE ADDRS
1198 010744 103001      BCC     5$            ;ADDR OF DRIVE
1199 010746 010120      MOV      R1,(R0)+      ;NEXT SWITCH TO CARRY
1200 010750 062701 020000 5$:  ADD      #20000,R1     ;SWITCH NOT SET
1201 010754 001372      BNE     6$            ;SWITCH SET, SO MOVE ADDR TO TABLE
1202 010756 012720 177777      MOV      #-1,(R0)+    ;ADDR OF NEXT DRIVE
1203 010762 105737 001326 4$:  TSTB    OSPFLG        ;ALL DONE WHEN ZERO
1204 010766 001165      BNE     RWSRDY        ;TABLE TERMINATOR
1205 010770 104401 010776      TYPE    ,69$           ;TYPED ONCE?
1206 010774 000432      BR      68$           ;IF YES, DONT RETYPE
1207      ;:69$: .ASCIZ <15><12>/TOGGLE THE "FIRST CYLINDER ADDRESS" (OUTER LIMIT)/
1208 68$:
1209 011062 65$:  TYPE    ,71$           ;;TYPE ASCIZ STRING
    
```

```

1210 011066 000441          BR      70$          ;;GET OVER THE ASCIZ
1211          ;;71$: .ASCIZ <15><12>/INTO THE 'LOW BYTE (BIT0-7) OF THE SWITCH REGISTER AND THE "LAST
70$:
1212 011172          TYPE      73$          ;;TYPE ASCIZ STRING
1213 011172 104401 011200    BR      72$          ;;GET OVER THE ASCIZ
1214 011176 000430          ;;73$: .ASCIZ <15><12>/CYLINDER 'ADDRESS" (INNER LIMIT) INTO THE HIGH/
1215          72$:
1216 011260          TYPE      75$          ;;TYPE ASCIZ STRING
1217 011260 104401 011266    BR      74$          ;;GET OVER THE ASCIZ
1218 011264 000424          ;;75$: .ASCIZ <15><12>/
1219          74$: BYTE (BIT8-15), THEN PRESS CONTINUE./
1220 011336          INCB     OSPFLG
1221 011336 105237 001326    BIT     #100,ARKDS
1222 011342 032777 000100 170014 RWSRDY: BEQ     RWSRDY
1223 011350 001774          BEQ     #SWREG,SWR
1224 011352 022737 000176 001140 TRYAGN: BNE     1$
1225 011360 001002          GTSWR
1226 011362 104405          BR      CONTIN
1227 011364 000401          1$: HALT
1228 011366 000000          CONTIN: MOV    #1,ARKCS
1229 011370 012777 000001 167772 TSTB   ARKCS
1230 011376 105777 167766    BPL    -4
1231 011402 100375          MOV    SWR,R5
1232 011404 013705 001140    1$: MOVB  (R5)+,R1
1233 011410 112501          BIC    #177400,R1
1234 011412 042701 177400    CMP    #312,R1
1235 011416 022701 000312    BPL    2$
1236 011422 100034          TYPE   65$
1237 011424 104401 011432    BR     64$
1238 011430 000430          ;;65$: .ASCIZ <15><12>/INVALID ADDRESS IN SWITCH REGISTER TRY AGAIN/
1239          64$:
1240 011512          BR     TRYAGN
1241 011512 000717          2$: ROL   R1
1242 011514 006101          ROL   R1
1243 011516 006101          ROL   R1
1244 011520 006101          ROL   R1
1245 011522 006101          ROL   R1
1246 011524 006101          ROL   R1
1247 011526 042701 160037    BIC    #160037,R1
1248 011532 032705 000001    BIT   #1,R5
1249 011536 001003          BNE   3$
1250 011540 010137 001402    MOV   R1,#SEEKI
1251 011544 000403          BR   SEKSET
1252 011546 010137 001404    3$: MOV   R1,#SEEKO
1253 011552 000716          BR   1$
1254          ;;GO BACK AND GET HIGH BYTE.
1255 011554 012703 000050    SEKSET: MOV  #50,R3
1256 011560 013704 001404          MOV  #SEEKO,R4
1257 011564 013705 001402          MOV  #SEEKI,R5
1258          ;;GET THE NUMBER OF SEEK CYCLES TO R3
1259 011570 012700 020474    LDSEEK: MOV  #DRIVO,R0
1260 011574 010477 167576    5$: MOV  R4,ARKDA
1261 011600 052077 167572          BIS  (R0)+,ARKDA
1262 011604 012777 000011 167556    MOV  #11,ARKCS
1263 011612 004737 005600          JSR  PC,SMTME
1264 011616 105777 167546    3$: TSTB ARKCS
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      #DRIVO,R0
1270 011636 012077 167534      MOV      (R0)+,DRKDA ;ADRES THE DRIVE
1271 011642 032777 000100 167514 6$:      BIT      #RMS,DRKDS ;SEEK DONE?
1272 011650 001774      BEQ      7$      ;NO, WAIT
1273 011652 022710 177777      CMP      #-1,(R0)      ;ALL DRIVES DONE?
1274 011656 001367      BNE      6$      ;NO
1275
1276 011660 012700 020474      MOV      #DRIVO,R0
1277 011664 010577 167506      MOV      RS,DRKDA    ;SET OUTER CYL ADRES
1278 011670 057777 167502      BIS      (R0)+,DRKDA ;SET DRIVE ADRES
1279 011674 012777 000011 167466      MOV      #11,DRKCS   ;ISSUE SEEK+GO! (FOR OUTER LIMIT)
1280 011702 004737 005600      JSR      PC,SMTME
1281 011706 105777 167456      TSTB    DRKCS
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      #DRIVO,R0
1288 011726 012077 167444      MOV      (R0)+,DRKDA ;SET DRIVE ADRES
1289 011732 032777 000100 167424 10$:      BIT      #RMS,DRKDS ;SEEK DONE?
1290 011740 011774      BEQ      11$
1291 011742 022710 177777      CMP      #-1,(R0)      ;ALL DONE?
1292 011746 01367      BNE      10$
1293 011750 05303      DEC      R3          ;DONE 50 SEEK CYCLES (100 SEEKS)
1294 011752 001306      BNE      ;IF NO BRANCH (KEEP CYCLING)
1295 011754 000605      BR       LDSEEK     ;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
1316 012110 000401      BR       2$        ;GET SWITCH REGISTER VALUE
1317 012112 000000      BR       ;CONTINUE
1318 012114 012737 000001 020324 1$:      HALT      ;WAIT FOR 'CONTINUE'
1319 012122 005037 020326      MOV      #1,SFCNT   ;SET SHIFT COUNT
1320 012126 033777 020324 167004 2$:      CLR      DRVCNT    ;CLEAR DRIVE COUNT
1321 012134 001011      BIT      SFCNT,SWR  ;IS THIS SW SET?
                        BNE      S10      ;YES FORMAT THIS DRIVE

```

```

1322 012136 006337 020324 S11: ASL SHFCNT
1323 012142 006337 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, RO
1328 012164 104401 001161 TYPE , SCRLF
1329 012170 104401 020330 TYPE , EM1 ; TYPE 'DRIVE'
1330 012174 010046 MOV RO, -(SP) ; TYPE DRIVE #
1331 012176 104402 TYPOC
1332 012200 000300 S-AB RO
1333 012202 006100 ROL RO
1334 012204 006100 ROL RO
1335 012206 006100 ROL RO
1336 012210 006100 ROL RO
1337 012212 006100 ROL RO
1338 012214 011137 001352 MOV RO, @#DSKTMP
1339 012220 012737 000000 001422 MOV #0, ERKCS
1340 012226 012737 000000 001424 MOV #0, ERKCS
1341 012234 012737 000000 001426 MOV #0, ERKCS
1342 012242 012737 000000 001430 MOV #0, ERKCS
1343 012250 012737 000000 001432 MOV #0, ERKCS
1344 012256 012737 177750 001416 S12: MOV #-24, RMC ; SET WORD COUNT FOR READ FORMAT.
1345 012264 012737 164000 001412 MOV #-6144, MC ; 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 012777 001352 167056 BIS @#SKTMP, @RKDA
1350 012320 100777 167044 S1: TSTB @RKCS ; IS THE CONTROLER READY?
1351 012324 100375 BPL S1 ; NO SO WAIT.
1352 012326 012777 000011 167034 MOV #11, @RKCS ; DO THE SEEK.
1353 012334 032777 000100 167022 S2: BIT #BIT6, @RKDS ; IS THE SEEK DONE?
1354 012342 001774 BEQ S2 ; NO SO WAIT.
1355 012344 105777 167020 S3: TSTB @RKCS ; IS CONTROLER READY?
1356 012350 100375 BPL S3 ; NO
1357 012352 012777 000015 167010 MOV #15, @RKCS ; DO A DRIVE RESET.
1358 012360 032777 000100 166776 S4: BIT #BIT6, @RKDS ; IS DRIVE RESET DONE.
1359 012366 001774 BEQ S4 ; NO
1360 012370 01201 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 AGAIN.
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
1379 012510 000612
1380 012512 000137 001440
1381
1382
1383
1384
1385
1386
1387 012516 013777 001412 166646
1388 012524 013777 001410 166644
1389 012532 053777 001352 166536
1390 012540 013777 001406 166626
1391 012546 012777 000000 166614
1392 012554 052777 010000 166606
1393 012562 052777 010002 166600
1394 012570 053777 001420 166572
1395 012576 052777 000001 166564
1396 012604 105777 166560
1397 012610 100375
1398 012612 005777 166552
1399 012616 100541
1400 012620 012737 000000 001422
1401
1402
1403
1404 012626 013777 001416 166536
1405 012634 013777 001410 166534
1406 012642 053777 001352 166526
1407 012650 013777 001414 166516
1408 012656 012777 000000 166504
1409 012664 052777 002000 166476
1410 012672 052777 000004 166470
1411 012700 053777 001420 166462
1412 012706 052777 000001 166454
1413 012714 105777 166450
1414 012720 100375
1415 012722 032777 040000 166440
1416 012730 001134
1417 012732 012737 000000 001424
1418
1419
1420
1421 012740 013777 001412 166424
1422 012746 013777 001410 166422
1423 012754 053777 001352 166414
1424 012762 013777 001406 166404
1425 012770 012777 000000 166372
1426 012776 052777 004400 166364
1427 013004 053777 001420 166356
1428 013012 052777 000006 166350
1429 013020 052777 000001 166342
1430
1431
1432
1433 013026 013703 001416

```

```

645:
GD1: BR S11
      JMP #START ;RESTART

;DISK I/O SUBROUTINE.

;SET UP FOR A WRITE/FORMAT.
10: MOV W0, @R0WC ;SET UP THE WORD COUNT REG.
     MOV DA, @R0DA ;SET UP THE DISK ADDRESS.
     BIS @R0DKTMP, @R0KDA ;SET THE UNIT NUMBER UP.
     MOV BA, @R0KBA ;SET UP THE BUSS ADDRESS.
     MOV #0, @R0KCS ;CLEAR THE CONTROL REG. FOR SET UP.
     BIS @BIT10+@BIT11, @R0KCS ;SET FORMAT&INHIBIT INC. BITS.
     BIS @BIT1, @R0KCS ;SET UP WRITE FUN.
     BIS EXTR, @R0KCS ;SET UP 12OR16 SECTOR PACK.
     BIS @BIT0, @R0KCS ;GO DO THE WRITE FORMAT.
15: TSTB @R0KCS ;IS WRITE FORMAT DONE?
     BPL IS ;NO SO WAIT.
     TST @R0KCS ;WAS THERE A ERROR?
     BMI WFERR ;YES GO SERVICE IT.
     MOV #0, ERRNF ;CLEAR OUT THE ERROR COUNTER.

;SET UP FOR A READ/FORMAT
MOV W0, @R0WC ;SET UP WORD COUNT REG.
MOV DA, @R0KDA ;SET UP DISK ADDRESS.
BIS @R0DKTMP, @R0KDA ;SET THE UNIT NUMBER
MOV BA, @R0KBA ;SET UP THE BUSS ADDRESS.
MOV #0, @R0KCS ;CLEAR THE CONTROL REG.
BIS @BIT10, @R0KCS ;SET THE FORMAT BIT.
BIS @BIT2, @R0KCS ;SET UP READ FUN.
BIS EXTR, @R0KCS ;SET UP 12 OR 16 SECTOR PACK.
BIS @BIT0, @R0KCS ;GO DO THE READ FORMAT.
25: TSTB @R0KCS ;IS THE READ FORMAT DONE?
     BPL 25 ;NO SO WAIT.
     BIT @BIT14, @R0KCS ;WAS TERE A ERROR?
     BNE RFERR ;YES GO SERVICE IT.
     MOV #0, ERRNF ;CLEAR OUT THE ERROR COUNT.

;SET UP FOR A WRITE CHECK.
MOV W0, @R0WC ;SET UP WORD COUNT REG.
MOV DA, @R0KDA ;SET UP DISK ADDRESS.
BIS @R0DKTMP, @R0KDA ;SET UP THE UNIT NUMBER
MOV BA, @R0KBA ;SET UP BUSS ADDRESS.
MOV #0, @R0KCS ;CLEAR THE CONTROL REG.
BIS @BIT11+@BIT8, @R0KCS ;SET INHIBIT INCR &STOP ON SOFT ERROR BITS.
BIS EXTR, @R0KCS ;SET 12 OR 16 SECTOR PACK.
BIS @BIT1+@BIT2, @R0KCS ;SET UP WRITE CHECK FUN.
BIS @BIT0, @R0KCS ;GO DO THE WRITE CHECK.

;CHECK HEADERS READ BY THE READ/FORMAT.
MOV W0, 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  IS:     TSTB     ZRKCS ;
1446 013070 100375          BPL     IS         ;THE CONTROLLER IS STILL BUSY.
1447 013072 005777 166272          TST     ZRKCS     ;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$: .ASCIZ <15><12>/SYSTEM ERROR/
1463          64$:
1464 013164 000000          HALT
1465 013166 000137 001440          JMP      START   ;LET THE TECH. BREATH.
1466 013172 012777 000000 166170  RETRY:  MOV      #0,ZRKCS ;RESTART THE TEST.
1467 013200 012777 000015 166162  MOV      #15,ZRKCS ;CLEAR OUT THE CONTROL REG.
1468 013206 032777 000100 166150  IS:     BIT      #BIT6,ZRKDS ;DO A DRIVE RESET.
1469 013214 001774          BEQ     IS         ;IS IT DONE.
1470 013216 000137 012516          JMP      IO       ;NO SO WAIT.
1471          ;TRY AGAIN.
1472
1473          ;ERRORS FOR READ/FORMAT.
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  IS:     TSTB     ZRKCS ;WAIT FOR THE WRITE CHECK.
1483 013250 100375          BPL     IS
1484 013252 022737 000004 001426  CMP      #4,ERRRFC ;IS IT 4?
1485 013  50 001401          BEQ     FAILED   ;PUT OUT FAILED MESSAGE.
1486 013  52 000743          BR      RETRY    ;NO SO GO TRY IT AGAIN.
1487 013  54 042777 000037 166104  FAILED: BIC      #37,ZRKDA ;PUT WHICH SECTORS HEADER
1488 013272 042702 177740          BIC     #177740,R2
1489 013276 060277 166074          ADD     R2,ZRKDA ;FAILED IN RKDA FOR THE MESSAGE.

```

1490 013302
1491 013302 104401 013310
1492 013306 000417
1493
1494 013346
1495 013346 017701 166024
1496 013352 010102
1497 013354 042702 177770
1498 013360 062702 000260
1499 013364 110237 013543
1500 013370 004337 013570
1501 013374 042702 177776
1502 013400 062702 000260
1503 013404 110237 013542
1504 013410 04337 013574
1505 013414 042702 177776
1506 013420 062702 000260
1507 013424 110237 013554
1508 013430 004337 013574
1509 013434 042702 177770
1510 013440 062702 000260
1511 013444 110237 013532
1512 013450 004337 013570
1513 013454 042702 177770
1514 013460 062702 000260
1515 013464 110237 013531
1516 013470 004337 013570
1517 013474 042702 177774
1518 013500 062702 000260
1519 013504 110237 013530
1520 013510 104401 013520
1521 013514 000137 013562
1522 013520 025015 054503 027114 15:
1523 013526 020040
1524 0 0 030060 020060
1525 0 1 042523 027103 020040
1526 0 2 030060 040
1527 0 45 123 051125 027106
1528 013552 020040
1529 013554 020060 005015 000
1530 013562
1531 013562 000000
1532 013564 000137 001440
1533
1534
1535
1536 013570 006201
1537 013572 006201
1538 013574 006201
1539 013576 010102
1540 013600 000203
1541
1542
1543
1544 013602 032777 040000 165560
1545 013610 001010

FAIL:
TYPE 655 ;:TYPE ASCII STRING
BR 648 ;:GET OVER THE ASCIIZ
;:655: .ASCIIZ <15><12>/PACK FAILED AT (IN OCTAL) /
648:
MOV 2RKDA,R1 ;:GENERAT THE CYL,SECTOR,SURFACE
MOV R1,R2 ;:MESSAGE FROM RKDA
BIC #177770,R2
ADD #260,R2
MOVB R2,SEC+1
JSR R3,SHF3
BIC #177776,R2
ADD #260,R2
MOVB R2,SEC
JSR R3,SHF1
BIC #177776,R2
ADD #260,R2
MOVB R2,SUR
JSR R3,SHF1
BIC #177770,R2
ADD #260,R2
MOVB R2,CYL+2
JSR R3,SHF3
BIC #177770,R2
ADD #260,R2
MOVB R2,CYL+1
JSR R3,SHF3
BIC #177774,R2
ADD #260,R2
MOVB R2,CYL
TYPE 1\$;:TYPE OUT THE GENERATED MESSAGE
JMP STOP ;:BYPASS THE INLINE MESSAGE
;:ASCII <15><12>/CYL. /
CYL: .ASCII /000 /
SEC: .ASCII /SEC. /
SUR: .ASCII /00 /
SUR: .ASCIIZ /0 /<15><12>
STOP: HALT ;:LET OPER DO HIS THING.
JMP START ;:RESTART THE TEST.
;:SHIFT SUBROUTINES.
SHF3: ASR R1 ;:HERE FOR A SHIFT OF 3.
SHF2: ASR R1 ;:HERE FOR A SHIFT OF 2.
SHF1: ASR R1 ;:HERE FOR A SHIFT OF 1.
MOV R1,R2 ;:PUT RESULTES IN THE WORKING REG.
RTS R3
;:ERRORS FOR WRITE CHECK.
WCERR: BIT #BIT14,2RKCS ;:WAS IT A HARD ERROR.
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),RO
1579 014032 104403          TYPOS
1580 014034 001          .BYTE 1
1581 014035 000          .BYTE 0
1582 014036 162700 000060          SUB     #60,RO
1583 014042 100736          BMI    BAD.ON
1584 014044 022700 000010          CMP     #10,RO
1585 014050 003733          BLE    BAD.ON
1586 014052 110037 001314          MOVB   RO,#DRIVE
1587 014056 000300          SWAB
1588 014060 006100          ROL
1589 014062 006100          ROL
1590 014064 006100          ROL
1591 014066 006100          ROL
1592 014070 006100          ROL
1593 014072 010037 001352          MOV     RO,#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      69S      ;;TYPE ASCIZ STRING
1603 014150 000425                BR      68S      ;;GET OVER THE ASCIZ
1604                                ;;69S: .ASCIZ <15><12>/PLACE DRIVE IN RUN ;SHOULD SEE THE RUN,/
1605 014224 000425                68S:
1606 014224 104401 014232      TYPE      71S      ;;TYPE ASCIZ STRING
1607 014230 000423                BR      70S      ;;GET OVER THE ASCIZ
1608                                ;;71S: .ASCIZ <15><12>/POWER, AND ON CYLINDER LAMPS LIGHT./
1609 014300 000425                70S:
1610 014300 104401 014306      TYPE      73S      ;;TYPE ASCIZ STRING
1611 014304 000425                BR      72S      ;;GET OVER THE ASCIZ
1612                                ;;73S: .ASCIZ <15><12>/MAKE DRIVE WRITE ENABLE PRESS CONTINUE/
1613 014360 000000                72S:
1614 014360 000000      HALT
1615 014362 004737 016140      JSR      PC,WRROCK      ;GO WRITE 0'S, RD 0'S, CHECK
1616 014366 104401 014374      TYPE      75S      ;;TYPE ASCIZ STRING
1617 014372 000430                BR      74S      ;;GET OVER THE ASCIZ
1618                                ;;75S: .ASCIZ <15><12><15><12>/WRITE PROTECT THE DRIVE THEN PRESS CONTINUE/
1619 014454 000000                74S:
1620 014454 000000      HALT
1621 014456 004737 016350      JSR      PC,WRPRO      ;GO TRY OVERWRITE
1622 014462 104401 014470      TYPE      77S      ;;TYPE ASCIZ STRING
1623 014466 000426                BR      76S      ;;GET OVER THE ASCIZ
1624                                ;;77S: .ASCIZ <15><12><15><12>/CLEAR WRITE PROTECT THEN PRESS CONTINUE/
1625 014544 000000                76S:
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      2#OSKTMP,2#RKDA ;SET UP DRIVE#
1629 014560 012777 000017 164602      MOV      #17,2#RKCS      ;FUNCTION WRITE PROTECT
1630 014566 004737 005600      JSR      PC,5#TME      ;GO WAIT TIME FOR RK11-C
1631 014572 105777 164572      IS:      TSTB      2#RKCS ;IS CONTROL READY SET
1632 014576 100375                BPL      IS           ;IF NO, BRANCH
1633 014600 004737 016350      JSR      PC,WRPRO      ;GO TRY OVERWRITE OF IERO'S
1634 014604 000431      PRTTWO:
1635 014604 104401 014612      TYPE      65S      ;;TYPE ASCIZ STRING
1636 014610 000431                BR      64S      ;;GET OVER THE ASCIZ
1637                                ;;65S: .ASCIZ <15><12><15><12>/CAUTION! TRY TO OPEN THE DOOR, DO NOT FORCE:/
1638 014674 000414                64S:
1639 014674 104401 014702      TYPE      67S      ;;TYPE ASCIZ STRING
1640 014700 000414                BR      66S      ;;GET OVER THE ASCIZ
1641                                ;;67S: .ASCIZ <15><12>/DOOR SHOULD NOT OPEN!/
1642 014732 000420                66S:
1643 014732 104401 014740      TYPE      69S      ;;TYPE ASCIZ STRING
1644 014736 000420                BR      68S      ;;GET OVER THE ASCIZ
1645                                ;;69S: .ASCIZ <15><12>/PRESS CONTINUE WHEN FINISHED/
1646 015000 000000                68S:
1647 015000 000000      HALT
1648 015002 104401 015010      TYPE      71S      ;;TYPE ASCIZ STRING
1649 015006 000426                BR      70S      ;;GET OVER THE ASCIZ
1650                                ;;71S: .ASCIZ <15><12><15><12>/PUT DRIVE IN LOAD, WAIT FOR LOAD LIGHT/
1651 015064 000420                70S:
1652 015064 104401 015072      TYPE      73S      ;;TYPE ASCIZ STRING
1653 015070 000420                BR      72S      ;;GET OVER THE ASCIZ
1654                                ;;73S: .ASCIZ <15><12>/PRESS CONTINUE WHEN FINISHED/
1655 015132 000000                72S:
1656 015132 000000      HALT
1657 015134 013777 001352 164234      MOV      2#OSKTMP,2#RKDA ;SET UP DISK ADDRESS

```

E05

MAINDEC-11-DZRKI-E MACY11 27(1006) 17-MAR-77 14:57 PAGE 32
 DZRKIE.P11 17-MAR-77 14:53 RK05 CONTROL PANEL TEST

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

```

1714
1715 016116
1716 016116 104401 016124
1717 016122 000404
1718
1719 016134
1720 016134 000137 001440
1721 016140 005037 001362
1722 016144 013777 001352 163224
1723 016152 012777 000001 163210
1724 016160 004737 005600
1725 016164 105777 163200
1726 016170 100375
1727 016172 013777 001352 163176
1728 016200 012777 001362 163166
1729 016206 013777 001344 163156
1730 016214 013777 001354 163146
1731 016222 004737 005600
1732 016226 105777 163136
1733 016232 100375
1734 016234 013777 001352 163134
1735 016242 012777 007342 163124
1736 016250 013777 001344 163114
1737 016 6 013777 001356 163104
1738 016264 004737 005600
1739 016270 105777 163074
1740 016274 100375
1741 016276 012704 007342
1742 016302 005005
1743 016304 020524
1744 016306 001414
1745 016310 104401 016316
1746 016314 000410
1747
1748 016336
1749 016336 000700
1750 016340 022704 010342
1751 016344 001357
1752 016346 000207
1753 016350 032777 000040 163006
1754 016356 001021
1755 016360 104401 016366
1756 016364 000416
1757
1758 016422
1759 016422 012737 177777 001362
1760 016430 013777 001352 162740
1761 016436 012777 001362 162730
1762 016444 013777 001344 162720
1763 016452 013777 001354 162710
1764 016460 004737 005600
1765 016464 105777 162700
1766 016470 100375
1767 016472 032777 020000 162666
1768 016500 001032
1769 016502 104401 016510

::695: .ASCIZ <15><12>/RUN...INTERLOCKS HAVE BEEN CHECKED/
685:
TYPE 715 ;:TYPE ASCIZ STRING
BR 705 ;:GET OVER THE ASCIZ
::715: .ASCIZ <15><12>/DONE!/
705:
JMP 28START
CLR 28PATTRN ;:MAKE A PATTERN OF ZERO'S
MOV 28OSKTMP,28RKDA ;:SET UP DRIVE ADDRESS
MOV 281,28RKCS ;:ISSUE A CONTROL RESET
JSR PC,28TME
55: TSTB 28RKCS ;:CONTROL READY SET
BPL 55 ;:IF NO BRANCH
MOV 28OSKTMP,28RKDA ;:SET UP DRIVE ADDRESS
MOV 28PATTRN,28A ;:GET BUSS ADDRESS
MOV 28SECCNT,28KWC ;:WORD COUNT 1 SECTOR
MOV 28ITCS,28RKCS ;:IBA + WRITE + GO
JSR PC,28TME ;:KILL TIME FOR RK11-C
15: TSTB 28RKCS ;:CONTROL READY SET
BPL 15 ;:IF NO BRANCH
MOV 28OSKTMP,28RKDA ;:SET UP RK REGISTERS
MOV 28UFF,28RKBA ;:TO READ ONE SECTOR
MOV 28CNT,28KWC ;:TO THE READ BUFFER
MOV 28DOCS,28RKCS
JSR PC,28TME ;:KILL TIME FOR RK11-C
25: TSTB 28RKCS ;:CONTROL READY SET
BPL 25 ;:IF NO BRANCH
MOV 28DDBUFF,R4 ;:GET BUFFER TO R4
CLR R5 ;:SET UP TO COMPARE
35: CMP R5,(R4)+ ;:FOR ZERO'S
BEQ 45 ;:IF OK, BRANCH
TYPE 655 ;:TYPE ASCIZ STRING
BR 645 ;:GET OVER THE ASCIZ
::655: .ASCIZ <15><12>/WRITE FAILED/
645:
BR WRROCK ;:GO BACK TRY AGAIN
45: CMP 28MANSEL,R4 ;:DONE ALL CHECKS
BNE 35 ;:IF NO BRANCH
RTS PC ;:IF YES, RETURN
WRPRO: BIT 28BITS,28RKDS ;:BIT 5 ON
BNE 15 ;:IF YES, BRANCH
TYPE 655 ;:TYPE ASCIZ STRING
BR 645 ;:GET OVER THE ASCIZ
::655: .ASCIZ <15><12>/WPS=BITS OF RKDS NOT SET/
645:
15: MOV 28177777,28PATTRN ;:GO LOAD ALL
MOV 28OSKTMP,28RKDA ;:RK REGISTERS
MOV 28PATTRN,28RKBA ;:TO WRITE
MOV 28SECCNT,28KWC ;:ALL ONES (WITH WRITE LOCK)
MOV 28WRITCS,28RKCS ;:OVER THE ZERO'S
JSR PC,28TME ;:KILL TIME FOR RK11-C
25: TSTB 28RKCS ;:CONTROL READY SET?
BPL 25 ;:IF NO BRANCH
BIT 28BIT13,28RKER ;:WLO BIT SET
BNE 35 ;:IF YES BRANCH
TYPE 675 ;:TYPE ASCIZ STRING
    
```

MAINDEC-11-DZRKI-E MACY11 27(1006) 17-MAR-77 14:57 PAGE 34
 DZRKIE.P11 17-MAR-77 14:53 RKOS CONTROL PANEL TEST

```

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$: MOV      #1,2RKCS          ;DO A CONTROL RESET
1774 016574 004737 005600          JSR      PC,SMTME          ;KILL TIME FOR RK11-C
1775 016600 105777 162564          4$: TSTB   2RKCS          ;CONTROL READY SET?
1776 016604 100375          BPL     4$              ;IF NO BRANCH
1777 016606 032777 020000 162552          BIT     #BIT13,2RKER          ;WLO BIT CLEAR
1778 016614 001431          BEQ     RDCHKO          ;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          RDCHKO: MOV     2#DSKTMP,2RKDA          ;SET UP RK REGISTERS
1784 016706 012777 007342 162460          MOV     #ROBUFF,2R#BF          ;TO READ SECTOR 0,
1785 016714 013777 001344 162450          MOV     2#CCNT,2RK1C          ;CYLINDER 0, HEAD 0
1786      6722 013777 001356 162440          MOV     2#HEADCS,2RKCS          ;TO ENSURE NO WRITE TOOK PLACE
1787      6730 004737 005600          JSR     PC,SMTME          ;KILL TIME
1788      J16734 105777 162430          3$: TSTB   2RKCS
1789 016740 100375          BPL     3$
1790 016742 012703 000005          MOV     #5,R3
1791 016746 012704 007342          MOV     #ROBUFF,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          TYP0S
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

```

H05

MAINDEC-11-DZRKI-E MACY11 27(1006) 17-MAR-77 14:57 PAGE 35
DZRKIE.P11 17-MAR-77 14:53 RKOS CONTROL PANEL TEST

1826
1827
1828

MAINDEC-11-DZRKI-E MACY11 27(1006) 17-MAR-77 14:57 PAGE 36
 DZRKIE.P11 17-MAR-77 14:53 RKOS CONTROL PANEL TEST

;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).

1829									
1830									
1831									
1832									
1833									
1834									
1835									
1836									
1837	017160	000240				SECT.4:	NOP		
1838	017162	012777	000001	162200			MOV	#1,DRKCS	
1839	017170	105777	162174		3\$:		TSTB	DRKCS	
1840	017174	100375					BPL	3\$	
1841	017176	012700	020474				MOV	#DRIVO,RO	
1842	017202	005001					CLR	R1	
1843	017204	005002					CLR	R2	
1844									
1845	017206	010210			1\$:		MOV	R2,(RO)	;SET UP ADDRESS TABLE
1846	017210	010277	162162				MOV	R2,DRKDA	;ADDRESS THE DRIVE
1847	017214	105777	162144				TSTB	DRKDS	;IS IT PRESENT?
1848	017220	100021					BPL	2\$;NO
1849									
1850	017222	104401	020330				TYPE	EM1	;TYPE 'DRIVE'
1851	017226	010146					MOV	R1,-(SP)	;TYPE OUT DRIVE #
1852	017230	104402					TYPOC		
1853	017232	104401	020341				TYPE	EM2	;TYPE 'ON LINE'
1854	017236	052710	000300				BIS	#BIT6+BIT7,(RO)	;SET BITS INDICATING THIS ;DRIVE PRESENT
1855									
1856									
1857	017242	012777	000015	162120			MOV	#15,DRKCS	;ISSUE A DRIVE RESET
1858	017250	004737	005600				JSR	PC,SMTME	;ALLOW SOME TIME
1859	017254	032777	000100	162102	4\$:		BIT	#RWS,DRKDS	;WAIT FOR RWS RDY
1860	017262	001774					BEG	4\$	
1861									
1862	017264	005720			2\$:		TST	(RO)+	
1863	017266	005201					INC	R1	
1864	017270	062702	020000				ADD	#20000,R2	;NXT DRIVE
1865	017274	001344					BNE	1\$;ALL DONE?
1866									
1867	017276	104401	001161				TYPE	,SCRLF	
1868									
1869									
1870									
1871									
1872									
1873									
1874	017302	012700	020474		BEGCT:		MOV	#DRIVO,RO	;INITIALIZE POINTERS
1875	017306	005001					CLR	R1	
1876									
1877	017310	011077	162062		BEGCT1:		MOV	(RO),DRKDA	;ADDRESS A DRIVE
1878	017314	042777	017777	162054			BIC	#17777,DRKDA	;MASK OUT NON DR# BITS
1879	017322	105777	162036				TSTB	DRKDS	;IS THIS DRIVE ON LINE?
1880	017326	100044					BPL	1\$;NO
1881									;YES
1882	017330	105710					TSTB	(RO)	;WAS IT 'ON LINE' LAST TIME?
1883	017332	100454					BMI	NXT1	;YES, NO MESSAGE TO REPORT
1884	017334	052710	000200				BIS	#BIT7,(RO)	;IT CHANGED FROM OFF LINE TO ON

```

1885 017340 104401 020330          TYPE      EM1          ;LINE, REPORT MESSAGE
1886 017344 010146          MOV        R1,-(SP)
1887 017346 104402          TYPOC
1888 017350 104401 020341          TYPE      EM2          ;TYPE 'ON LINE'
1889 017354 032777 000040 162002          BIT        #WPS,ARKDS  ;WRITE ENABLED?
1890 017362 001417          BEQ       2$          ;YES, OK
1891 017364 104401 017372          TYPE      65$        ;TYPE ASCIZ STRING
1892 017370 000414          BR        64$        ;GET OVER THE ASCIZ
1893          ;:65$: .ASCIZ <15><12>/ERROR,NOT WRT ENABLED/
1894 017422          64$:
1895 017422 012777 000017 161740          2$: MOV        #17,ARKCS ;WRITE PROT THE DISK
1896 017430 105777 161734          3$: TSTB      ARKCS
1897 017434 100375          BPL       3$
1898 017436 000412          BR        NXT1
1899
1900 017440 105710          1$: TSTB      (R0)      ;WAS THIS DRIVE OFF LINE LAST
1901 017442 100010          BPL       NXT1      ;TIME? BRNCH IF YES
1902 017444 104401 020330          TYPE      EM1          ;IF NOT, REPORT THE CHANGE
1903 017450 010146          MOV        R1,-(SP)  ;TYPE DRIVE #
1904 017452 104402          TYPOC
1905 017454 104401 020353          TYPE      EM3          ;TYPE 'OFF LINE'
1906 017460 042710 000200          BIC       #BIT7,(R0) ;CLEAR BIT TO INDICATE THIS
1907          ;DRIVE 'OFF LINE'
1908
1909          ;THIS CODE CHECKS 'WPS' BIT FOR EVERY DRIVE THAT IS IN 'DRY'
1910          ;CONDITION (ON LINE). IT REPORTS ANY CHANGE IN THE CONDITION OF
1911          ;THE 'WPS' BIT. IF THERE WAS NO CHANGE FROM LAST TIME NOTHING
1912          ;IS REPORTED. AT THE TIME OF ENTRY R0 POINTS TO DRIVE FLAG.
1913
1914 017464 105777 161674          NXT1: TSTB      ARKDS  ;IS THIS DRIVE PRESENT?
1915          ;ARKDA CONTAINS THE DRV #
1916 017470 100033          BPL       NXT2      ;NO, SKIP CHECKING
1917
1918 017472 032777 000040 161664          BIT        #WPS,ARKDS ;WPS BIT SET?
1919 017500 001014          BNE       1$        ;YES
1920          ;WPS BIT CLEAR
1921 017502 032710 000004          BIT        #BIT2,(R0) ;WAS IT CLR LAST TIME ALSO?
1922 017506 001424          BEQ       NXT2      ;YES, NOTHING TO REPORT.
1923          ;WPS CHANGED FROM 'SET'
1924 017510 104401 020330          TYPE      EM1          ;TO 'CLR', REPORT IT
1925 017514 010146          MOV        R1,-(SP)  ;TYPE DRIVE #
1926 017516 104402          TYPOC
1927 017520 042710 000004          BIC       #BIT2,(R0) ;INDICATE THAT 'WPS' IS CLEAR
1928 017524 104401 020401          TYPE      EM5          ;TYPE 'WPS CLEAR'
1929 017530 000413          BR        NXT2
1930
1931 017532 032710 000004          1$: BIT        #BIT2,(R0) ;WPS BIT IS SET
1932 017536 001010          BNE       NXT2      ;WAS IT SET LAST TIME ALSO?
1933          ;YES, NOTHING TO REPORT.
1934          ;WPS CHANGED, FROM 'CLR' TO
1935          ;'SET', REPORT THIS CHANGE
1936 017540 104401 020330          TYPE      EM1          ;TYPE 'DRIVE'
1937 017544 010146          MOV        R1,-(SP)  ;TYPE DRIVE #
1938 017546 104402          TYPOC
1939 017550 104401 020366          TYPE      EM4          ;TYPE 'WPS SET'
1940 017554 052710 000004          BIS       #BIT2,(R0) ;SET FLAG BIT INDICATING WPS SET

```

K05

```

1941
1942
1943
1944
1945
1946
1947
1948 017560 032710 000100 NXT2: BIT #BIT6,(R0) ; WAS THIS DRIVE PRESENT AT BEGNG
1949 017564 001403 BEQ 48 ; NO
1950 017566 105777 161572 TSTB 2RKDS ; IS IT PRESENT NOW?
1951 017572 100402 BMI 38 ; YES
1952 017574 000137 020232 48: JMP DN1DRV ; IF NOT SKIP THIS CHECK
1953
1954 017600 052777 000040 161570 38: BIS #40,2RKDA ; RKDA ALREADY HAS THE DRV #
1955 017606 012777 000011 161554 MOV #11,2RKCS ; SET CYL 1 ADDRESS
1956 017614 105777 161550 18: TSTB 2RKCS ; WAIT FOR CONTROL RDY?
1957 017620 100375 BPL 18 ; SOMETHING WRONG IF CNTAL RDY
1958 017622 032777 010000 161534 BIT #DPL,2RKDS ; DOES NOT COME BACK
1959 017630 001414 BEQ 28 ; DPL BIT SET?
1960 017632 032710 000001 BIT #BIT0,(R0) ; YES, DPL SET
1961 017636 001167 BNE CLRDPL ; WAS 'DPL' SET LAST TIME ALSO?
1962 017640 104401 020330 TYPE EM1 ; YES, NOTHING TO REPORT.
1963 017644 010146 MOV R1,-(SP) ; DPL CHANGED, GOT SET THIS
1964 017646 104402 TYPOC ; TIME, REPORT IT
1965 017650 104401 020417 TYPE EM6 ; TYPE 'POWER LO'
1966 017654 052710 000001 BIS #BIT0,(R0) ; SET FLAG BIT INDICATING THAT
1967 017660 000556 BR CLRDPL ; DPL SET THIS TIME
1968 017662 032710 000001 28: BIT #BIT0,(R0) ; 'DPL' BIT IS CLEAR
1969 017666 001410 BEQ WATSK ; WAS 'DPL' CLEAR LAST TIME ALSO?
1970 017670 104401 020330 TYPE EM1 ; YES, NOTHING TO REPORT
1971 017674 010146 MOV R1,-(SP) ; REPORT THAT 'DPL' BIT CHANGED,
1972 017676 104402 TYPOC ; FROM SET TO CLEAR
1973 017700 104401 020440 TYPE EM7 ; TYPE 'POWER UP'
1974 017704 042710 000001 BIC #BIT0,(R0) ; SET FLAG BIT INDICATING THAT DPL
1975 017710 012705 164220 WATSK: MOV #6000.,R5 ; IS CLEAR THIS TIME
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996

```

THIS CODE PERFORMS A SEEK FUNCTION ON A DRIVE AND CHECKS IF THE 'DPL' BIT SET AS A RESULT, (IF THE POWER WAS CUT OFF FROM THE DRIVE). NOTE THAT ONLY THOSE DRIVES ARE CHECKED WHICH WERE FOUND TO BE PRESENT AT BEGINNING (WHEN THIS TEST WAS ENTERED). SEEK IS DONE TO CYLINDER 1. AT THE TIME OF ENTRY RD POINTS TO THE DRIVE FLAG.

THIS CODE WAITS FOR THE SEEK (DONE ABOVE) TO FINISH. WAITING TIME IS APPROX. 50 MS (FOR THE WORST CASE). IF R/W/S RDY DOES NOT SET WITHIN 50 MS, THEN IT IS ASSUMED THAT A 'SIN' IS POSSIBLE AND THE PROGRAM WAITS FOR 1450 MS MORE, SO THAT THE 'SIN' CAN SET. IF 'SIN' DOES NOT SET WITHIN THIS TIME AN ERROR IS REPORTED:
 SIN DIDN'T OCCUR
 IF R/W/S RDY SETS WITHIN 50 MS THE PROGRAM PROCEEDS TO CHECK THE NEXT DRIVE.

```

1997 017714 032777 000100 161442 1S: BIT #RWS,ARKDS ;R/W/S. RDY SET?
1998 017722 001042 BNE 3S ;YES
1999 017724 005205 INC R5 ;WAIT
2000 017726 001372 BNE 1S
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 2S: BIT #SIN,ARKDS ;SIN SET?
2007 017744 001045 BNE SINST ;YES
2008 017746 005305 DEC R5 ;WAIT
2009 017750 001372 BNE 2S
2010 017752 005704 TST R4
2011 017754 001002 BNE 4S
2012 017756 005204 INC R4
2013 017760 000766 BR 2S
2014 ;1500 MS ELAPSED, BUT SIN
2015 ;DIDN'T SET. ERROR!
2016 017762 4S:
2017 017762 104401 017770 TYPE ,65S ;TYPE ASCIZ STRING
2018 017766 000415 BR 64S ;GET OVER THE ASCIZ
2019 ;65S: .ASCIZ <15><12>/SIN DIDN'T OCCUR, DRIVE/
2020 64S:
2021 020022 010146 MOV R1,-(SP) ;TYPE DRIVE #
2022 020024 104402 TYPOC
2023 020026 000501 BR DNIDRV
2024 020030 032710 000002 3S: BIT #BIT1,(R0) ;DID R/W/S RDY SET LAST TIME?
2025 020034 001476 BEQ DNIDRV ;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 DNIDRV
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 4S ;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 4S: MOV ARKDA,R5 ;SAVE RKDA
2044 020112 012777 000001 161250 MOV #1,ARKCS ;DO CONTROL RESET
2045 020120 105777 161244 1S: TSTB ARKCS ;WAIT FOR CONTROL RDY
2046 020124 100375 BPL 1S
2047 020126 010577 161244 MOV R5,ARKDA
2048 020132 012777 000015 161230 MOV #15,ARKCS ;DO DRIVE RESET. RKDA
2049 ;ALREADY HAS THE DRIVE #
2050 020140 105777 161224 2S: TSTB ARKCS ;WAIT FOR CNTRL RDY
2051 020144 100375 BPL 2S
2052 020146 005005 CLR R5

```

```

2053 020150 032777 000100 161206 3$: BIT #RWS,DRKDS ;R/W/S SET?
2054 020156 001025 BNE DN1DRV ;YES
2055 020160 005205 INC RS
2056 020162 001372 BNE 3$ ;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,DRKCS ;CONTROL RESET
2068 020224 105777 161140 1$: TSTB DRKCS ;WAIT FOR CNTRL RDY
2069 020230 100375 BPL 1$
2070 ;AT THIS STAGE THE DRIVE (# IN DRKDA) 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 ;RD 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 (RD),DRKDA ;GET DRIVE #
2082 020236 105777 161122 TSTB DRKDS ;DRIVE PRESENT?
2083 020242 100017 BPL 3$ ;NO
2084 020244 042777 017777 161124 BIC #17777,DRKDA ;CYL ADRES = 0
2085 020252 012777 000011 161110 MOV #11,DRKCS ;GO, SEEK
2086
2087 020260 105777 161104 1$: TSTB DRKCS ;WAIT FOR CNTRL RDY
2088 020264 100375 BPL 1$
2089
2090 020266 004737 005600 JSR PC,SMTME
2091 020272 032777 000100 161064 4$: BIT #RWS,DRKDS
2092 020300 001774 BEQ 4$
2093
2094 020302 005720 3$: TST (RD)+ ;INCREMENT POINTERS TO
2095 020304 005201 INC R1 ;NEXT DRIVE
2096 020306 020127 000010 CMP R1,#10 ;ALL DONE, THIS CYCLE?
2097 020312 001402 BEQ 2$ ;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 EMI: .ASCIZ <15><12>/DRIVE /
2108 020336 020105 000

```

2109	020341	040	047440	020116	EM2:	.ASCIZ / ON LINE/
2110	044514	040	042516	000		
2111	040	040	047440	043106	EM3:	.ASCIZ / OFF LINE/
2112	046040	040	047111	000105		
2113	020040	051127	020124	020124	EM4:	.ASCIZ / WRT PROT/
2114	051120	052117	000			
2115	040	053440	052122	052122	EM5:	.ASCIZ / WRT ENABLED/
2116	042440	040516	046102	046102		
2117	042105	000				
2118	040	042040	044522	044522	EM6:	.ASCIZ / DRIVE POWER LO/
2119	042526	050040	053517	053517		
2120	051105	040	000117	000117		
2121	040	047520	042527	042527	EM7:	.ASCIZ / POWER UP/
2122	020122	050125	000			
2123	040	051440	047111	047111	EM8:	.ASCIZ / SIN/
2124	000					
2125	020461	040	051440	042505	EM9:	.ASCIZ / SEEK OK/
2126	020466	020113	045517	000		

```

;DRIVE FLAGS FOR CONTROL PANEL TEST #2
;BITS 15,14,13 GIVE THE DRIVE NO (EX: 0,1,2,---)
;BIT 7 IS SET WHEN 'DRY' BIT IS SET FOR THE DRIVE (ON LINE)
;BIT 7 IS CLEAR WHEN 'DRY' IS CLEAR (WHEN DRIVE IS IN LOAD/OFF LINE,
;DRIVE POWER IS CUT OFF)
;BIT 6 IS SET IF A DRIVE IS FOUND TO BE PRESENT (DRY) AT
;THE BEGINING. UNLIKE BIT 7 THIS BIT DOES NOT GET SET OR
;CLEARED AS THE DRIVE CONDITIONS CHANGE. IT JUST INDICATES
;THAT THE DRIVE IS AVAILABLE FOR CHECKING.
;BIT 0 IS SET IF 'DPL' BIT GETS SET, IE: DRIVE POWER OFF
;BIT 0 IS CLEARED WHEN DRIVE POWER IS ON.
;BIT 1 IS SET WHEN SEEK INCOMPLETE 'SIN' OCCURS.
;BIT 1 IS CLEAR WHEN SEEK IS OK.
;BIT 2 IS SET WHEN WRT PROT IS SET FROM CONSOLE.
;BIT 2 IS CLEARED WHEN DRIVE IS WRITE ENABLED.

```

2144	020474					
2145	020474	000000	DRIV0:	.EVEN	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

2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210

020514 000240
020516 104401 020524
020522 000426

020600
020600 104401 020606
020604 000432

020672
020672 104401 020700
020676 000427

020756
020756 104401 021412
020762 005037 021406
020766 104410
020770 012601
020772 112100
020774 162700 000060
021000 002766
021002 022700 000067
021006 002763
021010 000241
021012 006000
021014 006000
021016 006000
021020 006000
021022 010037 020474
021026 112100
021030 001412
021032 020027 000106
021036 001347
021040 105711
021042 001345
021044 042737 020000 020474
021052 005237 021406
021056 013700 020474

```

: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-OSF
:THE LETTER 'F' IS ADDED AS A SUFFIX. FOR SELECTING SURFACE 0
:PUT SW2=0, FOR SELECTING SURFACE 1 PUT SW2=1. SET SW1 =1 TO SELECT
:CYLINDER 64. SET SW1=0 TO SELECT CYLINDER 105.
:IF THE DRIVE IS AN RK-OSF, 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 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?).
    
```

```

SECT.5: NOP
        TYPE      65$          ;;TYPE ASCIZ STRING
        BR        64$          ;;GET OVER THE ASCIZ
;;65$: .ASCIZ  <15><12>/SET SW2=0 FOR SURFACE 0, SW2=1 FOR SUR 1./
64$:   TYPE      67$          ;;TYPE ASCIZ STRING
        BR        66$          ;;GET OVER THE ASCIZ
;;67$: .ASCIZ  <15><12>/SET SW1=0 FOR CYLINDER 105, SW1=1 FOR CYLINDER 64/
66$:   TYPE      69$          ;;TYPE ASCIZ STRING
        BR        68$          ;;GET OVER THE ASCIZ
;;69$: .ASCIZ  <15><12>/PUT ANY SW FROM 2-15 HI TO SELECT NEW DRIVE/
68$:   HDALGN:  TYPE      EM10          ;;ASK FOR DRIVE #
        CLR      FFLAG          ;;FLAG FOR RK-OSF
        ROLIN          ;;GET OPR INPUT
        MOV      (SP)+,R1        ;;ADDR OF COMMAND STRING
        MOVB    (R1)+,RO        ;;FIRST CHAR
        SUB     #60,RO          ;;0 TO 7
        BLT     HDALGN          ;;TOO SMALL
        CMP     #67,RO          ;;MUST BE 7 OR LESS
        BLT     HDALGN          ;;TOO BIG
        CLC
        ROR     RO
        ROR     RO
        ROR     RO
        ROR     RO
        MOV     RO,DRIVO        ;;ADDRESS OF DRIVE
        MOVB    (R1)+,RO        ;;NEXT INPUT CHAR
        BEQ     $$              ;;ALL DONE IF C.R.
        CMP     RO,#'F          ;;IS IT F?
        BNE     HDALGN          ;;NO, SO ERROR
        TSTB   (R1)            ;;NEXT CHAR MUST BE C.R.
        BNE     HDALGN          ;;ELSE, ERROR
        BIC     #BIT13,DRIVO    ;;USE EVEN DRIVE IF RK-OSF
        INC     FFLAG          ;;SHOW F TYPE DRIVE
        MOV     DRIVO,RO        ;;DRIVE ADDR TO RO
    
```



```

2267 021372 032777 177774 157540 BIT #177774,2SMR ;EXIT OUT?
2268 021400 001713 BEQ 10$ ;NO, CONTINUE ON THIS DRIVE
2269 021402 000137 020756 JMP H0ALGN ;YES, GET NEW DRIVE
2270
2271 021406 000000 FFLAG: 0
2272 021410 000000 SWTCH: 0
2273 021412 005015 051104 053111 EMIO: .ASCIZ <15><12>/DRIVE?/
2274 021420 037505 000
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322

```

.SBTTL DISK POWER FAILURE TEST

```

;(DISK)POWER FAILURE (DURING DISK WRITE) TEST
;THIS TEST CHECKS THAT THE INFORMATION WRITTEN ON THE DISK IS
;NOT DESTROYED WHEN THE DISK SENSES A LOSS OF POWER WHILE DOING A WRITE
;AND RETRACTS THE HEADS. UPON ENTRY THE PROGRAM FINDS OUT THE
;FIRST AVAILABLE DRIVE INDICATES IT (DRIVE XX) AND PROCEEDS TO TEST.
;CYLINDERS 0 TO 15 ARE WRITTEN WITH UNIQUE PATTERNS, THEN THE HEADS
;ARE POSITIONED ON CYLINDER 10 (DECIMAL) AND A MESSAGE (DROP
;POWER) IS GIVEN. AFTER RECEIVING THIS MESSAGE THE USER SHOULD
;DROP POWER FROM THE DRIVE. ON SENSING A LOSS OF POW R, THE
;PROGRAM ASKS THE USER TO PUT BACK THE POWER. THE ERRORS (DPL)
;ARE CLEARED AND A WRITE-CHECK IS PERFORMED TO CHECK IF THE
;UNIQUE PATTERNS ON THE DISK (CYLINDERS 0-9 AND 11-15) ARE STILL
;THERE. IF NOT A WRITE CHECK ERROR IS REPORTED.
.EVEN

```

```

2293 021424
2295 021424 005000 SECT.6: CLR R0 ;INITIALIZE DRIVE #
2296 021426 005037 020474 CLR DRIVO
2297 021432 010077 157740 8$: MOV R0,2RKDA ;IS IT PRESENT?
2298 021436 105777 157722 TSTB 2RKDS ;IF NOT SKIP
2299 021442 100406 BMI 9$
2300 021444 005237 020474 10$: INC DRIVO
2301 021450 062700 020000 ADD #20000,R0
2302 021454 001366 BNE 8$
2303 021456 000762 BR SECT.6
2304 021460 104401 020330 9$: TYPE EM1
2305 021464 013746 020474 MOV DRIVO,-(SP) ;GET DRIVE #
2306 021470 104402 TYPOC
2307 021472 012777 000001 157670 MOV #1,2RKCS ;CONTROL RESET
2308 021500 105777 157664 TSTB 2RKCS
2309 021504 100375 BPL .-4
2310 021506 005005 CLR R5 ;INITIALIZE PATTERN TO BE WRITTEN
;0 ON CYL 0, 1 ON CYL 1, ETC
2311
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 R0,2R1
2317 021532 010537 022070 15: MOV R5,BUFR ;FILL THE PATTERN IN DATA BUFFER.
2318 021536 012713 022070 MOV #BUFR,2R3 ;BUS ADRES
2319 021542 012712 164000 MOV #-14000,2R2 ;WRITE 1 CYL (256X12X2 WORDS)
2320 021546 012714 004003 MOV #4003,2R4 ;WRITE, GO, IBA SET
2321 021552 105714 TSTB 2R4 ;WAIT FOR CONTROL READY
2322 021554 100376 BPL .-2

```

```

2323                                     ; DONE
2324 021556 005205                       INC     RS
2325 021560 020527 000020                CMP     RS,#20
2326 021564 001362                       BNE     1$
2327
2328 021566 010005                       MOV     R0,RS
2329 021570 052705 000500                BIS     #500,RS
2330 021574 012737 000012 022070        MOV     #12,BUFR
2331 021602 010511                       MOV     RS,R0
2332 021604 012712 164000                MOV     #-14000,R2
2333 021610 012713 022070                MOV     #BUFR,R3
2334 021614 012714 004003                MOV     #4003,R4
2335 021620 032777 000100 157536 2$:    BIT     #RWS,RKDS
2336 021626 001774                       BEQ     2$
2337 021630 104401 021636                TYPE   ,65$
2338 021634 000406                       BR      64$
2339                                     ;:65$: .ASCIZ /DROP POWER/
2340 021652                               64$:
2341
2342 021652 000407                       BR      5$
2343 021654 010511                       3$:   MOV     RS,R0
2344 021656 012712 164000                MOV     #-14000,R2
2345 021662 012713 022070                MOV     #BUFR,R3
2346 021666 012714 004003                MOV     #4003,R4
2347 021672 105714                       5$:   TSTB   R4
2348 021674 100376                       BPL    .-2
2349 021676 005714                       TST    R4
2350                                     ; WAIT FOR DRIVE POWER TO GO DOWN,
2351 021700 100365                       BPL    3$
2352                                     ; OTHERWISE, KEEP ON WRITING ON CYL 10
2353
2354 021702 104401 021710                TYPE   ,67$
2355 021706 000406                       BR      66$
2356                                     ;:67$: .ASCIZ <15><12>/POWER ON/
2357 021724                               66$:
2358 021724 105777 157434                TSTB   RKDS
2359 021730 100375                       BPL    .-4
2360 021732 012714 000001                MOV     #1,R4
2361 021736 105714                       TSTB   R4
2362 021740 100376                       BPL    .-2
2363 021742 010077 157430                MOV     R0,RKDA
2364 021746 005005                       CLR     RS
2365 021750 010537 022070                6$:   MOV     RS,BUFR
2366 021754 012713 022070                MOV     #BUFR,R3
2367 021760 012712 164000                MOV     #-14000,R2
2368 021764 012714 004007                MOV     #4007,R4
2369 021770 105714                       TSTB   R4
2370 021772 100376                       BPL    .-2
2371
2372 021774 005714                       TST    R4
2373 021776 100023                       BPL    7$
2374 022000 104401 022006                TYPE   ,69$
2375 022004 000416                       BR      68$
2376                                     ;:69$: .ASCIZ <15><12>/ERROR, ON PWR-UP, RKDA=/
2377 022042                               68$:
2378 022042 011146                       MOV     R1,-(SP)

```

2379 022044 104402
 2380 022046 005205
 2381 022050 020527 000012
 2382 022054 001774
 2383 022056 020527 000020
 2384 022062 001332
 2385
 2386 022064 000137 021444
 2387
 2388 022070 000000
 2389
 2390
 2391
 2392
 2393
 2394
 2395
 2396
 2397
 2398
 2399
 2400
 2401
 2402
 2403
 2404
 2405
 2406
 2407
 2408 022072 105737 001157
 2409 022076 104402
 2410 022100 000000
 2411 022102 000407
 2412 022104 010046
 2413 022106 017600 000002
 2414 022112 112046
 2415 022114 001005
 2416 022116 005726
 2417 022120 012600
 2418 022122 062716 000002
 2419 022126 000002
 2420 022130 122716 000011
 2421 022134 001430
 2422 022136 122716 000200
 2423 022142 001006
 2424 022144 005726
 2425 022146 104401
 2426 022150 001161
 2427 022152 105037 022306
 2428 022156 000755
 2429 022160 004737 022242
 2430 022164 123726 001156
 2431 022170 001350
 2432 022172 013746 001154
 2433
 2434 022176 105366 000001

75: TYPOC
 INC R5
 CMP R5,#12 ;CYLINDER 10 ?
 BEQ 75 ;BR IF YES, INCREMENT AGAIN
 CMP R5,#20
 BNE 65
 JMP 105
 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 15 ;; BR IF YES
HALT ;; HALT HERE IF NO TERMINAL
BR 35 ;; LEAVE
15: MOV R0,-(SP) ;; SAVE R0
MOV 22(SP),R0 ;; GET ADDRESS OF ASCIZ STRING
25: MOVB (R0)+,-(SP) ;; PUSH CHARACTER TO BE TYPED ONTO STACK
BNE 45 ;; BR IF IT ISN'T THE TERMINATOR
TST (SP)+ ;; IF TERMINATOR POP IT OFF THE STACK
605: MOV (SP)+,R0 ;; RESTORE R0
35: ADD #2,(SP) ;; ADJUST RETURN PC
RTI ;; RETURN
45: CMPB #HT,(SP) ;; BRANCH IF <HT>
BEQ 85 ;; BRANCH IF NOT <CR LF>
CMPB #CRLF,(SP)
BNE 55
TST (SP)+ ;; POP <CR><LF> EQUIV
TYPE ;; TYPE A CR AND LF
$CRLF
$CHARCNT ;; CLEAR CHARACTER COUNT
CLR R0 ;; GET NEXT CHARACTER
BR 25 ;; GO TYPE THIS CHARACTER
55: JSR PC,$TYPEC ;; IS IT TIME FOR FILLER CHARS.?
65: CMPB $FILLC,(SP)+ ;; IF NO GO GET NEXT CHAR.
BNE 25 ;; GET # OF FILLER CHARS. NEEDED
MOV $NULL,-(SP) ;; AND THE NULL CHAR.
75: DECB 1(SP) ;; DOES A NULL NEED TO BE TYPED?

```

```

2435 022202 002770          BLT      65          ;; 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   $CHARCNT  ;; DO NOT COUNT AS A COUNT
2438 022214 000770          BR       75          ;; LOOP
2439
2440          ;HORIZONTAL TAB PROCESSOR
2441
2442 022216 112716 000040    85:     MOVB   #' (SP)      ;; REPLACE TAB WITH SPACE
2443 022222 004737 022242    95:     JSR      PC,$TYPEC  ;; TYPE A SPACE
2444 022226 132737 000007 022306    BITB   #',$CHARCNT  ;; BRANCH IF NOT AT
2445 022234 001372          BNE     95          ;; TAB STOP
2446 022236 005726          TST    (SP)+        ;; POP SPACE OFF STACK
2447 022240 000724          BR      25          ;; GET NEXT CHARACTER
2448 022242 105777 156702    $TYPEC: TSTB   2$TPS     ;; WAIT UNTIL PRINTER IS READY
2449 022246 100375          BPL    $TYPEC
2450 022250 116677 000002 156674    MOVB   2(SP),2$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     15          ;; BRANCH IF NO
2453 022266 105037 022306    CLRB   $CHARCNT     ;; YES--CLEAR CHARACTER COUNT
2454 022272 000406          BR      $TYPEX      ;; EXIT
2455 022274 122766 000012 000002    15:     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          $TYPOS: MOV    2(SP),-(SP)  ;; PICKUP THE MODE
2487 022316 116637 000001 022535    MOVB   1(SP),$OFILL  ;; LOAD ZERO FILL SWITCH
2488 022324 112637 022537          MOVB   (SP)+,$SOMODL+1 ;; NUMBER OF DIGITS TO TYPE
2489 022330 062716 000002          ADD    #2,(SP)      ;; ADJUST RETURN ADDRESS
2490 022334 000406          BR     $TYPON

```

H06

MAINDEC-11-DZRKI-E MACY11 27(1006) 17-MAR-77 14:57 PAGE 48
DZRKIE.P11 17-MAR-77 14:53 BINARY TO OCTAL (ASCII) AND TYPE

2491	022336	112737	000001	022535	STYPOC:	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	STYPON:	MOVB	#5,\$OCNT	:: SET THE ITERATION COUNT
2494	022350	010346				MOV	R3,-(SP)	:: SAVE R3
2495	022352	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		15:		ROL	R5	:: ROTATE MSB INTO "C"
2505	022420	000404				BR	3\$:: GO DO MSB
2506	022422	006105		25:		ROL	R5	:: FORM THIS DIGIT
2507	022424	006105				ROL	R5	
2508	022426	006105				ROL	R5	
2509	022430	010503				MOV	R5,R3	
2510	022432	006103		35:		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		45:		INC	R4	:: DON'T SUPPRESS ANYMORE 0'S
2518	022456	052703	000060			BIS	#'0,R3	:: MAKE THIS DIGIT ASCII
2519	022462	052703	000040		55:	BIS	#' R3	:: MAKE ASCII IF NOT ALREADY
2520	022466	110337	022532			MOVB	R3,#5	:: SAVE FOR TYPING
2521	022472	104401	022532			TYPE	8\$:: GO TYPE THIS DIGIT
2522	022476	105337	022534		75:	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		65:		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		85:		.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								
2542	022540	000000				.ENABL	LSB	
2543	022542	000000				\$TKCNT:	.WORD 0	:: NUMBER OF ITEMS IN QUEUE
2544	022544	000000				\$TKQIN:	.WORD 0	:: INPUT POINTER
2545	022546	000001				\$TKQOUT:	.WORD 0	:: OUTPUT POINTER
2546	022547					\$TKQSR:	.BLKB 1	:: TTY KEYBOARD QUEUE
						\$TKQEND:	.	

```

2547          022550          .EVEN
2548
2549          ;*TK INITIALIZE ROUTINE
2550          ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
2551          ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
2552
2553          ;*CALL:
2554          ;*      JSR      PC,$TKINT
2555          ;*      RETURN
2556
2557          $TKINT: CLR      $TKCNT          ;: CLEAR COUNT OF ITEMS IN QUEUE
2558          MOV      $TKOSRT,$TKQIN      ;: MOVE THE STARTING ADDRESS OF THE
2559          MOV      $TKQIN,$TKQOUT      ;: QUEUE INTO THE INPUT & OUTPUT POINTERS.
2560          MOV      $TKSRV,$TKVEC      ;: INITIALIZE THE KEYBOARD VECTOR
2561          MOV      #200,$TKVEC+2      ;: "BR" LEVEL 4
2562          TST      $TKB                ;: CLEAR DONE FLAG
2563          MOV      #100,$TKS          ;: ENABLE TTY KEYBOARD INTERRUPT
2564          RTS      PC                  ;: RETURN TO CALLER
2565
2566          ;*TK SERVICE ROUTINE
2567          ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
2568          ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
2569          ;*IT IN THE QUEUE.
2570
2571          $TKSRV: MOVB     $TKB,-(SP)    ;: PICKUP THE CHARACTER
2572          BIC      #177,(SP)          ;: STRIP THE JUNK
2573          1$:  CMP      (SP),#7        ;: IS IT A CONTROL G?
2574          BNE     2$                ;: BRANCH IF NO
2575          CMP      $SWREG,$SWR        ;: IS SOFT-SWR SELECTED?
2576          BEQ     6$                ;: GO TO SWR CHANGE
2577
2578          2$:
2579          CMP      #1,$TKCNT          ;: IS THE QUEUE FULL?
2580          BNE     3$                ;: BRANCH IF NO
2581          TYPE     $BELL              ;: RING THE TTY BELL
2582          TST     (SP)+              ;: CLEAN CHARACTER OFF OF STACK
2583          BR      5$                ;: EXIT
2584          3$:  CMP      (SP),#23      ;: IS IT A CONTROL-S?
2585          BNE     32$               ;: BRANCH IF NO
2586          CLR     $TKS              ;: DISABLE TTY KEYBOARD INTERRUPTS
2587          TST     (SP)+              ;: CLEAN CHAR OFF STACK
2588          31$:  TSTB    $TKS          ;: WAIT FOR A CHAR
2589          BPL     31$              ;: LOOP UNTIL ITS THERE
2590          MOVB    $TKB,-(SP)        ;: GET THE CHARACTER
2591          BIC     #177,(SP)         ;: MAKE IT 7-BIT ASCII
2592          CMP     (SP)+,#21         ;: IS IT A CONTROL-Q?
2593          BNE     31$              ;: BRANCH IF NO
2594          MOV     #100,$TKS         ;: REENABLE TTY KEYBOARD INTERRUPTS
2595          RTI
2596          32$:  INC      $TKCNT        ;: COUNT THIS CHARACTER
2597          CMP     (SP),#140         ;: IS IT UPPER CASE?
2598          BLT     4$                ;: BRANCH IF YES
2599          CMP     (SP),#175         ;: IS IT A SPECIAL CHAR?
2600          BGT     4$                ;: BRANCH IF YES
2601          BIC     #40,(SP)         ;: MAKE IT UPPER CASE
2602          4$:  MOVB     (SP)+,$TKQIN  ;: AND PUT IT IN QUEUE
    
```

```

2603 022766 005237 022542          INC      STKQIN          ;; UPDATE THE POINTER
2604 022772 023727 022542 022547  CMP      STKQIN,#STKQEND ;; GO OFF THE END?
2605 023000 001003                BNE      SS              ;; BRANCH IF NO
2606 023002 012737 022546 022542  MOV      #STKQRT,STKQIN ;; RESET THE POINTER
2607 023010 000002                RTI                    ;; RETURN
2608
2609
2610
2611
2612
2613
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    #STKS          ;; IS A CHAR WAITING?
2617 023026 100101                BPL      15$            ;; IF NOT, EXIT
2618 023030 117746 156112          MOVB    #STKB,-(SP)    ;; YES
2619 023034 042716 177600          BIC     #1C17,(SP)    ;; MAKE IT 7-BIT ASCII
2620 023040 021627 000007          CMP     (SP),#?       ;; IS IT A CONTROL-G?
2621 023044 001300                BNE      2$            ;; IF NOT, PUT IT IN THE TTY QUEUE
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658

```

 *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 *ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
 *SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
 *CALL WHEN OPERATING IN TTY INTERRUPT MODE.

 *CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
 *ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
 *CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.

```

6$:  CMPB   #AUTOB,#1          ;; ARE WE RUNNING IN AUTO-MODE?
      BEQ    2$                ;; BRANCH IF YES
      TST   (SP)+             ;; CLEAR CONTROL-G OFF STACK
      JSR   PC,STKINT         ;; FLUSH THE TTY INPUT QUEUE
      CLR   #STKS             ;; DISABLE TTY KEYBOARD INTERRUPTS
      MOVB  #1,$INTAG         ;; SET INTERRUPT MODE INDICATOR
      TYPE  ,SCNTLG           ;; ECHO THE CONTROL-G (↑G)
      TYPE  ,SMSWR           ;; TYPE CURRENT CONTENTS
      MOV   SWREG,-(SP)       ;; SAVE SWREG FOR TYPEOUT
      TYPOC ;,SMNEW          ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
      TYPE  ,SMNEW           ;; PROMPT FOR NEW SWR
      CLR   -(SP)             ;; CLEAR COUNTER
      CLR   -(SP)             ;; THE NEW SWR
      TSTB  #STKS            ;; CHAR THERE?
      BPL   7$                ;; IF NOT TRY AGAIN
      MOVB  #STKB,-(SP)      ;; PICK UP CHAR
      BIC   #1C17,(SP)       ;; MAKE IT 7-BIT ASCII
      9$:  CMP   (SP),#25      ;; IS IT A CONTROL-U?
      BNE   10$              ;; BRANCH IF NOT
      TYPE  ,SCNTLU          ;; YES, ECHO CONTROL-U (↑U)
      ADD   #6,SP            ;; IGNORE PREVIOUS INPUT
      BR    19$              ;; LET'S TRY IT AGAIN
      10$: CMP   (SP),#15     ;; IS IT A <CR>?
      BNE   16$              ;; 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 023178 062706 000006          AOD     @6,SP         ;; CLEAR UP STACK
2663 023180 104401 001161          TYPE   $CRLF         ;; ECHO <CR> AND <LF>
2664 023214 123727 001135 000001    CMPB   $INTAG,#1     ;; RE-ENABLE TTY KBD INTERRUPTS?
2665 023216 001003          BNE     15$           ;; BRANCH IF NOT
2666 023218 012777 000100 155712    MOV     @100,@STKS   ;; RE-ENABLE TTY KBD INTERRUPTS
2667 023220 000002          RTI                     ;; RETURN
2668 023234 004737 022242          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 A II
2674 023260 005766 000002          TST     2(SP)        ;; IS THIS THE FIRST CHAR
2675 023264 001403          BEQ     17$           ;; BRANCH IF YES
2676 023266 006316          PSL     (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          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          TYPE   $QUES        ;; TYPE ?<CR><LF>
2683 023312 000720          BR      20$         ;; SIMULATE CONTROL-U
2684          .DSABL  LSB

```

```

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

```

```

2695 023314 011646 000004 000002  SRDCHR: MOV     (SP),-(SP)    ;; PUSH DOWN THE PC AND
2696 023316 016666 000004          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   @STKGOUT,4(SP) ;; GET ONE CHARACTER
2706 023360 005237 022544          INC     $TKGOUT     ;; UPDATE THE POINTER
2707 023364 023727 022544 022547    CMP     $TKGOUT,@STKGEND ;; DID IT GO OFF OF THE END?
2708 023372 001003          BNE     2$           ;; BRANCH IF NO
2709 023374 012737 022546 022544    MOV     @STKQSR,@STKGOUT ;; RESET THE POINTER
2710 023402 000002          RTI                     ;; RETURN

```

```

*****
; THIS ROUTINE WILL INPUT A STRING FROM THE TTY
; CALL:
;   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 SRDLIN: MOV R3, -(SP) ;: SAVE R3
2719 023406 005046 CLR -(SP) ;: CLEAR THE RUBOUT KEY
2720 023410 012703 023640 1$: MOV #STTYIN, R3 ;: GET ADDRESS
2721 023414 022703 023670 2$: CMP #STTYIN+30, R3 ;: BUFFER FULL?
2722 023420 101456 BLOS 4$ ;: BR IF YES
2723 023422 104407 ROCHR ;: 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 001303 6$: DEC R3 ;: BACKUP BY ONE
2733 023460 001027 023640 CMP R3, #STTYIN ;: STACK EMPTY?
2734 023464 103434 BLO 4$ ;: 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 #SCNTLU ;: 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 #SCRLF ;: TYPE A "CR" & "LF"
2751 023550 104401 023640 TYPE #STTYIN ;: TYPE THE INPUT STRING
2752 023554 000717 BR 2$ ;: GO PICKUP ANOTHER CHARACTER
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 023580 001305 BNE 2$ ;: LOOP IF NOT RETURN
2759 023582 101303 177777 CLRB -1(R3) ;: CLEAR RETURN (THE 15)
2760 023586 101301 001162 TYPE #LF ;: TYPE A LINE FEED
2761 023592 005726 TST (SP)+ ;: CLEAN RUBOUT KEY FROM THE STACK
2762 023594 012603 MOV (SP)+, R3 ;: RESTORE R3
2763 023596 011646 MOV (SP), -(SP) ;: ADJUST THE STACK AND PUT ADDRESS OF THE
2764 023600 016666 000004 000002 MOV 4(SP), 2(SP) ;: FIRST ASCII CHARACTER ON IT
2765 023604 012766 023640 000004 MOV #STTYIN, 4(SP)
2766 023608 000002 RTI ;: RETURN
2767 023636 000 .BYTE 0 ;: STORAGE FOR ASCII CHAR. TO TYPE
2768 023637 000 .BYTE 0 ;: TERMINATOR
2769 023640 000030 STTYIN: .BLKB 30 ;: RESERVE 30 BYTES FOR TTY INPUT
2770 023670 177607 000377 SBELL: .ASCIZ <26><377><377> ;: CODE FOR BELL

```

```

2771 021674 052536 005015 000 $CNTLU: .ASCIZ /↑U/<15><12> ;;CONTROL "U"
2772 021701 0136 00507 000012 $CNTLG: .ASCIZ /↑G/<15><12> ;;CONTROL "G"
2773 021706 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 000004 000002 $RDOCT: MOV (SP),-(SP) ;; PROVIDE SPACE FOR THE
2788 023732 015666 000004 000002 MOV 4(SP),2(SP) ;; INPUT NUMBER
2789 023740 010046 MOV RO,-(SP) ;; PUSH RO ON STACK
2790 023742 010146 MOV R1,-(SP) ;; PUSH R1 ON STACK
2791 023744 010246 MOV R2,-(SP) ;; PUSH R2 ON STACK
2792 023746 104410 15: ROLIN ;; READ AN ASCIZ LINE
2793 023750 012600 MOV (SP)+,RO ;; GET ADDRESS OF 1ST CHARACTER
2794 023752 007001 CLR R1 ;; CLEAR DATA WORD
2795 023754 007002 CLR R2
2796 023756 112046 25: MOVB (RO)+,-(SP) ;; PICKUP THIS CHARACTER
2797 023760 001412 BEQ 3$ ;; IF ZERO GET OUT
2798 023762 015301 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 #1C7,(SP) ;; STRIP THE ASCII JUNK
2805 024002 062601 ADD (SP)+,R1 ;; ADD IN THIS DIGIT
2806 024004 000764 BR 2$ ;; LOOP
2807 024006 005726 35: 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)+,RO ;; POP STACK INTO RO
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 000002 $STRAP: MOV RO,-(SP) ;; SAVE RO
2824 024034 016600 000002 MOV 2(SP),RO ;; GET TRAP ADDRESS
2825 024040 005740 TST -(RO) ;; BACKUP BY 2
2826 024042 111000 MOVB (RO),RO ;; GET RIGHT BYTE OF TRAP

```

```

2827 024044 006300          ACL      RO          ;; POSITION FOR INDEXING
2828 024046 016000 024066  MOV     $TRPAD(RO),RO  ;; INDEX TO TABLE
2829 024052 000200          RTS      RO          ;; GO TO ROUTINE
2830
2831
2832          ;; THIS IS USE TO HANDLE THE "GETPRI" MACRO
2833
2834 024054 011646          STRAP2: 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          $RDOCHR ;; CALL=RDOCHR TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
2855 024106 023404          $RDOLIN ;; CALL=RDOLIN TRAP+10(104410) TTY TYPEIN STRING ROUTINE
2856 024110 023730          $RDOCT ;; CALL=RDOCT TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY
2857
2858          .SBTTL POWER DOWN AND UP ROUTINES
2859
2860          ;; *****
2861          ;; POWER DOWN ROUTINE
2862          ;; *****
2861 024112 012737 024252 000024 $PWRDN: MOV     $ILLUP,@PWRVEC ;; SET FOR FAST UP
2862 024120 012737 000340 000026  MOV     @340,@PWRVEC+2 ;; PRIO:7
2863 024126 010046          MOV     RO,-(SP)      ;; PUSH RO ON STACK
2864 024130 010146          MOV     R1,-(SP)      ;; PUSH R1 ON STACK
2865 024132 010246          MOV     R2,-(SP)      ;; PUSH R2 ON STACK
2866 024134 010346          MOV     R3,-(SP)      ;; PUSH R3 ON STACK
2867 024136 010446          MOV     R4,-(SP)      ;; PUSH R4 ON STACK
2868 024140 010546          MOV     R5,-(SP)      ;; PUSH R5 ON STACK
2869 024142 017746 154772  MOV     @SWR,-(SP)     ;; PUSH @SWR ON STACK
2870 024146 010637 024256  MOV     SP,$SAVR6     ;; SAVE SP
2871 024152 012737 024164 000024  MOV     @SPWRUP,@PWRVEC ;; SET UP VECTOR
2872 024160 000000          HALT
2873 024162 000776          BR      -2          ;; HANG UP
2874
2875          ;; *****
2876          ;; POWER UP ROUTINE
2877          ;; *****
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  IS:    INC     $SAVR6  ;; WAIT FOR THE INC
2881 024206 001375          BNE     IS           ;; OF WORD
2882 024210 012677 154724  MOV     (SP)+,@SWR   ;; POP STACK INTO @SWR

```

```

2893 024214 012605      MOV      (SP)+,R5      ;; POP STACK INTO R5
2894 024216 012604      MOV      (SP)+,R4      ;; POP STACK INTO R4
2895 024218 012603      MOV      (SP)+,R3      ;; POP STACK INTO R3
2896 024220 012602      MOV      (SP)+,R2      ;; POP STACK INTO R2
2897 024222 012601      MOV      (SP)+,R1      ;; POP STACK INTO R1
2898 024224 012600      MOV      (SP)+,R0      ;; POP STACK INTO R0
2899 024226 012737 024112 000024  MOV      @SPWRDN,@PWRVEC ;; SET UP THE POWER DOWN VECTOR
2900 024228 012737 000340 000026  MOV      @340,@PWRVEC+2 ;; Prio:7
2901 024230 104401      TYPE      SPOWER      ;; REPORT THE POWER FAILURE
2902 024232 024260      SPWRMG: .WORD SPOWER ;; POWER FAIL MESSAGE POINTER
2903 024234 000002      RTI
2904 024236 000000      $ILLUP: HALT      ;; THE POWER UP SEQUENCE WAS STARTED
2905 024238 000776      BR      .-2      ;; BEFORE THE POWER DOWN WAS COMPLETE
2906 024240 000000      $$AVR6: 0      ;; PUT THE SP HERE
2907 024260 005015 047520 042527  SPOWER: .ASCIZ <15><12>"POWER"
2908 024266 000122
2909
2910      .EVEN
2911      .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#	
BUSAOR	001336	253#	793#	864	924#					
CHECK1	004434	868#	878	882						
CHKCNT	00350	258#	1011#	1052#						
CKSWR =	104406	2853#								
CLROPL	020216	1965	1973	2067#						
CNTSIN	001322	242#	702#	973#						
COM	012304	1348#	1361							
COMMON	012300	1347#								
COMNO	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#							

PATRN	001362	263#	789*	793	800*	803*	917*	937*	940*	1013	1721*	1728	1759*	1761
PIR9 =	177772	34#												
PIR9VE=	000240	128#												
PRNUM	001313	235#	491*	516*										
PRO1	003272	485	516#											
PRO2	003144	483	491#	515										
PRTTMO	014604	1634#												
PRO =	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*										
PSM =	177776	32#												
PMRVEC=	000024	123#	328*	329*	2861*	2862*	2871*	2877*	2889*	2890*				
RBA	001414	278#	1407	1435	1436									
RBUFF	013660	278	1559#											
ROBUFF	007342	924	1012	1134#	1735	1741	1784	1791	2261					
ROCHK	006434	929	1007#											
ROCHKO	016700	1778	1783#											
ROCHR =	104407	479	497	1577	2723	2854#								
ROLIN =	104410	449	2189	2792	2855#									
ROLINK	005616	533	544	613	899#	1130								
ROOCT =	104411	413	579	1120	2856#									
ROTB	001226	207#												
RO1	005642	905#	910	921	944									
RO2	005654	904	908#											
RO3	005662	909	911#											
RO4	005724	920	922#	943										
RO5	005732	923#	932	938										
RO6	005736	924#												
RO7	006036	935	939#											
READCS	001356	261#	927	1737	1786									
RESTR	006216	962	975#											
RESVEC=	000010	118#												
RETR2	007254	1115#												
RETR2	006432	965	974	1002#										
RETR3	005360	852	854#											
RETR4	004022	630	632#											
RETRY	013172	1458	1466#	1476	1486	1553								
RFERR	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*	1906	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*

MFERR	013122	1399	1456#							
MPS =	000040	291#	1889	1918						
MROCNT	001340	254#	794*	865	926*					
MRTCS	001354	260#	795	1730	1763					
MRLINK	005070	532	612	785#						
MFRPO	016350	1621	1633	1753#						
MROCK	016140	1615	1627	1721#	1749					
MRTNBY	001317	239#	915*	1023						
X	015476	1685	1690#							
XX	015712	1703#								
SAUTOB	001134	177#	359*	2628	2777					
SPADR	001122	172#								
S DAT	001126	174#								
SELL	023670	2581	2770#							
SCHARC	022306	2427*	2437*	2444	2453*	2458#				
SCKSWR	023012	2614#	2853							
SCHTAG	001100	160#	319	320						
SCH3 =	000000	190#								
SCNTLG	023701	2635	2772#							
SCNTLU	023674	2652	2745	2771#						
SCRFL	001161	191#	1328	1867	2426	2461	2663	2750	2770	
SENDAD	001400	149	270#							
SERFLG	001103	163#								
SERMAX	001115	169#								
SERRPC	001116	170#								
SERRTB	001434	307#								
SERTTL	001112	167#								
SFILLC	001156	188#	2430	2461						
SFILLS	001155	187#	2461							
SGADR	001120	171#								
SGOAT	001124	173#								
SGTSWR	023102	2636#	2851							
SHD =	000003	11	12							
SHIOCT	024030	2809#	2814#							
SICNT	001104	164#								
SILLUP	024252	2861	2877	2894#						
SINTAG	001135	178#	2633*	2664	2777					
SITEMB	001114	168#								
SLF	001162	192#	2461	2760	2770					
SLPADR	001106	165#								
SLPERR	001110	166#								
SMAIL =	***** U	346	355	2414						
SNEW	023717	2639	2775#							
SOR	023706	2636	2773#							
SNULL	001154	186#	2432	2461						
SOCNT	022534	2493#	2522*	2535#						
SOMODE	022536	2468#	2492*	2497	2500*	2511*	2537#			
SPASS	001100	161#	1126*	1131						
SPWER	024260	2892	2897#							
SPWRDN	024112	328	2861#	2889						
SPWRMG	024246	2892#								
SPWRUP	024164	2871	2877#							
SQUES	001160	190#	419	2461	2662	2753	2770			
SROCHR	023314	2695#	2854							
SRODEC=	***** U	2857								
SROLIN	023404	2718#	2855							

L07

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)