

RK05/RK11

UTILITY PACKAGE
MD-11-DZRKI-D

EP-DZRKI-D-DL-B
COPYRIGHT © 1976
FICHE 1 OF 1

DEC 1976
digital
MADE IN USA

IDENTIFICATION

SEE CCC:

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

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, 1976 BY DIGITAL EQUIPMENT CORPORATION

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

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 RKCS OR RKOSF DISK DRIVES (DRIVE TYPES MAY BE MIXED)
- 2.2 STORAGE
THIS PROGRAM REQUIRES 8K
- 2.3 PRELIMINARY PROGRAMS
THIS IS NOT A DIAGNOSTIC, PACKAGE 1* IS ASSUMED THAT ALL EQUIPMENT IS FUNCTIONAL

3. LOADING PROCEDURE

- 3.1 METHOD
PROCEDURE FOR NORMAL BINARY TAPES SHOULD BE FOLLOWED
- ABSOLUTE LOADER MUST BE IN MEMORY.
 - PLACE BINARY TAPE IN READER.
 - LOAD ADDRESS #7500 (*DETERMINED BY LOCATION OF LOADER).
 - 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 ='

SEG 0004

EACH TIME SWITCH SETTING ARE ENTERED, THE ENTIRE SWITCH REGISTER IMAGE MUST BE ENTERED. LEADING ZEROS ARE NOT REQUIRED. 'RUBCU' AND 'CONTROL U' FUNCTIONS MAY BE USED TO CORRECT TYPING ERRORS DURING SWITCH ENTRY.

ON PROCESSORS WITH HARDWARE SWITCH REGISTERS, THE 'SOFTWARE' SWITCH REGISTER MAY BE USED. IF THE PROGRAM FINDS ALL 16 SWITCHES IN THE 'UP' POSITION, ALL SWITCH REGISTER REFERENCES WILL BE TO THE 'SOFTWARE' REGISTER AND THE PROCEDURES DESCRIBED ABOVE MUST BE FOLLOWED.

PROGRAM WILL TYPE MINI MONITOR ROUTINE

5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
SEE SEC. 9.0 FOR SWITCHES APPLICABLE TO INDIVIDUAL ROUTINES.
 - 5.2 SUBROUTINE ABSTRACTS
NOT APPLICABLE
 - 5.3 PROGRAM AND/OR OPERATOR ACTOR
SEE INDIVIDUAL PACKAGE DESCRIPTION (PARAGRAPH 9)
6. ERRORS
 - 6.1 ERROR HALTS AND DESCRIPTION
IF HALTED A MAJOR PROBLEM EXIST CHECK CODE AT HALT PC TO DETERMINE WHAT OCCURRED.
 - 6.2 ERROR RECOVERY
EXPLAINED IN DETAIL IN INDIVIDUAL PACKAGE DESCRIPTION (PARAGRAPH 9).
7. RESTRICTIONS
 - 7.1 STARTING RESTRICTIONS
IT IS NOT RECOMMENDED THAT YOU START AT AN ADDRESS OTHER THAN 200. (REASON EXPLAINED IN PARAGRAPH 9.1) UNLESS DIRECTED TO BY THE PROGRAM.
 - 7.2 OPERATIONAL RESTRICTIONS
EXPLAINED IN DETAIL IN INDIVIDUAL PACKAGE DESCRIPTIONS (PARAGRAPH. 9)
8. EXECUTION TIME
VARIES WITH SELECTED ROUTINE, NUMBER OF DRIVES, ETC.
9. PROGRAM DESCRIPTION

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

SECTION	NAME
0 I N D E X	INDEX
	COMPATIBILITY TEST
	OSCILLATING SEEK PACKAGE
	FORMATTER SURFACE VERIFIER
	FRONT PANEL TEST
	RK05 CONTROL PANEL TEST #2
	HEAD ALIGNMENT ROUTINE
	POWER FAILURE (DURING WRITE) TEST

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

9.1 SECTION 0 INDEX

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

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

USE: THIS IS EXAMPLE OF THE ACTUAL OUTPUT:

RK11 UTILITY PACKAGE

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

TYPE=X

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

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

9.2 SECTION 1 COMPATIBILITY PACKAGE

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

DESCRIPTION: THIS PACKAGE ALLOWS A USER TO AUTOMATICALLY TEST

GO1

SEE CODE

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 IMPLIES MORE THAN THE FACT THAT INFORMATION WHICH WAS WRITTEN ON ONE DRIVE CAN BE READ ON ANOTHER. FOR DRIVES TO BE CONSIDERED TRULY COMPATIBLE ANY DRIVE SHOULD BE ABLE TO READ WHAT WAS WRITTEN BY ANY OTHER DRIVE AND ALSO MUST BE ABLE TO OVERWRITE A PORTION OF INFORMATION WRITTEN BY ANOTHER DRIVE, WITH NEW INFORMATION, AND READ IT BACK. THIS IS A VERY BROAD DEFINITION BUT IS THE BASIC PREMISE OF TRUE COMPATIBILITY. 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 :

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

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

32. 0007

INDEX	NAME	TYPE
	COMPATIBILITY PACKAGE	0 1

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

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=1

DRIVE NUMBERS ON SYSTEM 1=1,0

IS THERE A SECOND SYSTEM?Y

DRIVE # =0

MOUNT PACK ON DRIVE #1

MAKE PACK WRITE ENABLE

PRESS CONTINUE WHEN DRIVE RDY

MOUNT PACK ON DRIVE #0

MAKE PACK WRITE ENABLE

PRESS CONTINUE WHEN DRIVE RDY

LOAD AND START ADDRESS 210 ON SYSTEM #2

AND TYPE THE BELOW WHEN ASKED ON SYSTEM #2

AND TYPE THE BELOW WHEN ASKED FOR IT.

WORD 1=000002
WORD 2=000200

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

COMPATIBILITY-SYSTEM#2
WORD 1=000002

WORD 2=000200

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

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

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

WORD=000077

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

RK11 UTILITY PACKAGE

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

TYPE=

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

9.3 SECTION 2 OSCILLATING SEEK PACKAGE

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

DESCRIPTION: SELECT TYPE 2, THE USER THEN INSERTS THE DRIVES TO BE TESTED IN SW0 TO SW7 OF THE SWITCH REGISTER. A SWITCH IS SET FOR EACH DRIVE (E.G. SW2 TO TEST DRIVE 2). THE USER THEN INSERTS THE ADDRESS TO SEEK IN THE SWR. IF BOTH ADDRESS ARE LEGAL, 50 CYCLES (100 SEEKS) WILL BE MADE BETWEEN THE SPECIFIED ADDRESS THEN THE PROGRAM WILL LOOK AT THE SWR FOR POSSIBLE CHANGES THIS SHOULD ALLOW FOR GOOD STABLE TRACES ON AN OSCILLISCOPE. IT SHOULD BE NOTED THAT THE OSCILLATING SEEKS BETWEEN THE SPECIFIED CYLINDERS ARE DONE ON ALL AVAILABLE DRIVES. THE ONLY WAY TO EXIT IS HALT!, LOAD ADDRESS 200. HIT START.

USE: SELECT TYPE 2, RESPOND TO QUESTION WITH UNIT NUMBER...
 TYPE=2
 OSCILLATING SEEK PACKAGE
 SET SW0 TO SW7 TO SELECT THE DRIVES TO TEST AND CONTINUE. IF ALL SWITCHES ARE RESET, ALL AVAILABLE DRIVES WILL BE TESTED. TOGGLE THE "FIRST CYLINDER ADDRESS" (OUTER LIMIT) INTO THE LOW BYTE (BITS-7) OF THE SWITCH REGISTER AND THE "LAST CYLINDER ADDRESS" (INNER LIMIT) INTO THE HIGH BYTE (BITS-15). THEN PRESS CONTINUE
 ... FOLLOW INSTRUCTIONS TYPED

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-05F SHOULD NOT BE SELECTED FOR TESTING AT THE SAME TIME.

9.4 SECTION 3 FORMATTER-SURFACE VERIFIER

PURPOSE: TO FORMAT VIRGIN PACKS OR REFORMAT AN OLDER PACK AND VERIFY ITS SURFACE

DESCRIPTION: SELECT TYPE 3, RESPOND TO THE QUESTION BY SETTING SWITCHES CORRESPONDING TO DRIVE NUMBERS TO BE FORMATTED. THUS IF DRIVES 0,1,2 ARE TO BE FORMATTED SET SWITCHES 0,1,2. THE DRIVES ARE FORMATTED ONE AFTER ANOTHER AT COMPLETION PACK GOOD

K01

USE:

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

SEQ 0010

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

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

PACK GOOD.

RK11 UTILITY PACKAGE

INDEX	NAME	TYPE
	COMPATIBILITY PACKAGE	0

AFTER THE PACK IS FORMATTED A GOOD MESSAGE IS
GIVEN AND A CHECK IS MADE TO SEE IF THERE ARE
ANY MORE PACKS TO BE FORMATTED. IF THERE ARE
NONE CONTROL IS TRANSFERRED TO THE MINI-MONITOR
ERROR INFO: DRIVE PROBLEM, IF THE MESSAGE....

SYSTEM ERROR

.... IS TYPED IT INDICATES A FAULTY DRIVE OR
CONTROLLER. RUN DIAGNOSTICS, THE PROCESSOR WILL HALT
PRESS CONTINUE TO RETURN TO MINI MONITOR.
BAD SPOT, OR SURFACE PROBLEM, ETC.

PACK FAILED AT (IN OCTAL) CYLINDER SECTOR SURFACE

9.5 SECTION 4 RK05 CONTROL PANEL TEST

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

DESCRIPTION: SELECT TYPE 4, RESPOND TO QUESTION WITH UNIT NUMBER. FOLLOW
DIRECTIONS GIVEN. AT COMPLETION MESSAGE "DONE!" IS GIVEN

USE: SELECT TYPE 4, RESPOND TO QUESTION WITH THE UNIT NUMBER....

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

TYPE=4

RK05 CONTROL PANEL TEST. WHICH DRIVE'D
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
	OSCILLATING SEEK PACKAGE	1
	FORMATTER-SURFACE VERIFIER	2
	RK05 CONTROL PANEL TEST	3
	RK05 CONTROL PANEL TEST #2	4
	HEAD ALIGNMENT ROUTINE	5
	POWER FAILLRE (WRITE) TEST	6
		7

TYPE=

9.6 SECTION 5 RK05 CONTROL PANEL TEST #2

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

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

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

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

THE PROGRAM, THEN STARTS SCANNING ALL DRIVES, ONE AFTER THE OTHER. CHECKS IF THE DRIVE IS ON LINE OR OFF LINE (DRY SET OR CLEAR). THEN IT CHECKS IF THE DRIVE IS WRITE ENABLED OR WRITE PROTECTED. THEN A SEEK (TO CYLINDER 1) IS DONE AND 'DPL' BIT IS CHECKED TO SEE IF DRIVE POWER IS LOW OR OK. IF THE DRIVE IS POWERED, IT IS CHECKED IF THE SEEK IS DONE OR SEEK INCOMPLETE OCCURS. WHEN EVER ANY CHANGE IN THE STATUS IS FOUND, IT IS REPORTED. IF THE 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 AFFECTED BY THE USER. IF ANY CHANGE IN STATUS IS NOT DETECTED AND REPORTED BY THE PROGRAM IT MIGHT IMPLY AN ERROR CONDITION.

9.7 SECTION 6 HEAD ALIGNMENT ROUTINE

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

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

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

TYPE=6
DRIVE=0<CR>

THE UPPER OR THE LOWER HEAD CAN BE SELECTED BY SWITCH 0. IF SURFACE 0 IS TO BE SELECTED, PUT SW 0 TO 0. IF SURFACE 1 IS TO BE SELECTED PUT SW 0 ON 1. THE HEADS MAY BE 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 THAT DRIVE:
DROP POWER
MEANWHILE WRITE IS BEING DONE ON CYLINDER 10. ON GETTING THE ABOVE MESSAGE THE USER SHOULD DROP THE POWER ON THAT DRIVE. ON SENSING A LOSS OF POWER, THE PROGRAM WILL ASK THE USER TO PUT THE POWER ON AGAIN:
POWER ON
ON RECEIVING THE ABOVE MESSAGE THE USER SHOULD PUT THE POWER ON. ON DETECTING POWER UP THE PROGRAM PROCEEDS TO CHECK THAT THE DATA WRITTEN ON CYLINDERS 0 TO 15 WAS INTACT. IF A WRITE CHECKS ERROR OCCURS (POSSIBLY MEANING THAT SOME OF THE DATA WAS DESTROYED DURING THE LOSS OF POWER) IT IS REPORTED AS FOLLOWING:
ERROR, ON POWER-UP, RKDA=XXXX
XXXX IS THE CONTENTS OF RKDA AT THE TIME OF ERROR.

THE PROGRAM DOES THE ABOVE POWER FAIL TEST

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

9.9 SECTION SPECIAL

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

ERROR EXAMPLE 1

```

FORMATTER-SURFACE VERIFIER      3
RKDS 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
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 SYTEM 1=0.
    
```

```

IS THERE A SECOND SYSTEM?N
MOUNT PACK ON DRIVE #0
MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE
DRIVE NOT READY
DRIVE NOT READY
DRIVE NOT READY
DRIVE NOT READY
DRIVE NOT READY
    
```

:CONTINUE PRESSED BUT
:DRIVE NOT READY. IF UP
:TO SPEED ETC. AND MESSAGE
:OCCURRING - STATIC SHOULD BE

DRIVE NUMBERS ON SYTEM 1=0.

```

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

```

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

ERROR EXAMPLE 5

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

RK11 UTILITY PACKAGE

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

TYPE=1
DRIVE NUMBERS ON SYTEM 1=0.

```

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

```

RK11 UTILITY PACKAGE

NAME	TYPE
INDEX	0
COMPATIBILITY PACKAGE	1

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

ERROR EXAMPLE 6

RK11 UTILITY PACKAGE

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

TYPE=1
DRIVE NUMBERS ON SYSTEM 1=0.

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

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

RK11 UTILITY PACKAGE

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

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

ERROR EXAMPLE 7

MOUNT PACK ON DRIVE #0
 MAKE PACK WRITE ENABLE
 PRESS CONTINUE WHEN DRIVE RDY
 ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
 ADDR=000367 EXPCTD=077400 RECD=077600
 ADDR=000367 EXPCTD=077400 RECD=037600
 ADDR=000367 EXPCTD=077400 RECD=037600
 ADDR=000367 EXPCTD=077400 RECD=037600
 ADDR=000367 EXPCTD=077400 RECD=037600
 ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
 ADDR=002564 EXPCTD=077400 RECD=077600
 ADDR=002564 EXPCTD=077400 RECD=037600
 ADDR=002564 EXPCTD=077400 RECD=037600
 ADDR=002564 EXPCTD=077400 RECD=037600
 ADDR=002564 EXPCTD=077400 RECD=037600
 ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
 ADDR=002764 EXPCTD=077400 RECD=077600
 ADDR=002764 EXPCTD=077400 RECD=037600
 ADDR=002764 EXPCTD=077400 RECD=037600
 ADDR=002764 EXPCTD=077400 RECD=037600
 ADDR=002764 EXPCTD=077400 RECD=037600
 ERROR! DATA WRITTEN BY DRIVE 1 CANNOT BE READ.
 ADDR=002767 EXPCTD=077400 RECD=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 RKR REGISTER
 BEFORE THE PROGRAM CORRECTS ANY ERRORS WHICH
 WHICH MAY HAVE OCCURRED.
2. IF PLAGED BY CHECKSUM ERRORS AND THE USER WISHES
 MORE ERROR MAPPING THEN HE MAY MODIFY THE
 MASK WORD AT LOCATION 'ERRCHK+2' TO ONLY RECOGNIZE
 HARD ERRORS.
3. TO INCREASE OR DECREASE THE NUMBER OF PTRYs ALLOWED

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

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

A FINAL LOOK: THE FOLLOWING SECTION SHOWS ALL PACKAGES
 CALLED IN SEQUENCE, NONE WITH ERRORS.
 RK11 UTILITY PACKAGE

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

TYPE=0

RK11 UTILITY PACKAGE

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

TYPE=1

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

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

MAKE PACK WRITE ENABLE
PRESS CONTINUE WHEN DRIVE RDY
DONE!

RK11 UTILITY PACKAGE

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

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

RK11 UTILITY PACKAGE

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

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

RK11 UTILITY PACKAGE

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

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

CLEAR WRITE PROTECT THEN PRESS CONTINUE

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

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

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

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

RK11 UTILITY PACKAGE

23
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500

BASIC DEFINITIONS
TRAP CATCHER
STARTING ADDRESS(ES)
ACT11 HOOKS
COMMON TAGS
ERROR POINTER TABLE
INITIALIZE THE COMMON TAGS
TYPE PROGRAM NAME
GET VALUE FOR SOFTWARE SWITCH REGISTER
COMPATIBILITY TEST
OSCILLATING SEEK ROUTINE
FORMATTER-SURFACE VERIFIER
RK05 CONTROL PANEL TEST
CONTROL PANEL TEST # 2
HEAD ALIGNMENT ROUTINE
DISK POWER FAILURE TEST
TYPE ROUTINE
BINARY TO OCTAL (ASCII) AND TYPE
TTY INPUT ROUTINE
READ AN OCTAL NUMBER FROM THE TTY
TRAP DECODER
TRAP TABLE
POWER DOWN AND JP ROUTINES

```

.*TITLE MAINDEC-11-DZRKI-D
.*COPYRIGHT (C) 1974,1976
.*DIGITAL EQUIPMENT CORP.
.*MAYNARD, MASS. 01754
.*
.*PROGRAM BY BOB COLLINS
.*
.*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
.*PACKAGE (MAINDEC-11-DZQAC-C2), SEPT 14, 1976.
.*
$TN=1
$SWR=160000 ;;HALT ON ERROR, LOOP ON TEST. INHIBIT ERROR TYP0UT

```

000001
160000

```

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

```

.SBTTL BASIC DEFINITIONS

001100

```

.*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

000011
000012
000015
000200
177776

177774
177772
177570
177570

```

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 ;;HARDWARE SWITCH REGISTER
DDISP= 177570 ;;HARDWARE DISPLAY REGISTER

```

.*GENERAL PURPOSE REGISTER DEFINITIONS

000000
000001
000002
000003
000004
000005
000006
000007
000006
000007

```

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

000000
000040
000100
000140
000200
000240

```

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

```

000000
000001
000002
000003
000004
000005
000006
000007
000008
000009
000010
000011
000012
000013
000014
000015
000016
000017
000018
000019
000020
000021
000022
000023
000024
000025
000026
000027
000028
000029
000030
000031
000032
000033
000034
000035
000036
000037
000038
000039
000040
000041
000042
000043
000044
000045
000046
000047
000048
000049
000050
000051
000052
000053
000054
000055
000056
000057
000058
000059
000060
000061
000062
000063
000064
000065
000066
000067
000068
000069
000070
000071
000072
000073
000074
000075
000076
000077
000078
000079
000080
000081
000082
000083
000084
000085
000086
000087
000088
000089
000090
000091
000092
000093
000094
000095
000096
000097
000098
000099
000100
000101
000102
000103
000104
000105
000106
000107
000108
000109
000110
000111
000112
000113
000114
000115
000116
000117
000118
000119
000120
000121
000122
000123
000124
000125
000126
000127
000128
000129
000130
000131
000132
000133
000134
000135
000136
000137
000138
000139
000140
000141
000142
000143
000144
000145
000146
000147
000148
000149
000150
000151
000152
000153
000154
000155
000156
000157
000158
000159
000160
000161
000162
000163
000164
000165
000166
000167
000168
000169
000170
000171
000172
000173
000174
000175
000176
000177
000178
000179
000180
000181
000182
000183
000184
000185
000186
000187
000188
000189
000190
000191
000192
000193
000194
000195
000196
000197
000198
000199
000200
000201
000202
000203
000204
000205
000206
000207
000208
000209
000210
000211
000212
000213
000214
000215
000216
000217
000218
000219
000220
000221
000222
000223
000224
000225
000226
000227
000228
000229
000230
000231
000232
000233
000234
000235
000236
000237
000238
000239
000240
000241
000242
000243
000244
000245
000246
000247
000248
000249
000250
000251
000252
000253
000254
000255
000256
000257
000258
000259
000260
000261
000262
000263
000264
000265
000266
000267
000268
000269
000270
000271
000272
000273
000274
000275
000276
000277
000278
000279
000280
000281
000282
000283
000284
000285
000286
000287
000288
000289
000290
000291
000292
000293
000294
000295
000296
000297
000298
000299
000300
000301
000302
000303
000304
000305
000306
000307
000308
000309
000310
000311
000312
000313
000314
000315
000316
000317
000318
000319
000320
000321
000322
000323
000324
000325
000326
000327
000328
000329
000330
000331
000332
000333
000334
000335
000336
000337
000338
000339
000340
000341
000342
000343
000344
000345
000346
000347
000348
000349
000350
000351
000352
000353
000354
000355
000356
000357
000358
000359
000360
000361
000362
000363
000364
000365
000366
000367
000368
000369
000370
000371
000372
000373
000374
000375
000376
000377
000378
000379
000380
000381
000382
000383
000384
000385
000386
000387
000388
000389
000390
000391
000392
000393
000394
000395
000396
000397
000398
000399
000400
000401
000402
000403
000404
000405
000406
000407
000408
000409
000410
000411
000412
000413
000414
000415
000416
000417
000418
000419
000420
000421
000422
000423
000424
000425
000426
000427
000428
000429
000430
000431
000432
000433
000434
000435
000436
000437
000438
000439
000440
000441
000442
000443
000444
000445
000446
000447
000448
000449
000450
000451
000452
000453
000454
000455
000456
000457
000458
000459
000460
000461
000462
000463
000464
000465
000466
000467
000468
000469
000470
000471
000472
000473
000474
000475
000476
000477
000478
000479
000480
000481
000482
000483
000484
000485
000486
000487
000488
000489
000490
000491
000492
000493
000494
000495
000496
000497
000498
000499
000500

100
101
102
103
104
105
106
107
108
109
110
111
112

000300
000340

100000
040000
020000
010000
004000
002000
001000
000400
000200
000100
000040
000020
000010
000004
000002
000001

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

.*"SWITCH REGISTER" SWITCH DEFINITIONS

SW15= 100000
SW14= 40000
SW13= 20000
SW12= 10000
SW11= 4000
SW10= 2000
SW09= 1000
SW08= 400
SW07= 200
SW06= 100
SW05= 40
SW04= 20
SW03= 10
SW02= 4
SW01= 2
SW00= 1

.EQUIV SW09, SW9
.EQUIV SW08, SW8
.EQUIV SW07, SW7
.EQUIV SW06, SW6
.EQUIV SW05, SW5
.EQUIV SW04, SW4
.EQUIV SW03, SW3
.EQUIV SW02, SW2
.EQUIV SW01, SW1
.EQUIV SW00, SW0

.*DATA BIT DEFINITIONS (BIT00 TO BIT15)

BIT15= 100000
BIT14= 40000
BIT13= 20000
BIT12= 10000
BIT11= 4000
BIT10= 2000
BIT09= 1000
BIT08= 400
BIT07= 200
BIT06= 100
BIT05= 40
BIT04= 20
BIT03= 10
BIT02= 4
BIT01= 2
BIT00= 1

.EQUIV BIT09, BIT9
.EQUIV BIT08, BIT8
.EQUIV BIT07, BIT7
.EQUIV BIT06, BIT6
.EQUIV BIT05, BIT5
.EQUIV BIT04, BIT4
.EQUIV BIT03, BIT3
.EQUIV BIT02, BIT2

```

113 .EQUIV BIT01,BIT1
114 .EQUIV BIT00,BIT0
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152

```

```

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

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

; *****
; HOOKS REQUIRED BY ACT11
$SVPC=. ; SAVE PC
.=46
$ENDAD ; ;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .SEOP
.=52
.WORD 0 ; ;2)SET LOC.52 TO ZERO
.= $SVPC ;: RESTORE PC

```

153
154
155
156
157
158
159
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

.SETTL COMMON TAGS

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

SCMTAG: . = 1100

\$PASS: .WORD 0
\$STNM: .BYTE 00
\$ERFLG: .BYTE 00
\$ICNT: .WORD 00
\$LPADR: .WORD 00
\$LPERR: .WORD 00
\$ERTTL: .WORD 00
\$ITEMB: .BYTE 0
\$ERMAX: .BYTE 1
\$ERRPC: .WORD 0
\$GDADR: .WORD 0
\$BDADR: .WORD 0
\$GDDAT: .WORD 0
\$BDDAT: .WORD 0
\$AUTOB: .BYTE 0
\$INTAG: .BYTE 0
\$SWR: .WORD DSWR
\$DISPLAY: .WORD DDISP
\$TKS: 177560
\$TKB: 177562
\$TPS: 177564
\$TPB: 177566
\$NULL: .BYTE 0
\$FILLS: .BYTE 2
\$FILLC: .BYTE 12
\$TPFLG: .BYTE 0
\$QUES: .ASCII /?/
\$CRLF: .ASCII <15>
\$LF: .ASCII <12>

;; START OF COMMON TAGS
;; CONTAINS PASS COUNT
;; CONTAINS THE TEST NUMBER
;; CONTAINS ERROR FLAG
;; CONTAINS SUBTEST ITERATION COUNT
;; CONTAINS SCOPE LOOP ADDRESS
;; CONTAINS SCOPE RETURN FOR ERRORS
;; CONTAINS TOTAL ERRORS DETECTED
;; CONTAINS ITEM CONTROL BYTE
;; CONTAINS MAX. ERRORS PER TEST
;; CONTAINS PC OF LAST ERROR INSTRUCTION
;; CONTAINS ADDRESS OF 'GOOD' DATA
;; CONTAINS ADDRESS OF 'BAD' DATA
;; CONTAINS 'GOOD' DATA
;; CONTAINS 'BAD' DATA
;; RESERVED--NOT TO BE USED
;; AUTOMATIC MODE INDICATOR
;; INTERRUPT MODE INDICATOR
;; ADDRESS OF SWITCH REGISTER
;; ADDRESS OF DISPLAY REGISTER
;; TTY KBD STATUS
;; TTY KBD BUFFER
;; TTY PRINTER STATUS REG. ADDRESS
;; TTY PRINTER BUFFER REG. ADDRESS
;; CONTAINS NULL CHARACTER FOR FILLS
;; CONTAINS # OF FILLER CHARACTERS REQUIRED
;; INSERT FILL CHARS. AFTER A "LINE FEED"
;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
;; QUESTION MARK
;; CARRIAGE RETURN
;; 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

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

001274
001275
001276
001277
001278
001279
001280
001281
001282
001283
001284
001285
001286
001287
001288
001289
001290
001291
001292
001293
001294
001295
001296
001297
001298
001299
001300
001301
001302
001303
001304
001305
001306
001307
001308
001309
001310
001311
001312
001313
001314
001315
001316
001317
001318
001319
001320
001321
001322
001323
001324
001325
001326
001327
001328
001329
001330
001331
001332
001333
001334
001335
001336
001337
001338
001339
001340
001341
001342
001343
001344
001345
001346
001347
001348
001349
001350
001351
001352
001353
001354
001355
001356
001357
001358
001359
001360
001361
001362
001363
001364
001365
001366
001367
001368
001369
001370
001371
001372
001373
001374
001375
001376
001377
001378
001379
001380
001381
001382
001383
001384
001385
001386
001387
001388
001389
001390
001391
001392
001393
001394
001395
001396
001397
001398
001399
001400

MSKTBL: .BYTE 377
.BYTE 177
.BYTE 077
.BYTE 037
.BYTE 017
.BYTE 007
.BYTE 003
.BYTE 001

BASE: .BYTE 0
.BYTE 50
.BYTE 120
.BYTE 170
.BYTE 240
.BYTE 303

CYLTBL: .BLKB 11

SECTBL: .BYTE 3
.BYTE 4
.BYTE 7
.BYTE 13

DRONT1: .BYTE 0
MODE: .BYTE 00
PRONUM: .BYTE 00
DRIVE: .BYTE 00
CYLBAS: .BYTE 00
COMND: .BYTE 00
WRTNBY: .BYTE 00
HDRFLG: .BYTE 00
ECNT: .BYTE 00
CNTSIN: .BYTE 00
TIMR2: .BYTE 00
IDEX: .BYTE 00
STFLG: .BYTE 0
OSPFLG: .BYTE 0

.EVEN

KYTEMP: .WORD 0
CONTRL: .WORD 00
DSKADR: .WORD 00
BUSADR: .WORD 00
WRDCNT: .WORD 0
CYLCNT: .WORD -6000
SECCNT: .WORD -400
TIMR: .WORD 0
CHKCNT: .WORD 0
DSKTMP: .WORD 0
WRITCS: .WORD 4003
READCS: .WORD 5
ERRFLG: .WORD 0
PATTPN: .WORD 0
RKDS: .WORD 177400

:TABLE OF CYLINDER BASE FOR AUTO MODE

:CYL 0 BASE CYLINDER ADDRESS
:CYL 40 BASE CYLINDER ADDRESS
:CYL 80 BASE CYLINDER ADDRESS
:CYL 120 BASE CYLINDER ADDRESS
:CYL 160 BASE CYLINDER ADDRESS
:CYL 195 BASE CYLINDER ADDRESS

:TABLE OF SELECTED BASES

:SECTOR 3
:SECTOR 4
:SECTOR 7
:SECTOR 12

:COUNT OF NUMBER OF DRIVES ON SYS. 1
:IF -1 START 210 SELECTED
:IF 0 1 PROCESSOR SELECTED
:DRIVE # UNDER TEST (MAN+AUTO MODE)
:BASE SELECTED (MANUAL MODE)
:IF 0 WRITE COMMAND
:DRIVE WHICH DID WRITE (READ OPERATION)
:FLAG FOR ONE HEADER PRINTOUT
:ERROR COUNTER
:SEEK INCOM. COUNTER
:SECOND PASS TIMER
:CURRENT INDEX #

:TEMP. KEYBOARD BUFFER
:TEMP. CONTROL+STATUS WORD
:TEMP. DISK ADDRESS WORD
:TEMP. BUS ADDRESS WORD
:TEMP. WORD COUNT
:WORD COUNT OF 1 CYLINDER
:WORD COUNT OF 1 SECTOR
:TIMER FOR OPERATIONS
:NUMBER OF ERROR PRINTOUTS
:SAVE OF CURRENT DISK #
:IBA+WRITE+GO
:READ+GO
:ERROR FLAG INHIBIT ADDRESS CHANGE
:DATA PATTERN

.SBTTL ERROR POINTER TABLE

:*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
:*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
:*LOCATION \$ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
:*NOTE1: IF \$ITEMB IS 0 THE ONLY PERTINENT DATA IS (\$ERRPC).
:*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:

- * EM ::POINTS TO THE ERROR MESSAGE
- * DH ::POINTS TO THE DATA HEADER
- * DT ::POINTS TO THE DATA
- * DF ::POINTS TO THE DATA FORMAT

\$ERRTB:

::*****
:THE ERROR TABLE IS UNUSED IN THIS PROGRAM
::~*****

001434

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
114
115
116
117
118
119
120
121
122
123
124
125

001760	104401	001766
001764	000423	
002034		
002034	104401	002042
002040	000421	
002104		
002104	104401	002112
002110	000421	
002154		
002154	104401	002162
002160	000421	
002224		
002224	104401	002232
002230	000421	
002274		
002274	104401	002302
002300	000421	
002344		
002344	104401	002352
002350	000421	
002414		
002414	104401	002422
002420	000421	
002464		
002464	104401	002472
002470	000421	
002534		
002534		
002534	104401	002542
002540	000405	
002554		
002554	104411	
002556	012600	
002560	022700	000010
002564	003403	
002566	006100	
002570	000170	002602
002574	104401	001160
002600	000755	
002602	001754	
002604	002622	
002606	010350	
002610	011762	

```

TYPE 655 ::TYPE ASCIZ STRING
BR 645 ::GET OVER THE ASCIZ
::655: .ASCIZ <15><12><15><12> NAME TYPE
645:
TYPE 675 ::TYPE ASCIZ STRING
BR 665 ::GET OVER THE ASCIZ
::675: .ASCIZ <15><12>/INDEX 0/
665:
TYPE 695 ::TYPE ASCIZ STRING
BR 685 ::GET OVER THE ASCIZ
::695: .ASCIZ <15><12>/COMPATIBILITY PACKAGE 1/
685:
TYPE 715 ::TYPE ASCIZ STRING
BR 705 ::GET OVER THE ASCIZ
::715: .ASCIZ <15><12>/OSCILLATING SEEK PACKAGE 2/
705:
TYPE 735 ::TYPE ASCIZ STRING
BR 725 ::GET OVER THE ASCIZ
::735: .ASCIZ <15><12>/FORMATTER-SURFACE VERIFIER 3/
725:
TYPE 755 ::TYPE ASCIZ STRING
BR 745 ::GET OVER THE ASCIZ
::755: .ASCIZ <15><12>/RK05 CONTROL PANEL TEST 4/
745:
TYPE 775 ::TYPE ASCIZ STRING
BR 765 ::GET OVER THE ASCIZ
::775: .ASCIZ <15><12>/RK05 CONTROL PANEL TEST #2 5/
765:
TYPE 795 ::TYPE ASCIZ STRING
BR 785 ::GET OVER THE ASCIZ
::795: .ASCIZ <15><12>/HEAD ALIGNMENT ROUTINE 6/
785:
TYPE 815 ::TYPE ASCIZ STRING
BR 805 ::GET OVER THE ASCIZ
::815: .ASCIZ <15><12>/POWER FAILURE (WRITE) TEST 7/
805:
TABLTY:
TYPE 655 ::TYPE ASCIZ STRING
BR 645 ::GET OVER THE ASCIZ
::655: .ASCIZ <15><12><15><12>/TYPE=
645:
RDOCT
MOV (SP)+,R0 :GET THE TEST NUMBER FROM THE OPERATOR
CMP #10,R0 :STORE IT IN R0
BLE NG :VALID NUMBER ?
ROL R0 :BR IF NOT
JMP @BEGIN(R0) :ALIGN THE NUMBER FOR DISPATCHING
NG: TYPE @SQUES :GO TO THE SELECTED TEST
BR TABLTY :?

;TEST ENTRY DISPATCH TABLE
BEGIN: STRT1
SECT.3
SECT.2
SECT.1

```

G03

426 002612 013744
 427 002614 017154
 428 002616 020510
 429 002620 021402

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

.SBTTL COMPATIBILITY TEST

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

430 002622 000240
 431 002624 104401 002632
 432 002624 000415
 433 002630 000415
 434 002664 104401 002672
 435 002664 000417
 436 002730
 437 002730 104410
 438 002732 012600
 439 002734 012701 001166
 440 002740 105037 001311
 441 002744 005037 001330
 442 002750 112037 001330
 443 002754 122737 000054 001330
 444 002762 001770
 445 002764 162737 000060 001330
 446 002772 100403
 447 002774 004737 004052
 448 003000 000761
 449 003002 122740 000056
 450 003006 001402
 451 003010 004737 004122
 452 003014 022701 001206
 453 003020 001403
 454 003022 012721 100000
 455 003026 000772

```

SECT.3: NOP ;NO-OP
AUTSL2:
  TYPE 655 ;:TYPE ASCIZ STRING
  BR 645 ;:GET OVER THE ASCIZ
::655: .ASCIZ <15><12>>TERMINATE WITH '<CR>'
645:
  TYPE 675 ;:TYPE ASCIZ STRING
  BR 665 ;:GET OVER THE ASCIZ
::675: .ASCIZ <15><12>>DRIVE NUMBERS ON SYSTEM 1=
665:
  RDLIN
  MOV (SP)+,R0 ;:PICK UP THE ADDRESS OF THE INPUT BUFFER
  MOV #LOGA,R1 ;:GET THE ADDRESS OF THE LOGICAL UNIT 'BL.
  CLRB @DRCNT1 ;:CLEAR THE DRIVE COUNTER
  CLR @KYTEMP ;:CLEAR TEMP
  15: MOVB (R0)+,@KYTEMP ;:GET THE FIRST DRIVE #
  CMPB #54,@KYTEMP ;:IS IT A COMMA THAT WAS TYPED?
  BEQ 15 ;:IF YES GO BACK
  SUB #60,@KYTEMP ;:MAKE ASCII A DRIVE #
  BMI 25 ;:IF RESULT NEGATIVE BRANCH
  JSR PC,STORE ;:IF RESULT POSITIVE JUMP
  BR 15 ;:AFTER STORING GET NEXT #
  25: CMPB #56,-(R0) ;:WAS NEGATIVE RESULT A TERMINATOR?
  BEQ 35 ;:IF YES BRANCH
  JSR PC,ILEGAL ;:IF NO BAD CHARACTER JUMP
  35: CMP #DRVD,R1 ;:IS THE TABLE FULL
  BEQ SECSYS ;:IF YES BRANCH
  MOV #100000,(R1)+ ;:IF NO FILL TABLE WITH DOWN INDICATOR.
  BR 35 ;: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

470
 471 003030
 472 003030 104401 003036
 473 003034 000416
 474
 475 003072
 476 003072 000240
 477 003074 104407
 478 003076 012637 001330
 479 003102 104401 001330
 480 003106 022737 000131 001330
 481 003114 001411

```

SECSYS:
  TYPE 655 ;:TYPE ASCIZ STRING
  BR 645 ;:GET OVER THE ASCIZ
::655: .ASCIZ <15><12>>IS THERE A SECOND SYSTEM?
645:
  NOP ;***
  RDCHR ;:READ A CHARACTER
  MOV (SP)+,@KYTEMP ;:GET THE RESPONSE
  TYPE ,KYTEMP ;:ECHO
  CMP #131,@KYTEMP ;:WAS IT A "Y" (FOR YES)?
  BEQ PROC2 ;:IF YES BRANCH TO PROCESSOR 2
  
```

```

0000 003116 022737 000116 001330      CMP      #16,2#KYTEMP      :WAS RESPONSE LEGAL IN FOR NO.
0001 003124 001460                      BEQ      PR01             :IF LEGAL BRANCH
0002 003126 104401 003134      TYPE    ,67$             :TYPE ASCIZ STRING
0003 003132 000401                      BR       66$             :GET OVER THE ASCIZ
0004                                ;:67$:
0005                                66$:
0006 003136                      BR       SECSYS          :GO BACK ASK AGAIN
0007 003136 000734                      9:5B      #377,2#PRONUM  :SET FLAG TWO PROCESSORS
0008 003140 152737 000377 001313  PR02:  NOP                    :***
0009 003146 000240                      TYPE    ,65$            :TYPE ASCIZ STRING
0010 003150 104401 003156      BR       64$            :GET OVER THE ASCIZ
0011 003154 000406                      ;:65$:
0012                                64$:
0013 003172                      RDCHR                    :READ A CHARACTER
0014 003172 104407                      MOV      (SP)+,2#KYTEMP  :PICK UP THE RESPONSE
0015 003174 012637 001330      TYPE    ,KYTEMP         :ECHO
0016 003200 104401 001330      SUB     #60,2#KYTEMP     :MAKE IT A NUMBER
0017 003204 162737 000060 001330  BMI     BADINP           :IF NOT A NUMBER, BRANCH
0018 003212 100420                      CMP     #10,2#KYTEMP     :IS IT A LEGAL #?
0019 003214 022737 000010 001330  BLE     BADINP           :IF NO BRANCH
0020 003222 003414                      SWAB    2#KYTEMP        :GET THE DRIVE # TO THE HIGH BYTE
0021 003224 000337 001330      BIS     #BIT14,2#KYTEMP  :SET THE SECOND SYSTEM BIT
0022 003230 052737 040000 001330  MOVB   2#DRCNT1,R5      :GET THE DRIVE COUNT
0023 003236 113705 001311      ROL     R5               :MAKE IT AN INDEX
0024 003242 006105                      MOV     2#KYTEMP,LOGA(R5) :STORE THE SYSTEM #2 WORD
0025 003244 013765 001330 001166  BR     GO                :GO DO THE TEST
0026 003252 000407                      BADINP:
0027 003254                                TYPE    ,65$            :TYPE ASCIZ STRING
0028 003254 104401 003262      BR     64$            :GET OVER THE ASCIZ
0029 003260 000401                      ;:65$:
0030                                64$:
0031 003264                                BR     PR02             :GO BACK ASK AGAIN
0032 003264 000725 001313  PR01:  CLRB   2#PRONUM        :CLEAR THE FLAG ONE PROCESSOR
0033 003266 105037
0034                                ;THIS IS THE ACTUAL PROGRAM
0035
0036 003272 012700 001166      GC:    MOV     #LOGA,RO   :GET THE TABLE ADDRESS TO RO
0037 003276 105037 001324      CLRB   2#IDEX          :CLRB THE INDEX
0038 003302 000405      BR     GO2             :BRANCH AROUND INCREMENT ROUTINE
0039
0040 003304 062700 000002      G01:   ADD     #2,RO      :INDEX THRU THE TABLE
0041 003310 022700 001206      CMP     #DRVO,RO       :DONE?
0042 003314 001414                      BEQ     EXIT           :IF YES GET OUT
0043 003316 005710      G02:   TST     (RO)      :IS THE DRIVE ACTIVE
0044 003320 100001                      BPL    GO3            :IF YES BRANCH
0045 003322 000770                      BR     GO1            :NO TRY THE NEXT ONE
0046 003324 011037 001164      G03:   MOV     (RO),2#DRACTV :PICK UP THE ACTIVE DRIVE WORD
0047 003330 004737 004154      JSR    PC,CYCLE        :CALL CYCLE (PICK UP DRIVE #, CYL BASE, AND CALL MOUNT)
0048 003334 004737 005064      JSR    PC,WRLINK       :CALL WRITE LINK TO LOAD REGISTERS FOR WRITE
0049 003340 004737 005612      JSR    PC,RDLINK       :CALL READ LINK TO LOAD REGISTERS FOR READ
0050 003344 000757                      BR     GC1            :GO GET NEXT DRIVE
0051 003346 012700 001166      EXIT:  MOV     #LOGA,RO
0052 003352 022700 001206  EXTFR2: CMP     #DRVO,RO
0053 003356 001414                      BEQ     EXITX
0054 003360 032710 040000      BIT     #BIT14,(RO)
0055 003364 001011                      BNE    EXITX

```

```

538 003366 005720
539 003370 100770
540 003372 014001
541 003374 004737 00423C
542 003400 004737 005612
543 003404 005720
544 003406 000761
545 003410
546 003410 104401 003416
547 003414 000404
548
549 003426
550 003426 000137 00144C
551
552
553
554
555 003432
556 003432 104401 00344C
557 003436 000415
558
559 003472
560 003472 012704 000200
561 003476 012703 000001
562 003502 012702 001166
563 003506
564 003506 104401 003514
565 003512 000405
566
567 003526
568 003526 000240
569 003530 010346
570 003532 104403
571 003534 006
572 003535 000
573 003536 104401 003544
574 003542 000401
575
576 003546
577 003546 104411
578 003550 012600
579 003552 110001
580 003554 000300
581 003556 042700 177400
582 003562 042701 177400
583 003566 006000
584 003570 103003
585 003572 052700 000200
586 003576 000241
587 003600 006001
588 003602 103002
589 003604 052701 000200
590 003610 110162 000001
591 003614 004737 004010
592 003620 110062 000001
593 003624 004737 004010

```

```

TST (R0)+
BMI EXTR2
MOV -(R0),R1
JSR PC,D02
JSR PC,RD_LNK
TST (R0)+
BR EXTR2

EXITX:
TYPE 65$ ;;TYPE ASCIZ STRING
BR 64$ ;;GET OVER THE ASCIZ
65$: .ASCIZ <15><12>/DONE!/
64$:
JMP 2*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
TYP0S
.BYTE 6
.BYTE 0
TYPE 67$ ;;TYPE ASCIZ STRING
BR 66$ ;;GET OVER THE ASCIZ
67$: .ASCIZ /=/
66$:
RDOCT
MOV (SP)+,R0 ;PICK UP THE OCTAL WORD
MOVB R0,R1 ;GET THE FIRST DRIVE TO R1
SWAB R0 ;GET THE SECOND DRIVE TO R0 LOW BYTE
BIC #177400,R0 ;CLEAR THE UNUSED BITS
BIC #177400,R1 ;CLEAR THE UNUSED BITS
ROR R0 ;ROTATE RIGHT R0
BCC 2$ ;IF CARRY IS CLEAR BRANCH
BIS #BIT7,R0 ;SET DOWN BIT IF CARRY SET
CLC ;AND CLEAR THE CARRY BIT
ROR R1 ;NOW DO THE SAME FOR R1
BCC 3$ ;IF NO ERROR, BRANCH
BIS #BIT7,R1 ;IF ERROR SET THE BIT
MOVB R1,1(R2) ;SET DRIVE FIRST IN TABLE
JSR PC,MASKER ;CALL THE MASK CONTROL SUBROUTINE
MOVB R0,1(R2) ;SET DRIVE SECOND IN TABLE (NEXT WORD)
JSR PC,MASKER ;CALL THE MASK CONTROL SUBROUTINE

```

```

594 003630 005203 INC R3 ; INCREMENT THE WORD COUNTER
595 003632 000725 BR INSYS2 ; GO BACK AND GET NEXT WORD
596 003634 050412 EXITA: BIS R4,(R2) ; FILL THE TABLE (LAST WORD)
597 003636 006004 ROR R4 ; WITH ALL BITS SET
598 003640 103375 BCC EXITA ; GO BACK IF NOT DONE
599 003642 042712 174000 EXITB: BIC #174000,(R2) ; CLEAR DOWN AND SECOND SYSTEM BITS AT TABLE
600 003646 010200 MOV R2,R0 ; GET ADDRESS TO RD (CURRENT TABLE)
601 003650 005722 TST (R2)+ ; ADD 2 TO THE POINTER
602 003652 022702 001206 FIL.DN: CMP #DRVD,R2 ; TABLE FULL?
603 003656 001403 BEQ 2$ ; IF YES BRANCH
604 003660 012722 100000 MOV #BIT15,(R2)+ ; FILL THE TABLE
605 003664 000772 BR FIL.DN ; GO BACK TRY AGAIN
606 003666 011001 2$: MOV (R0),R1 ; GET THE WORD TO R1
607 003670 110102 MOVB R1,R2
608 003672 004737 005214 JSR PC,MASK ; FORM DRIVE # FOR MOUNT
609 003676 004737 004230 JSR PC,D02 ; WRITE NEW INFO
610 003702 004737 005064 JSR PC,WRLINK ; READ SAMPLE (ALL DRIVES)
611 003706 004737 005612 JSR PC,RDLINK ; READ SAMPLE (ALL DRIVES)
612 003712 104401 003720 TYPE ,65$ ; TYPE ASCIZ STRING
613 003716 000427 BR 64$ ; GET OVER THE ASCIZ
614 65$: .ASCIZ <15><12>/DONE SYSTEM 2 RESTART SYSTEM 1, TYPE WORD /
615 64$: MOV (R0),-(SP) ; GET WORD FOR SYSTEM 1 AND TYPE
616 003776 011046 TYPOS .BYTE 6
617 004000 104403 .BYTE 1
618 004002 006 1$: HALT ; HALT (FINISHED SYSTEM #2)
619 004003 001 ; GO BACK TO HALT
620 004004 000000 BR 1$
621 004006 000776
622
623 ; THIS ROUTINE SETS UP THE MASK BITS IN THE LOG TABLE FOR
624 ; SYSTEM #2. IF A DRIVE IS DOWN NO BIT IS SET BUT THE MASK
625 ; IS SHIFTED.
626
627 004010 005712 MASKER: TST (R2) ; IS THE DRIVE UP
628 004012 100401 BMI RETRN4 ; IF NO, BRANCH
629 004014 110412 MOVB R4,(R2) ; MOVE THE MASK BIT IN THE TABLE
630 004016 006004 RETRN4: ROR R4 ; ROTATE THE MASK
631 004020 103003 BCC 1$ ; DONE?, IF NO, BRANCH
632 004022 012716 003642 MOV #EXITB,(SP) ; SET UP FOR RETURN
633 004026 000207 RTS PC
634 004030 032712 040000 1$: BIT #BIT14,(R2) ; IS THIS SYSTEM # 2'S DRIVE?
635 004034 001403 BEQ 2$ ; IF NO, BRANCH
636 004036 012716 003634 MOV #EXITA,(SP) ; SET UP FOR RETURN
637 004042 000207 RTS PC
638 004044 062702 000002 2$: ADD #2,R2 ; INDEX THRU LOGA TABLE
639 004050 000207 RTS PC ; RETURN
640
641 ; ROUTINE TO BUILD THE ACTIVE LOGICAL UNIT TABLE
642
643 004052 022737 000010 001330 STORE: CMP #10,@#KYTEMP ; IS INPUT A LEGAL NUMBER?
644 004060 000240 NOP ; ***
645 004062 003417 BLE ILEGAL ; IF NOT BRANCH
646 004064 000337 001330 SWAB @#KYTEMP ; ALIGN DRIVE # FOR TABLE
647 004070 013721 001330 MOV @#KYTEMP,(R1)+ ; PUT THE WORD IN THE TABLE
648 004074 005037 001330 CLR @#KYTEMP ; CLEAR THE TEMP WORD
649 004100 105237 001311 INCB @#DRCNT ; INCREMENT THE COUNTER

```

```

650 004104 022701 001206      CMP      #DRVD,R1      ;IS THE TABLE FULL?
651 004110 001401      BEQ      TBLFUL      ;IF YES BRANCH
652 004112 000207      RTS      PC          ;IF NO RETURN
653 004114 012716 003030      TBLFUL: MOV     #SECSYS,(SP) ;IF TABLE FULL SET UP FOR SYSTEM #2
654 004120 000207      RTS      PC          ;RETURN
655 004122 012716 002624      ILEGAL: MOV     #AUTSL2,(SP) ;IF ILLEGAL RESPONSE, GO BACK
656 004126 104401 004134      TYPE    ,65$        ;TYPE ASCIZ STRING
657 004132 000407      BR      64$         ;GET OVER THE ASCIZ
658      ;:65$: .ASCIZ /ILLEGAL INPUT/
659 004152      64$:
660 004152 000207      RTS      PC          ;RETURN
661
662      ;THIS ROUTINE GETS A DRIVE # AND BUILDS A TABLE OF
663      ;CYLINDER ADDRESS FOR USE BY WRITE. (RO)=ADDRESS OF ACTIVE
664      ;WORD IN LOGICAL TABLE. IT ALSO SETS AND CLEARS THE MASK BITS
665      ;OF THE TABLE AS OPERATIONS INDICATE
666
667 004154 011001      CYCLE:  MOV     (RO),R1 ;GET THE LOGICAL UNIT ACTIVE WORD TO R1
668 004156 032701 040000      BIT     #BIT14,R1   ;IS IT ON SYSTEM #2
669 004162 001402      BEQ      CYCL2      ;IF NO BRANCH
670 004164 000137 006732      JMP     @#SECCONE  ;GO TO SYSTEM #2
671 004170 113703 001324      CYCL2:  MOVVB  @#IDEX,R3 ;GET THE INDEX VALUE
672 004174 116302 001256      MOVVB  MSKTBL(R3),R2 ;GET THE MASK TO R2
673 004200 004737 005214      CYCLE2: JSR     PC,MASK ;CALL THE MASK SUBROUTINE
674 004204 012703 001166      LDPLG:  MOV     #LOGA,R3 ;GET TABLE ADDRESS TO R3
675 004210 020003      2$:    CMP     R0,R3    ;IS ACTIVE ADDRESS=TO FIRST ADDRESS
676 004212 001002      BNE     3$         ;IF NO BRANCH
677 004214 050223      BIS     R2,(R3)+   ;IF YES SET BITS TO SHOW WRITE
678 004216 000401      BR      4$         ;BRANCH TO SEE IF DONE
679 004220 040223      3$:    BIC     R2,(R3)+  ;CLEAR BITS TO SHOW OVER-WRITE
680 004222 022703 001206      4$:    CMP     #DRVD,R3 ;DONE
681 004226 001370      BNE     2$         ;IF NO GO BACK
682 004230 000301      D02:   SWAB    R1    ;GET DRIVE # TO LOW BYTE
683 004232 000240      NOP
684 004234 110102      MOVVB  R1,R2      ;GET IT TO R2
685 004236 042702 000370      BIC     #370,R2   ;CLEAR THE UNUSED BITS
686 004242 110237 001314      MOVVB  R2,@#DRIVE ;GET THE DRIVE #
687 004246 006102      ROL    R2        ;SHIFT THE DRIVE # TO
688 004250 006102      ROL    R2        ;ALIGN IT FOR THE DRIVE ADDR.
689 004252 006102      ROL    R2        ;KEEP IT MOVING!
690 004254 006102      ROL    R2        ;A LITTLE MORE!
691 004256 006102      ROL    R2        ;THERE IT IS
692 004260 110237 001353      MOVVB  R2,@#DSKTMP+1 ;GET IT TO DISK ADDR. TEMP.
693 004264 110237 001335      MOVVB  R2,@#DSKADR+1 ;CALL MOUNT
694 004270 004737 004276      JSR    PC,MOUNT  ;RETURN
695 004274 000207      RTS
696
697 004276 105237 001324      MOUNT:  INCB   @#IDEX ;INCREMENT THE INDEX
698 004302 000240      NOP
699 004304 112737 000005 001321      MOVVB  #5,@#ECNT  ;SET ERROR CNTR TO 5
700 004312 112737 000003 001322      MOVVB  #3,@#CNTSIN ;SET SIN CNTR TO 3
701 004320 104401 004326      TYPE    ,65$        ;TYPE ASCIZ STRING
702 004324 000414      BR      64$         ;GET OVER THE ASCIZ
703      ;:65$: .ASCIZ <15><12>/MOUNT PACK ON DRIVE #/
704 64$:
705 004356 013746 001314      MOV    DRIVE,-(SP) ;SAVE DRIVE FOR TYPEOUT

```

```

706 004362 104403          TYPOS          ;;GO TYPE--OCTAL ASCII
707 004364          001          .BYTE 1          ;;TYPE 1 DIGIT(S)
708 004365          000          .BYTE 0          ;;SUPPRESS LEADING ZEROS
709 004366 104401 004374    TYPE 67$        ;;TYPE ASCIZ STRING
710 004372 000415          BR 66$         ;;GET OVER THE ASCIZ
711          ;;67$: .ASCIZ <15><12>/MAKE PACK WRITE ENABLE/
712 004426          66$:
713 004426 104401 004434    TYPE 69$        ;;TYPE ASCIZ STRING
714 004432 000420          BR 68$         ;;GET OVER THE ASCIZ
715          ;;69$: .ASCIZ <15><12>/PRESS CONTINUE WHEN DRIVE RDY/
716 004474          68$:
717 004474 000000          HALT
718 004476 004737 004504    JSR PC,INITIL  ;CALL INITIALIZER
719 004502 000207          RTS PC         ;RETURN

;;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

724 004504 010046          INITIL: MOV RO, -(SP) ;SAVE RO
725 004506 013700 001334    MOV 2#0SKADR,RO ;GET THE DRIVE # TO RO
726 004512 042700 001777    BIC 2#1777,RO  ;CLEAR THE UNUSED BITS
727 004516 005037 001346    INITI2: CLR 2#TIMR ;CLEAR THE TIMER
728 004522 105037 001323    CLRB 2#TIMR2
729 004526 000240          NOP
730 004530 010077 174642    MOV RO,2#RKDA ;***
731 004534 012777 000001 174626 MOV 2#1,2#RKCS ;GET DRIVE # TO 'DA' REGISTER
732 004542 004737 005574    JSR PC,SMTME  ;ISSUE CONTROL RESET + GO
733 004546 105777 174616    1$: TSTB 2#RKCS ;DID CONTROL READY SET
734 004552 100423          BMI 2$        ;IF YES BRANCH
735 004554 005237 001346    INC 2#TIMR    ;IF NO INCREMENT THE TIMER
736 004560 001372          BNE 1$        ;IF TIMER NOT ZERO BRANCH
737 004562 104401 004570    TYPE 65$      ;;TYPE ASCIZ STRING
738 004566 000415          BR 64$        ;;GET OVER THE ASCIZ
739          ;;65$: .ASCIZ <15><12>/CONTROL RESET TIME OUT/
740 004622          64$:
741 004622 010077 174550    2$: MOV RO,2#RKDA ;DRIVE NUMBER TO 'DA' REG.
742 004626 105777 174532    TSTB 2#RKDS  ;IS DRIVE READY
743 004632 100415          BMI 3$        ;IF YES BRANCH
744 004634 104401 004642    TYPE 67$      ;;TYPE ASCIZ STRING
745 004640 000411          BR 66$        ;;GET OVER THE ASCIZ
746          ;;67$: .ASCIZ <15><12>/DRIVE NOT READY/
747 004664          66$:
748 004664 000714          BR INITI2     ;GO BACK TRY AGAIN
749 004666 032777 000040 174470 3$: BIT 2#BITS,2#RKDS ;IS DRIVE WRITE LOCKED?
750 004674 001420          BEQ 4$        ;IF NO, BRANCH
751 004676 104401 004704    TYPE 69$      ;;TYPE ASCIZ STRING
752 004702 000414          BR 68$        ;;GET OVER THE ASCIZ
753          ;;69$: .ASCIZ <15><12>/DRIVE WRITE PROTECTED/
754 004734          68$:
755 004734 000670          BR INITI2     ;YES, GO BACK TRY AGAIN
756 004736 005037 001346    4$: CLR 2#TIMR    ;CLEAR THE TIMER
757 004742 010077 174430    MOV RO,2#RKDA ;GET THE DRIVE # TO 'DA' REGISTER
758 004746 012777 000015 174414 MOV 2#15,2#RKCS ;ISSUE DRIVE RESET + GO
759 004754 004737 005574    JSR PC,SMTME
760 004760 105777 174404    5$: TSTB 2#RKCS
761 004764 100375          BPL 5$
    
```

```

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

```

818 005240 001401          BEQ      2$          ; IF NO BRANCH
819 005242 110425          MOVVB   R4,(R5)+      ; MOVE THE CYLINDER BASE TO THE TABLE
820 005244 105204          2$: INCB   R4          ; INCREMENT THE BASE
821 005246 006003          ROR    R3          ; ROTATE THE COMPARE MASK
822 005250 103372          BCC    1$          ; IF NOT DONE GO BACK
823 005252 112715 000377  MOVVB   #377,(R5)    ; IF DONE LOAD FINISH FLAG
824 005256 012605          MOV    (SP)+,R5     ; RESTORE R5
825 005260 000207          RTS    PC          ; RETURN TO CALLER
826
827
828 ; THIS ROUTINE FORMS A CYLINDER ADDRESS FOR BOTH THE READ AND WRITE ROUTINES
829 ; WHEN THE BASE TABLE IS FULLY INDEXED IT RETURNS TO PC+2 OF CALLER
830 005262 012703 001266  CYLADR: MOV    #BASE,R3      ; GET THE CYLINDER TABLE ADDRESS
831 005266 012702 001274  CYLADR2: MOV   #CYLTBL,R2   ; GET THE SELECTED CYL BASE ADDR.
832 005272 111204          CYLOFF: MOVVB  (R2),R4     ; GET THE SELECTED CYL VALUE TO R4
833 005274 000240          NOP
834 005276 122704 000377  CMPB   #377,R4      ; IS IT THE TABLE TERMINATOR?
835 005302 001416          BEQ    BASINC      ; IF YES BRANCH
836 005304 005046          CLR    -(SP)      ; INSURE CLEAN WORD
837 005306 111316          MOVVB  (R3),(SP)    ; GET THE CYL ADDRESS ON THE STACK
838 005310 062604          ADD   (SP)+,R4    ; AND IT TO THE SELECTED OFFSET
839 005312 006104          ROL   R4          ; SHIFT THIS RESULT
840 005314 006104          ROL   R4          ; TO ALIGN THE NEWLY FORMED
841 005316 006104          ROL   R4          ; CYLINDER ADDRESS WITH BITS
842 005320 006104          ROL   R4          ; 5 THRU 12 OF RKDA AND
843 005322 006104          ROL   R4          ; STORE THIS IN DSKADR
844 005324 042737 017777 001334  BIC    #017777,@#DSKADR ; CLEAR ALL BUT DRIVE NUMBER
845 005332 050437 001334  BIS    R4,@#DSKADR   ; PUT IT IN DSKADR
846 005336 000207          RTS    PC
847 005340 005203          BASINC: INC   R3    ; PICK UP ADDRESS OF NEXT BASE CYL.
848 005342 000240          NOP
849 005344 022703 001274  CMP    #CYLTBL,R3   ; ARE YOU FINISHED?
850 005350 001401          BEQ    RETRN3      ; IF YES, BRANCH
851 005352 000745          BR    CYLADR2     ; NO GO BACK
852 005354 062716 000002  RETRN3: ADD   #2,(SP) ; SET-UP FOR PC+2
853 005360 000207          RTS    PC          ; RETURN
854
855 ; ROUTINE TO PERFORM INDICATED FUNCTION AND CHECK FOR
856 ; DONE AND ERRORS
857
858 005362 005037 001346  EXECUT: CLR    @#TIMR   ; CLEAR THE TIMER
859 005366 000240          NOP
860 005370 105037 001323  CLRB   @#TIMR2     ; HALT HERE TO CHECK COMMAND ABOUT TO BE EXECUTED
861 005374 013777 001334 173774  MOV    @#DSKADR,@#RKDA ; CLEAR SECOND TIMER
862 005402 013777 001336 173764  MOV    @#BUSADR,@#RKBA ; LOAD THE DISK ADDRESS REGISTER
863 005410 013777 001340 173754  MOV    @#WRDCNT,@#RKWC ; LOAD THE BUS ADDRESS REGISTER
864 005416 013777 001332 173744  MOV    @#CONTRL,@#RKCS ; LOAD THE WORD COUNT REGISTER
865 005424 004737 005574          JSR   PC,SMTME    ; LOAD THE CONTROL REGISTER
866 005430 105777 173734  CHECK1: TSTB   @#RKCS   ; KILL TIME FOR RK11-C
867 005434 100011          BPL   TIME        ; IS CONTROL READY SET
868 005436 004737 006062  JSR   PC,ERRCHK   ; IF NO BRANCH
869 005442 005737 001360  TST    @#ERRFLG    ; ERROR?
870 005446 001403          BEQ    1$         ; IF NO BRANCH
871 005450 005037 001360  CLR    @#ERRFLG    ; CLEAR THE FLAG
872 005454 000742          BR    EXECUT      ; TRY AGAIN
873 005456 000207          1$: RTS    PC
  
```

```

005460 005237 001346
005464 000240
005466 001360
005470 105737 001323
005474 001003
005478 105237 001323
005502 000752
005504 004737 004504
005510 104401 005516
005514 000426
005572 000673
005574 012737 000500 001346
005602 005337 001346
005606 001375
005610 000207
005612 010046
005614 000240
005616 152737 000377 001316
005624 012701 001166
005630 012705 001305
005634 002405
005636 062701 000002
005642 022701 001206
005646 001503
005650 005711
005652 100001
005654 000770
005656 011100
005660 010002
005662 000300
005664 042700 177770
005670 110037 001317
005674 006100
005676 016037 001206 001362
005704 004737 005214
005710 004737 005262
005714 000401
005716 000747
005720 053737 001352 001334
005722 152537 001334
005732 012737 007336 001336
005740 000240
005742 013737 001344 001340
005750 013737 001356 001332
005756 004737 005362
005762 004737 006430
005766 042737 000017 001334
005774 022705 001311

```

```

TIME: INC @@TIMR
NOP
BNE CHECK1 :***
TSTB @@TIMR2 :SECOND TIMEOUT?
BNE IS :IF YES BRANCH
INCB @@TIMR2 :INDICATE SECOND TIMEOUT
BR CHECK1 :GO BACK
IS: JSR PC,INITIL
TYPE 655 :TYPE ASCII STRING
BR 645 :GET OVER THE ASCII
:655: .ASCII (<15><12> TIMED OUT ON OPERATION RETRY IN PROGRESS)
645: BR EXECUT
SMTME: MOV @500,@@TIMR
IS: DEC @@TIMR
BNE IS
RTS PC

```

```

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

```

```

ROLINK: MOV RO,-(SP) :SAVE RO
NOP :***
BISB @377,@@COMND :INDICATE READ OPERATION
MOV @LOGA,R1 :GET THE TABLE ADDRESS TO R1
MOV @SECTBL,R5 :GET THE SECTOR TABLE ADDRESS
BR RD2 :SKIP OVER THE INDEX, FIRST PASS
RD1: ADD @2,R1 :ADD 2 TO THE ADDRESS
CMP @DRVD,R1 :ARE YOU THRU THE ENTIRE TABLE
BEQ EXIT2 :IF YES EXIT
RD2: TST (R1) :IS THE DRIVE ACTIVE
BPL RD3 :IF YES BRANCH
BR RD1 :IF NO GET NEXT WORD
RD3: MOV (R1),RO :GET THE ACTIVE WORD TO RO
MOV RO,R2 :COPY THE WORD
SWAB RO :GET THE DRIVE # TO THE LOW BYTE
BIC @177770,RO :CLR ALL BUT THE DRIVE #
MCOVB RO,@@WRINBY :SAVE THE DRIVE #
ROL RO :MAKE IT AN INDEX
MOV DRVD(RO),@@PATRN :PICK UP THE DATA PATTERN
JSR PC,MASK :CALL THE MASK SUBROUTINE
JSR PC,CYLADR :GO FORM A CYLINDER ADDRESS
BR RD4 :RETURN HERE IF NOT LAST BASE ADDR.
BR RD1 :IF LAST BASE ADDRESS, RETURN HERE
RD4: BIS @@DSKTMP,@@DSKADR :SET THE DRIVE # BITS IN DISK ADDR.
RD5: BISB (R5)+,@@DSKADR :SET THE SECTOR BITS IN DISK ADDR.
RD6: MOV @RDBUFF,@@BUSADR :GET THE BUFFER ADDR.
NOP :***
MOV @@SECCNT,@@WRDCNT :GET THE WORD COUNT
MOV @READCS,@@CONTRL :GET THE READ CONTROL WORD
JSR PC,EXECUT :DO THE READ
JSR PC,RDCHK :CHECK THE DATA
BIC @17,@@DSKADR :CLR THE SECTOR BITS IN DISK ADDR.
CMP @DRCNT1,R5 :WAS THIS THE LAST SECTOR?

```

```

930 006000 001352          BNE      RD5      ;IF NO GO BACK
931 006002 012705 001305    MOV      @SECTBL,RS ;IF YES RESET SECTOR POINTER
932 006006 032737 000020 001334  BIT      @BIT4,@DSKADR ;WAS IT SURFACE "1" THAT WAS READ?
933 006014 001006          BNE      RD7      ;IF YES BRANCH
934 006016 052737 000020 001334  BIS      @BIT4,@DSKADR ;NO, SET SURFACE "1" BIT
935 006024 105137 001362    COMB    @PATTN     ;MAKE HEAD "1" PATTERN
936 006030 000736          BR       RD5      ;GO BACK AND EXECUTE
937 006032 042737 000020 001334  RD7:    BIC      @BIT4,@DSKADR ;CLEAR THE SURFACE BIT
938 006040 105137 001362    COMB    @PATTN     ;MAKE IT SURFACE 0 DATA
939 006044 005202          INC      R2        ;INCREMENT THE SELECTED CYL TABLE POINTER
940 006046 004737 005272    JSR     PC,CYLOFF  ;GO FORM NEXT ADDRESS
941 006052 000722          BR       RD4      ;IF HERE IT IS NOT THE LAST BASE ADDR
942 006054 000670          BR       RD1      ;IF HERE GET NEXT WORD-DRIVE FINISHED

943
944 006056 012600          EXIT2:  MOV     @SP)+,RD ;RESTORE RD
945 006060 000207          RTS      PC        ;RETURN TO MAIN LINE CODE

946
947 ;THE ERROR CHECK ROUTINE CHECKS IF AN ERROR OCCURRED
948 ;ON WRITING OR READING.
949
950 006062 032777 140000 173300  ERRCHK: BIT      @140000,@RKCS ;HARD ERROR OR ERROR SET?
951 006070 000240          NOP                    ;HALT HERE TO EXAMINE ERROR REG..ECT.
952 006072 001420          BEQ     TSTSN1       ;IF NO, GO TEST 'SIN' BIT
953 006074 012777 000001 173266  MOV      @1,@RKCS    ;IF YES, ISSUE CNTROL RESET + GO
954 006102 004737 005574          JSR     PC,SMTME
955 006106 012777 177777 173244  MOV      @-1,@ERRFLG ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)
956 006114 105777 173250  :S:      TSTB    @RKCS   ;CNTROL READY BIT SET (FROM CNTROL RESET)
957 006120 100375          BPL     IS           ;IF NO WAIT. (IF HUNG HERE RUN STATIC)
958 006122 105337 001321    DECB    @DECNT       ;DECREMENT ERROR COUNTER
959 006126 001002          BNE     TSTSN1       ;HAVE ERROR BITS SET 5 TIMES?
960 006130 000137 006212    JMP     @RESTRT      ;IF HERE 5 ERRORS HAVE OCCURRED
961 006134 032777 001000 173222  TSTSN1: BIT      @1000,@RKDS ;SEEK INCOMPLETE SET?
962 006142 000240          NOP                    ;***
963 006144 001530          BEQ     RETRN2       ;BRANCH IF NO
964 006146 012777 000015 173214  MOV      @15,@RKCS   ;IF YES, ISSUE DRIVE RESET, GO
965 006154 012777 177777 173176  MOV      @-1,@ERRFLG ;FLAG AN ERROR (PREVENT UPDATE OF ADDR)
966 006162 004737 005574          JSR     PC,SMTME
967 006166 105777 173176  2S:      TSTB    @RKCS   ;
968 006172 100375          BPL     2S          ;
969 006174 032777 000100 173162  BIT      @100,@RKDS  ;"R/W/S READY" BIT SET?
970 006202 001771          BEQ     2S          ;IF NO WAIT! (IF HUNG HERE RUN STATIC)
971 006204 105337 001322    DECB    @CNTSIN      ;DECREMENT SEEK INCOMPLETE COUNTER
972 006210 001106          BNE     RETRN2       ;IF 3 'SIN' ERRORS FALL THROUGH
973 006212 105737 001321  RESTRT: TSTB    @DECNT ;
974 006216 000240          NOP                    ;***
975 006220 001421          BEQ     IS         ;
976 006222 104401 006230  TYPE    @65S        ;:TYPE ASCIZ STRING
977 006226 000415          BR      @64S        ;:GET OVER THE ASCIZ
978 ;:65S: .ASCIZ <15><12><3> 'SIN' ERRORS OCCURRED/
979 ;:64S:
980 006262 000415          BR      2S         ;
981 006264 ;:6S:
982 006264 104401 006272  TYPE    @67S        ;:TYPE ASCIZ STRING
983 006270 000412          BR      @66S        ;:GET OVER THE ASCIZ
984 ;:67S: .ASCIZ <15><12><5> ERRORS OCCURRED.
985 ;:66S:

```

```

986 006316
987 006316 104401 006324
988 006322 000422
989
990 006370
991 006370 105737 001316
992 006374 100006
993 006376 062706 000004
994 006402 012600
995 006404 052710 100000
996 006410 000207
997 006412 052710 100000
998 006416 012706 001100
999 006422 000137 003304
1000 006426 000207
1001
1002
1003
1004
1005 006430 010446
1006 006432 010546
1007 006434 105037 001320
1008 006440 000240
1009 006442 012737 000005 001350
1010 006450 012704 007336
1011 006454 013705 001362
1012 006460 020524
1013 006462 001515
1014 006464 105737 001320
1015 006470 001046
1016 006472 104401 006500
1017 006476 000420
1018
1019 006540
1020 006540 152737 000377 001320
1021 006546 113746 001317
1022 006552 104403
1023 006554 001
1024 006555 000
1025 006556 104401 006564
1026 006562 000411
1027
1028 006606
1029 006606
1030 006606 104401 006614
1031 006612 000405
1032
1033 006626
1034 006626 013746 001334
1035 006632 104403
1036 006634 006
1037 006635 001
1038 006636 104401 006644
1039 006642 000405
1040
1041 006656

```

```

25:
TYPE 695 ::TYPE ASCIZ STRING
BR 685 ::GET OVER THE ASCIZ
::695: .ASCIZ / DRIVE DECLARED DOWN!! NOT TESTED !/
685:
TSTB 2#COMND :TEST THE COMMAND
BPL 35 :IF WRITE, BRANCH
ADD 84,SP :POINT TO SAVE RD
MOV (SP)+,RO :GET RD BACK
BIS #BIT15,(RO) :SET THE DOWN BIT
RTS PC :RETURN TO MAIN CODE
35: BIS #BIT15,(RO) :SET THE DOWN BIT
MOV #STACK,SP :RESTORE THE STACK
JMP 2#GO1
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 RS,-(SP) :SAVE RS FOR RDLINK
CLRB 2#HDRFLG :CLEAR THE PRINT HEADER FLAG
NOP :***
MOV #5,2#CHKCNT :PUT ERROR COUNT IN CHECK COUNT
MOV #RDBUFF,R4 :GET THE TABLE ADDRESS TO R4
MOV 2#PATRN,RS :GET THE EXPECTED DATA TO RS
15: CMP RS,(R4)+ :ARE THEY THE SAME
BEQ 35 :IF YES BRANCH
TSTB 2#HDRFLG :IS THE HEADER FLAG CLEAR
BNE 25 :IF NO BRANCH
TYPE 655 ::TYPE ASCIZ STRING
BR 645 ::GET OVER THE ASCIZ
::655: .ASCIZ <15><12>/ERROR! DATA WRITTEN BY DRIVE
645:
BISB #377,2#HDRFLG :SET THE HEADER FLAG
MOV 2#WRNBY,-(SP) :PICK UP THE DRIVE # THAT WROTE
TYPOS
.BYTE 1
.BYTE 0
TYPE 675 ::TYPE ASCIZ STRING
BR 665 ::GET OVER THE ASCIZ
::675: .ASCIZ / CANNOT BE READ./
665:
25:
TYPE 695 ::TYPE ASCIZ STRING
BR 685 ::GET OVER THE ASCIZ
::695: .ASCIZ <15><12>/ ADDR=/
685:
MOV 2#DSKADR,-(SP) :PICK UP THE ADDRESS THAT FAILED
TYPOS
.BYTE 6
.BYTE 1
TYPE 715 ::TYPE ASCIZ STRING
BR 705 ::GET OVER THE ASCIZ
::715: .ASCIZ / EXPCTD=/
705:

```

```

1042 006656 010546          MOV      RS,-(SP)          :PICK UP THE EXPECTED DATA (GOOD)
1043 006660 104404          TYPON
1044 006662 104401 006670      TYPE      73$           ;;TYPE ASCIZ STRING
1045 006666 000405          BR        72$           ;;GET OVER THE ASCIZ
1046 006702          ;;73$: .ASCIZ / RECD=
1047 006702          72$:
1048 006702 016446 177776      MOV      -2(R4),-(SP)    :PICK UP THE RECEIVED DATA (BAD)
1049 006706 104404          TYPON
1050 006710 005337 001350      DEC      #CHKCNT        :DECREMENT THE CHECK COUNT
1051 006714 001403          BEQ      4$             :IF ZERO, BRANCH
1052 006716 022704 010336      3$:      CMP      #MANSEL,R4   :DONE ALL CHECKS?
1053 006722 001256          BNE      1$             :IF NO, GO BACK
1054 006724 012605      4$:      MOV      (SP)+,R5       :RESTORE R5
1055 006726 012604          MOV      (SP)+,R4       :RESTORE R4
1056 006730 000207          RTS      PC              :RETURN TO CALLER

:THIS ROUTINE BUILDS A TABLE OF PARAMETERS TO PASS TO THE SECONO SYSTEM
:IT PACKS THE INFO FOR TWO DRIVES INTO ONE WORD

1061 006732 005003      SECONO: CLR      R3          :CLEAR THE WORD COUNTER
1062 006734 000240          NOP
1063 006736 012702 001256      MOV      #MSKTBL,R2     :***
1064 006742 005042      6$:      CLR      -(R2)
1065 006744 022702 001246      CMP      #PASTBL,R2
1066 006750 001374          BNE      6$
1067 006752 012700 001166      MOV      #LOGA,R0       :GET THE ACTIVE TABLE ADDRESS
1068 006756 012001      1$:      MOI      (R0)+,R1       :PICK UP THE WORD
1069 006760 000301          SWAB     R1             :GET THE DRIVE # TO THE LOW BYTE
1070 006762 042701 177400      BIC      #177400,R1     :CLEAR THE UNWANTED BITS
1071 006766 106101          ROLB     R1             :ROTATE THE BYTE. WAS DOWN SET?
1072 006770 103002          SCC      2$             :IF NO BRANCH
1073 006772 052701 000001      BIS      #BITC,R1       :SHOW THE DRIVE AS DOWN IN THE TABLE
1074 006776 105701      2$:      YSTB     R1             :IS THIS THE SYSTEM #2 DRIVE?
1075 007000 100404          BMI      3$             :BRANCH IF YES
1076 007002 142701 000360      BICB     #360,R1        :CLEAR THE UNUSED BITS
1077 007006 110122          MOVB     R1,(R2)+       :GET THIS # TO THE PASS TABLE
1078 007010 000762          BR       1$             :GET THE NEXT WORD FROM ACTIVE TABLE
1079 007012 110112      3$:      MOVB     R1,(R2)        :GET THE LAST DRIVE TO THE PASS TABLE
1080 007014 012702 001246      MOV      #PASTBL,R2     :RESTORE THE TABLE POINTER
1081 007020 104401 007026      TYPE     65$           :TYPE ASCIZ STRING
1082 007024 000425          BR       64$           :GET OVER THE ASCIZ
1083          ;;65$: .ASCIZ <15><12>/LOAD AND START ADDRESS 210 ON SYSTEM #2
1084          64$:
1085 007100          TYPE     67$           :TYPE ASCIZ STRING
1086 007104 104401 007106      BR       66$           :GET OVER THE ASCIZ
1087          ;;67$: .ASCIZ <15><12>/AND TYPE THE BELOW WHEN ASKED FOR IT./
1088          66$:
1089 007156 005203      4$:      INC      R3             :INCREMENT THE WORD COUNTER
1090 007160 104401 007166      TYPE     69$           :TYPE ASCIZ STRING
1091 007164 000405          BR       68$           :GET OVER THE ASCIZ
1092          ;;69$: .ASCIZ <15><12>/WORD /
1093          68$:
1094 007200          MOV      R3,-(SP)      :GET THE WORD COUNT ON THE STACK
1095 007202 104403          TYPOS
1096 007204 006
1097 007205 000

```

```

1098 007206 104401 007214          TYPE      715          ::TYPE ASCIZ STRING
1099 007212 000401                BR      705          ::GET OVER THE ASCIZ
1100                ::715: .ASCIZ  /=/
1101                705:
1102 007216                MOV      (R2)+,-(SP)    ;GET THE FIRST TO THE STACK
1103 007216 012246                TYPOS
1104 007220 104403                .BYTE   6
1105 007222      006                .BYTE   1
1106 007223      001                BIT     #BIT15,-2(R2)  ;WAS THIS THE TABLE TERMINATOR
1107 007224 032762 100000 177776      BNE     55            ;BRANCH IF YES
1108 007232 001004                BIT     #BIT7,-2(R2)  ;TERMINATOR?
1109 007234 032762 000200 177776      BEQ     45            ;IF NO BRANCH
1110 007242 001745                TST     -(R0)
1111 007244 005740                SS:
1112 007246 000000                HALT

1113                RETFR2:
1114 007250                TYPE      655          ::TYPE ASCIZ STRING
1115 007250 104401 007256                BR      645          ::GET OVER THE ASCIZ
1116 007254 000404                ::655: .ASCIZ  <15><12>/WORD=/
1117                645:
1118 007266                RDOCT
1119 007266 104411                MOV      (SP)+,R2      ;GET THE WORD FROM SYSTEM 2 TO TABLE
1120 007270 012602                BIC     #177400,R2
1121 007272 042702 177400                MOV      #LOGA,R4      ;SET POINTER LOOK FOR FIRST "UP" DRIVE
1122 007276 012704 001166                TST     (R4)+          ;DRIVE UP?
1123                IS:
1124 007302 005724                BMI     15            ;IF NO BRANCH
1125 007304 100776                MOV      R4,2*SPASS
1126 007306 010437 001100                MOV      -(R4),R1
1127 007312 014401                NOP
1128 007314 000240                ***
1129 007316 004737 004204                JSR     PC,LDFLG       ;CALL D02+
1130 007322 004737 005612                JSR     PC,RDLINK      ;CALL READ CHECK
1131 007326 013700 001100                MOV      2*SPASS,R0
1132 007332 000137 003352                JMP     2*EXTFR2      ;GO TO END OF TEST

1133                ROBUFF: .BLKW  400
1134                MANSEL: NOP          ;TABLE TERMINATOR

1135
1136
1137
1138
1139
1140 010340                BADONE:
1141 010340 104401 010346                TYPE      655          ::TYPE ASCIZ STRING
1142 010344 000401                BR      645          ::GET OVER THE ASCIZ
1143                ::655: .ASCIZ  /?/
1144                645:
1145 010350                .SBTTL  OSCILLATING SEEK ROUTINE

1146
1147
1148
1149 010350                SECT.2:
1150 010350 104401 010356                TYPE      655          ::TYPE ASCIZ STRING
1151 010354 000416                BR      645          ::GET OVER THE ASCIZ
1152                ::655: .ASCIZ  <15><12>/OSCILLATING SEEK PACKAGE/
1153                645:
1154 010412

```

```

1154
1155 010412 012700 020470      MOV      #DRIVD,R0      ;FIND OUT WHICH DRIVES ARE
1156 010416 005001          CLR      R1             ;PRESENT AND PUT THE
1157 010420 010107 170752      1S:     MOV      R1,DRKDA ;DRIVE #'S IN A TABLE STARTING
1158 010424 105777 170734      TSTB    DRKCS          ;AT 'DRIVD'. BITS 15-13 CONTAINS
1159 010430 100001          BPL      2S            ;THE DRIVE #
1160 010432 010120          MOV      R1,(R0)+
1161 010434 062701 020000      2S:     ADD      #20000,R1
1162 010440 001367          BNE     1S
1163 010442 012710 177777      MOV      #-1,(R0)      ;SET THE TERMINATOR TO THE TABLE
1164
1165 010446 013702 001376      INIT.2: MOV      RKDA,R2
1166 010452 012777 000001 170710      MOV      #1,DRKCS      ;ISSUE CONTROL RESET + GO !
1167 010460 004737 005574      JSR      PC,SMTME
1168 010464 105777 170700      1S:     TSTB    DRKCS      ;DID CONTROL READY SET?
1169 010470 100375          BPL      1S            ;IF NO WAIT! (IF HUNG RUN STATIC)
1170 010472 012700 020470      MOV      #DRIVD,R0
1171 010476 012077 170674      3S:     MOV      (R0)+,DRKDA
1172 010502 012777 000015 170660      MOV      #15,DRKCS     ;ISSUE DRIVE RESET + GO!
1173 010510 004737 005574      JSR      PC,SMTME
1174 010514 105777 170650      2S:     TSTB    DRKCS
1175 010520 100375          BPL      2S
1176 010522 022710 177777      CMP      #-1,(R0)
1177 010526 001363          BNE     3S
1178 010530 104401 010536      TYPE    ,65S           ;;TYPE ASCIZ STRING
1179 010534 000432          BR      64S           ;;GET OVER THE ASCIZ
1180      ;;65S: .ASCIZ <15><12>/SET SWD TO SW7 TO SELECT DRIVES FOR TEST AND CONT/
1181 010622          64S:
1182 010622 104401 010630      TYPE    ,67S           ;;TYPE ASCIZ STRING
1183 010626 000426          BR      66S           ;;GET OVER THE ASCIZ
1184      ;;67S: .ASCIZ <15><12>/RESET SWD TO SW7 TO TEST ALL AVAIL DRIVES/
1185 010704          66S:
1186 010704 022737 000176 001140 7S:     CMP      #SWREG,SWR    ;SOFTWARE SWITCH REGISTER IN USE ?
1187 010712 001003          BNE     8S            ;BR IF NOT
1188 010714 104405          GTSWR          ;REQUEST NEW CONTENTS FOR SWITCH REG
1189 010716 000400          BR      9S            ;CONTINUE
1190 010720 000000      9S:     HALT          ;WAIT FOR OPERATOR TO ENTER NEW SWR VALLE
1191 010722 117704 170212 8S:     MOV      2SWR,R4      ;SWD TO SW7 TO R4
1192 010726 001413          BEQ     4S            ;NONE SET, SO TEST ALL
1193 010730 012700 020470      MOV      #DRIVD,R0    ;TABLE TO STORE DRIVE ADDRS
1194 010734 005001          CLR      R1            ;ADDR OF DRIVE
1195 010736 006004      6S:     ROR      R4            ;NEXT SWITCH TO CARRY
1196 010740 103001          BCC     5S            ;SWITCH NOT SET
1197 010742 010120          MOV      R1,(R0)+     ;SWITCH SET, SO MOVE ADDR TO TABLE
1198 010744 062701 020000      5S:     ADD      #20000,R1
1199 010750 001372          BNE     6S            ;ADDR OF NEXT DRIVE
1200 010752 012720 177777      MOV      #-1,(R0)+    ;ALL DONE WHEN ZERO
1201 010756 105737 001326      4S:     TSTB    OSPFLG       ;TABLE TERMINATOR
1202 010762 001165          BNE     RWSRDY        ;TYPED ONCE?
1203 010764 104401 010772      TYPE    ,69S           ;IF YES, DONT RETYPE
1204 010770 000432          BR      68S           ;;TYPE ASCIZ STRING
1205      ;;69S: .ASCIZ <15><12>/TOGGLE THE "FIRST CYLINDER ADDRESS" (OUTER LIMIT)/
1206 011056          68S:
1207 011056 104401 011064      TYPE    ,71S           ;;TYPE ASCIZ STRING
1208 011062 000441          BR      70S           ;;GET OVER THE ASCIZ
1209      ;;71S: .ASCIZ <15><12>/INTO THE LOW BYTE (BIT0-7) OF THE SWITCH REGISTER AND THE "LAST

```

```

1210 011166 70$: TYPE 73$ ;:TYPE ASCIZ STRING
1211 011166 104401 011174 BR 72$ ;:GET OVER THE ASCIZ
1212 011172 000430 ;:73$: .ASCIZ <15><12>/CYLINDER ADDRESS" (INNER LIMIT) INTO THE HIGH
1213 ;:72$:
1214 011254 ;:75$: TYPE 75$ ;:TYPE ASCIZ STRING
1215 011254 104401 011262 BR 74$ ;:GET OVER THE ASCIZ
1216 011260 000424 ;:75$: .ASCIZ <15><12>/ ;:BYTE (BIT8-15), THEN PRESS CONTINUE./
1217 ;:74$:
1218 011332 18: INCB OSPFLG
1219 011332 105237 001326 RWSRDY: BIT #100,DRKDS ;:"READ/WRITE/SEEK READY" BIT SET?
1220 011336 032777 000100 170020 BEQ RWSRDY ;:IF NO WAIT! (IF HUNG RUN STATIC)
1221 011344 001774 TRYAGN: CMP #SWREG,SWR ;:SOFTWARE SWITCH REGISTER IN USE ?
1222 011346 022737 000176 001140 BNE 15 ;:BR IF NOT
1223 011354 001002 GTSWR ;:GET NEW CONTENTS
1224 011356 104405 BR CONTIN ;:CONTINUE
1225 011360 000401 13: HALT
1226 011362 000000 CONTIN: MOV #1,DRKCS ;:CONTROL RESET
1227 011364 012777 000001 167776 TSTB DRKCS
1228 011372 105777 167772 BPL -4
1229 011376 100375 MOV SWR,R5 ;:GET THE ADDRESS OF THE SWITCH REG. TO R5
1230 011400 013705 001140 15: MOVB (R5)+,R1 ;:GET A BYTE TO R1
1231 011404 112501 BIC #177400,R1 ;:CLEAR THE UNUSED BITS
1232 011406 042701 177400 CMP #312,R1 ;:IS ADDRESS LEGAL?
1233 011412 022701 000312 BPL 25 ;:BRANCH IF YES
1234 011416 100034 TYPE 65$ ;:TYPE ASCIZ STRING
1235 011420 104401 011426 BR 64$ ;:GET OVER THE ASCIZ
1236 011424 000430 ;:65$: .ASCIZ <15><12>/INVALID ADDRESS IN SWITCH REGISTER TRY AGAIN/
1237 ;:64$:
1238 011506 BR TRYAGN ;:GO BACK FOR NEW PARAMETERS
1239 011506 000717 2$: ROL R1 ;:ROTATING THIS REGISTER
1240 011510 006101 ROL R1 ;:MAKES THE BYTE FROM THE
1241 011512 006101 ROL R1 ;:SWITCH REGISTER LINE UP
1242 011514 006101 ROL R1 ;:WITH THE CYLINDER BITS OF
1243 011516 006101 ROL R1 ;:THE RKDA REGISTER.
1244 011520 006101 BIC #160037,R1 ;:CLEAR THE UNUSED BITS
1245 011522 042701 160037 BIT #1,R5 ;:IS THIS THE LOW BYTE?
1246 011526 032705 000001 BNE 35 ;:BRANCH IF YES
1247 011532 001003 MOV R1,#SEEKI ;:STORE THE INNER LIMIT OF THE SEEK
1248 011534 010137 001402 BR SEKSET ;:START SEEKS
1249 011540 000403 3$: MOV R1,#SEEKO ;:STORE THE OUTER LIMIT OF THE SEEK
1250 011542 010137 001404 BR 15 ;:GO BACK AND GET HIGH BYTE.
1251 011546 000716 SEKSET: MOV #50,R3 ;:GET THE NUMBER OF SEEK CYCLES TO R3
1252 011550 012703 000050 MOV #SEEKO,R4 ;:ADD THE OUTER LIMIT CYLINDER ADDRESS
1253 011554 013704 001404 MOV #SEEKI,R5 ;:ADD THE INNER LIMIT CYLINDER ADDRESS
1254 011560 013705 001402 LDSEEK: MOV #DRIVO,R0 ;:INITIALIZE POINTER
1255 011564 012700 020470 5$: MOV R4,DRKDA ;:GET INNER LIMIT CYL ADRES
1256 011570 010477 167602 BIS (R0)+,DRKDA ;:SET DRIVE ADRES
1257 011574 052077 167576 MOV #11,DRKCS ;:ISSUE SEEK+GO! (FOR INNER LIMIT)
1258 011600 012777 000011 167562 JSR PC,SMTME
1259 011606 004737 005574 3$: TSTB DRKCS
1260 011612 105777 167552 BPL 35
1261 011616 100375
1262 011620 022710 177777 CMP #-1,(R0) ;:ALL DRIVES DONE"

```

```

1266 011624 001361 BNE 55 ;NO
1267 011626 012700 020470 MOV #DRIVO,RO
1268 011632 012077 167540 65: MOV (RO)+,DRKDA ;ADRES THE DRIVE
1269 011636 032777 000100 167520 75: BIT #RWS,DRKDS ;SEEK DONE?
1270 011644 001774 BEQ 75 ;NO, WAIT
1271 011646 022710 177777 CMP #-1,(RO) ;ALL DRIVES DONE?
1272 011652 001367 BNE 65 ;NO
1273
1274 011654 012700 020470 MOV #DRIVO,RO
1275 011660 010577 167512 85: MOV R5,DRKDA ;SET OUTER CYL ADRES
1276 011664 052077 167506 BIS (RO)+,DRKDA ;SET DRIVE ADRES
1277 011670 012777 000011 167472 MOV #11,DRKCS ;ISSUE SEEK+GO! (FOR OUTER LIMIT)
1278 011676 004737 005574 JCR PC,SMTME
1279 011702 105777 167462 95: TSTB DRKCS
1280 011706 100375 BPL 95
1281
1282 011710 022710 177777 CMP #-1,(RO) ;ALL DONE?
1283 011714 001361 BNE 85 ;NO
1284
1285 011716 012700 020470 MOV #DRIVO,RO
1286 011722 012077 167450 105: MOV (RO)+,DRKDA ;SET DRIVE ADRES
1287 011726 032777 000100 167430 115: BIT #RWS,DRKDS ;SEEK DONE?
1288 011734 001774 BEQ 115
1289 011736 022710 177777 CMP #-1,(RO) ;ALL DONE?
1290 011742 001367 BNE 105
1291 011744 005303 DEC R3 ;DONE 50 SEEK CYCLES (100 SEEKS)
1292 011746 001306 BNE LDSEEK ;IF NO BRANCH (KEEP CYCLING)
1293 011750 000605 BR CONTIN ;CHECK SWR FOR CHANGE AND CONTINUE
1294
1295
1296
1297
1298
1299 .SBTTL FORMATTER-SURFACE VERIFIER
1300
1301 011752 BAD.IN: TYPE 655 ;;TYPE ASCIZ STRING
1302 011752 104401 011760 BR 645 ;;GET OVER THE ASCIZ
1303 011756 000401 ;;655: .ASCIZ /?/
1304 645:
1305 011762 SECT.1: TYPE 655 ;;TYPE ASCIZ STRING
1306 011762 BR 645 ;;GET OVER THE ASCIZ
1307 011762 104401 011770 ;;655: .ASCIZ <15><12>/FORMATTER-SURFACE VERIFIER,SET SW REG FOR DRV #'S. PRESS CONT./
1308 011766 000441 645:
1309
1310 012072 022737 000176 001140 CMP #SWREG,SWR ;SOFTWARE SWITCH REGISTER IN USE ?
1311 012100 001002 BNE 15 ;BR IF NOT
1312 012102 104405 GTSWR ;GET SWITCH REGISTER VALUE
1313 012104 000401 BR 25 ;CONTINUE
1314 012106 000000 15: HALT ;WAIT FOR 'CONTINUE'
1315 012110 012737 000001 020320 25: MOV #1,SFCNT ;SET SHIFT COUNT
1316 012116 005037 020322 CLR DRVCNT ;CLEAR DRIVE COUNT
1317 012122 033777 020320 167010 513: BIT SFCNT,#SWR ;IS THIS SW SET?
1318 012130 001011 BNE S10 ;YES FORMAT THIS DRIVE
1319 012132 006337 020320 511: ASL SFCNT
1320 012136 005237 020322 INC DRVCNT

```

```

1322 012142 022737 000010 020322      CMP      #10,DRVCNT      ;ALL DONE?
1323 012150 001556                      BEQ      GD1
1324 012152 000763                      BR       S13
1325 012154 013700 020322      S10:     MOV      DRVCNT,RO
1326 012160 104401 001161      TYPE    ',SCRLF'
1327 012164 104401 020324      TYPE    ',EMI'      ;TYPE 'DRIVE'
1328 012170 010046                      MOV      RO,-(SP)    ;TYPE DRIVE #
1329 012172 104402                      TYPOC
1330 012174 000300                      SWAB
1331 012176 006100                      RCL     RO
1332 012200 006100                      ROL     RO
1333 012202 006100                      ROL     RO
1334 012204 006100                      ROL     RO
1335 012206 006100                      RCL     RO
1336 012210 010037 001352      MOV      RO,@#DSKTMP
1337 012214 012737 000000 001422      MOV      #0,ERRWF
1338 012222 012737 000000 001424      MOV      #0,ERRRF      ;CLEAR OUT ERROR COUNTS.
1339 012230 012737 000000 001426      MOV      #0,ERRRFC
1340 012236 012737 000000 001430      MOV      #0,ERRWCH
1341 012244 012737 000000 001432      MOV      #0,ERRWCS
1342 012252 012737 177750 001416      S12:     MOV      #-24,RWC      ;SET WORD COUNT FOR READ FORMAT.
1343 012260 012737 164000 001412      MOV      #-6144,WC     ;SET UP WORD COUNT FOR WRITES.
1344 012266 012737 000000 001420      MOV      #0,EXTR      ;CLEAR EXTRA BIT FOR 12 SECTOR PACK.
1345 012274 012701 177772      COMMON: MOV      #-6,R1      ;SET UP LOOP COUNT FOR THE CLEANER.
1346 012300 012777 014500 167070      COM:     MOV      #14500,ARKDA ;SET UP FOR A SEEK TO 202.
1347 012306 053777 001352 167062      BIS     @#DSKTMP,ARKDA
1348 012314 105777 167050      S1:      TSTB   ARKCS      ;IS THE CONTROLLER READY?
1349 012320 100375                      BPL     S1           ;NO SO WAIT.
1350 012322 012777 000011 167040      MOV      #11,ARKCS    ;DO THE SEEK.
1351 012330 032777 000100 167026      S2:      BIT     #BIT6,ARKDS ;IS THE SEEK DONE?
1352 012336 001774                      BEQ     S2           ;NO SO WAIT.
1353 012340 105777 167024      S3:      TSTB   ARKCS      ;IS CONTROLLER READY?
1354 012344 100375                      BPL     S3           ;NO
1355 012346 012777 000015 167014      MOV      #15,ARKCS    ;DO A DRIVE RESET.
1356 012354 032777 000100 167002      S4:      BIT     #BIT6,ARKDS ;IS DRIVE RESET DONE.
1357 012362 001774                      BEQ     S4           ;NO
1358 012364 005201                      INC     R1           ;COUNT THE CLEANER LOOP.
1359 012366 001344                      BNE     COM         ;MORE TO GO.
1360 012370 012737 000000 001410      MOV      #0,DA        ;START OUT AT CYL. 0.
1361 012376 012777 177777 167002      NEXT:   MOV      #177777,@BA ;PUT ALL ONE'S IN BUFFER.
1362 012404 004137 012512                      JSR     R1,IO        ;GO DO THE DISK THING.
1363 012410 012777 000000 166770      MOV      #0,@BA       ;PUT ALL ZERO'S IN BUFFER.
1364 012416 004137 012512                      JSR     R1,IO        ;GO DO IT AGAIN.
1365 012422 012777 125252 166756      MOV      #125252,@BA  ;PUT A ALT. PATTERN IN BUFFER.
1366 012430 004137 012512                      JSR     R1,IO        ;ONCE MORE.
1367 012434 005177 166746                      COM     @BA          ;COMPLEMENT THE LAST PATTERN.
1368 012440 004137 012512                      JSR     R1,IO        ;AND AGAIN.
1369 012444 062737 000040 001410      ADD     #40,DA        ;INCREMENT TO THE NEXT CYL.
1370 012452 022737 014540 001410      CMP     #14540,DA     ;ARE WE DONE WITH THIS ONE?
1371 012460 001346                      BNE     NEXT        ;NO SO DO THE NEXT CYL.
1372 012462
1373 012462 104401 012470      GOOT:   TYPE    ',655'      ;TYPE ASCIZ STRING
1374 012466 000406                      BR      645          ;GET OVER THE ASCIZ
1375
1376 012504      S5:     .ASCIZ '<CR><LF>/PACK GOOD/'
1377 012504 000612      S6:     BR      S11

```

```

1378 012506 000137 001440 GD1: JMP @START ;RESTART
1379
1380
1381
1382
1383
1384
1385 012512 013777 001412 166552
1386 012520 013777 001410 166550
1387 012528 053777 001352 166542
1388 012534 013777 001406 166532
1389 012542 012777 000000 166520
1390 012550 052777 006000 166512
1391 012556 052777 000002 166504
1392 012564 053777 001420 166576
1393 012572 052777 000001 166570
1394 012600 105777 166564
1395 012604 100375
1396 012606 005777 166556
1397 012612 100541
1398 012614 012737 000000 001422
1399
1400
1401
1402 012622 013777 001416 166542
1403 012630 013777 001410 166540
1404 012636 053777 001352 166532
1405 012644 013777 001414 166522
1406 012652 012777 000000 166510
1407 012660 052777 002000 166502
1408 012666 052777 000004 166474
1409 012674 053777 001420 166466
1410 012702 052777 000001 166460
1411 012710 105777 166454
1412 012714 100375
1413 012716 032777 040000 166444
1414 012724 001134
1415 012726 012737 000000 001424
1416
1417
1418
1419 012734 013777 001412 166430
1420 012742 013777 001410 166426
1421 012750 053777 001352 166420
1422 012756 013777 001406 166410
1423 012764 012777 000000 166376
1424 012772 052777 004400 166370
1425 013000 053777 001420 166362
1426 013006 052777 000006 166354
1427 013014 052777 000001 166346
1428
1429
1430
1431 013022 013703 001416
1432 013026 005403
1433 013030 063703 001414
    
```

DISK I/O SUBROUTINE.

SET UP FOR A WRITE/FORMAT.

```

10: MOV WC, @RKWC ;SET UP THE WORD COUNT REG.
    MOV DA, @RKDA ;SET UP THE DISK ADDRESS.
    BIS @DSKTMP, @RKDA ;SET THE UNIT NUMBER UP.
    MOV BA, @RKBA ;SET UP THE BUSS ADDRESS.
    MOV #0, @RKCS ;CLEAR THE CONTROL REG. FOR SET UP.
    BIS #BIT10+BIT11, @RKCS ;SET FORMAT&INHIBIT INC. BITS.
    BIS #BIT1, @RKCS ;SET UP WRITE FUN.
    BIS EXTR, @RKCS ;SET UP 12OR16 SECTOR PACK.
    BIS #BIT0, @RKCS ;GO DO THE WRITE FORMAT.
15: TSTB @RKCS ;IS WRITE FORMAT DONE?
    BPL 15 ;NO SO WAIT.
    TST @RKCS ;WAS THERE A ERROR?
    BMI WFERR ;YES GO SERVICE IT.
    MOV #0, ERRWF ;CLEAR OUT THE ERROR COUNTER.
    
```

SET UP FOR A READ/FORMAT

```

25: MOV RWC, @RKWC ;SET UP WORD COUNT REG.
    MOV DA, @RKDA ;SET UP DISK ADDRESS.
    BIS @DSKTMP, @RKDA ;SET THE UNIT NUMBER.
    MOV RBA, @RKBA ;SET UP THE BUSS ADDRESS.
    MOV #0, @RKCS ;CLEAR THE CONTROL REG.
    BIS #BIT10, @RKCS ;SET THE FORMAT BIT.
    BIS #BIT2, @RKCS ;SET UP READ FUN.
    BIS EXTR, @RKCS ;SET UP 12 OR 16 SECTOR PACK.
    BIS #BIT0, @RKCS ;GO DO THE READ FORMAT.
25: TSTB @RKCS ;IS THE READ FORMAT DONE?
    BPL 25 ;NO SO WAIT.
    BIT #BIT14, @RKCS ;WAS TRERE A ERROR?
    BNE RFERR ;YES GO SERVICE IT.
    MOV #0, ERRRF ;CLEAR OUT THE ERROR COUNT.
    
```

SET UP FOR A WRITE CHECK.

```

    MOV WC, @RKWC ;SET UP WORD COUNT REG.
    MOV DA, @RKDA ;SET UP DISK ADDRESS.
    BIS @DSKTMP, @RKDA ;SET UP THE UNIT NUMBER
    MOV BA, @RKBA ;SET UP BUSS ADDRESS.
    MOV #0, @RKCS ;CLEAR THE CONTROL REG.
    BIS #BIT11+BIT8, @RKCS ;SET INHIBIT INCR.&STOP ON SOFT ERROR BITS.
    BIS EXTR, @RKCS ;SET 12 OR 16 SECTOR PACK.
    BIS #BIT1+BIT2, @RKCS ;SET UP WRITE CHECK FUN.
    BIS #BIT0, @RKCS ;GO DO THE WRITE CHECK.
    
```

CHECK HEADERS READ BY THE READ/FORMAT.

```

    MOV RWC, R3 ;PUT NUMBER OF WORDS TO
    NEG R3 ;CHECK IN REG 3.
    ADD RBA, R3 ;SET REG 3 TO THE LAST WORD TO BE CHECKED.
    
```

```

143 013034 013702 001414      MOV      RBA,R2      ;SET REG 2 TO STARTING ADD. OF BUFF.
143 013040 023723 001410      MORE:    CMP      DA(12)+ ;CHECK THAT HEADER IS RIGHT
143 013044 031073          BNE      RFCERR      ;THIS HEADER WAS WRONG GO SERVICE IT.
143 013046 020302          CMP      R3,R2      ;ARE WE DONE?
143 013050 001373          BNE      MORE        ;NO GO CHECK THE NEXT ONE.
143 013052 012737 000000 001426      MOV      #0,ERRRFC   ;CLEAR OUT THE ERROR COUNT.

;LETS CHECK ON THE WRITE CHECK WE STARTED.
143 013060 105777 166304      IS:      TSTB     @RKCS
143 013064 100375          BPL      IS          ;THE CONTROLER IS STILL BUSY.
143 013066 035777 166276          TST     @RKCS      ;WAS THERE A ERROR?
143 013072 100407          BMI     WCERRR     ;YES GO SERVICE IT.
143 013074 012737 000000 001430      MOV      #0,ERRWCH   ;CLEAR OUT THE
143 013102 012737 000000 001432      MOV      #0,ERRWCS   ;ERROR COUNTERS.
143 013110 000201          RTS      R1         ;RETURN TO THE MAIN LINE.
143 013112 000137 013576      WCERRR:  JMP      WCERR

;ERRORS FOR WRITE FORMAT.
143 013116 005237 001422      WFERR:   INC      ERRWF      ;ADD ONE TO THE ERROR COUNT.
143 013122 022737 000004 001422      CMP      #4,ERRWF    ;HAS IT HAPPEND 4 TIMES ON THIS CYL.
143 013130 001016          BNE     RETRY       ;NO.
143 013132          SYSER:   TYPE     ,65$      ;;TYPE ASCIZ STRING
143 013132 104401 013140          BR      ,64$      ;;GET OVER THE ASCIZ
143 013136 000410          ;;65$: .ASCIZ  \15<12>/SYSTEM ERROR/
143 013160          64$:
143 013160 000000          HALT
143 013162 000137 001440          JMP      START      ;LET THE TECH. BREATH.
143 013166 012777 000000 166174      RETRY:   MOV      #0,@RKCS ;RESTART THE TEST.
143 013174 012777 000015 166166      MOV      #15,@RKCS  ;CLEAR OUT THE CONTROL REG.
143 013202 032777 000100 166154      IS:      BIT      @BIT6,@RKDS ;DO A DRIVE RESET.
143 013210 001774          BEQ     IS          ;IS IT DONE.
143 013212 000137 012512          JMP      IO         ;NO SO WAIT.
143 013212          ;TRY AGAIN.

;ERRORS FOR READ/FORMAT.
143 013216 005237 001424      RFERR:   INC      ERRRF      ;ADDONE TO ERROR COUNT.
143 013222 022737 000004 001424      CMP      #4,ERRRF    ;HAS IT HAPPEND 4 TIMES ON THIS CYL?
143 013230 001356          BNE     RETRY       ;NO DO IT AGAIN.
143 013232 000737          BR      SYSER       ;YES SO TELL HIM SO.

;READ/FORMAT ERRORS FOUND BY SOFTWARE CHECKS.
143 013234 005237 001426      RFCERR:  INC      ERRRFC     ;ADD ONE TO ERROR COUNT.
143 013240 105777 166124      IS:      TSTB     @RKCS   ;WAIT FOR THE WRITE CHECK.
143 013244 100375          BPL      IS
143 013246 022737 000004 001426      CMP      #4,ERRRFC   ;IS IT 4?
143 013254 001401          BEQ     FAILED      ;PUT OUT FAILED MESSAGE.
143 013256 000743          BR      RETRY       ;NO SO GO TRY IT AGAIN.
143 013260 042777 000037 166110      FAILED:  BIC      #37,@RKDA    ;PUT WHICH SECTORS HEADER
143 013266 042702 177740          BIC     #177740,R2
143 013272 060277 166100          ADD     R2,@RKDA    ;FAILED IN RKDA FOR THE MESSAGE.
143 013276          FAIL:   TYPE     ,65$      ;;TYPE ASCIZ STRING
143 013276 104401 013304

```

```

1490 013302 000417 BR 648 ;:GET OVER THE ASCIZ
1491 ;:655: .ASCIZ <15><12>/PACK FAILED AT (IN OCTAL) /
1492 ;:648:
1493 013342 017701 166030 MOV @RKDA,R1 ;:GENERAT THE CYL,SECTOR,SURFACE
1494 013346 010102 MOV R1,R2 ;:MESSAGE FROM RKDA
1495 013350 042702 177770 BIC #177770,R2
1496 013354 062702 000260 ADD #260,R2
1497 013358 110237 013537 MOVVB R2,SEC+1
1498 013364 004337 013564 JSR R3,SHF3
1499 013370 042702 177776 BIC #177776,R2
1500 013374 062702 000260 ADD #260,R2
1501 013400 110237 013536 MOVVB R2,SEC
1502 013404 004337 013570 JSR R3,SHF1
1503 013410 042702 177776 BIC #177776,R2
1504 013414 062702 000260 ADD #260,R2
1505 013420 110237 013550 MOVVB R2,SUR
1506 013424 004337 013570 JSR R3,SHF1
1507 013430 042702 177770 BIC #177770,R2
1508 013434 062702 000260 ADD #260,R2
1509 013440 110237 013526 MOVVB R2,CYL+2
1510 013444 004337 013564 JSR R3,SHF3
1511 013450 042702 177770 BIC #177770,R2
1512 013454 062702 000260 ADD #260,R2
1513 013460 110237 013525 MOVVB R2,CYL+1
1514 013464 004337 013564 JSR R3,SHF3
1515 013470 042702 177774 BIC #177774,R2
1516 013474 062702 000260 ADD #260,R2
1517 013500 110237 013524 MOVVB R2,CYL
1518 013504 104401 013514 TYPE 1$ ;:TYPE OUT THE GENERATED MESSAGE
1519 013510 000137 013556 JMP STOP ;:BYPASS THE INLINE MESSAGE
1520 013514 005015 054503 027114 1$: .ASCII <15><12>/CYL.
1521 013522 020040 CYL: .ASCII /000 /
1522 013524 030060 020060 .ASCII /SEC. /
1523 013530 042523 027103 020040 SEC: .ASCII /00 /
1524 013536 030060 040 .ASCII /SURF. /
1525 013541 123 051125 027106
1526 013546 020040
1527 013550 020060 005015 000 SUR: .ASCIZ /0 <<15><12>
1528 013556 .EVEN
1529 013556 000000 STOP: HALT ;:LET OPER DO HIS THING.
1530 013560 000137 001440 JMP START ;:RESTART THE TEST.
1531 ;:SHIFT SUBROUTINES.
1532
1533
1534 013564 006201 SHF3: ASR R1 ;:HERE FOR A SHIFT OF 3.
1535 013566 006201 SHF2: ASR R1 ;:HERE FOR A SHIFT OF 2.
1536 013570 006201 SHF1: ASR R1 ;:HERE FOR A SHIFT OF 1.
1537 013572 010102 MOV R1,R2 ;:PUT RESULTES IN THE WORKING REG.
1538 013574 000203 RTS R3
1539
1540 ;:ERRORS FOR WRITE CHECK.
1541
1542 013576 032777 040000 165564 WCERR: BIT #BIT14,@RKCS ;:WAS IT A HARD ERROR.
1543 013604 001010 BNE WCERR ;:YES GO PROSSES IT.
1544 013606 005237 001432 INC ERRWCS ;:ADD 1 TO THE SOFT ERROR COUNT.
1545 013612 022737 000004 001432 CMP #4,ERRWCS ;:HAS THERE BEEN 4 OF THEM?

```

```

1546 013620 001626 BEQ FAIL ;YES PUT OUT FAILED MESSAGE.
1547 013622 000137 012512 JMP IO ;NO SO TRY AGAIN.
1548 013626 005237 001430 WCHERR: INC ERRWCH ;ADD 1 TO THE HARD ERROR COUNT.
1549 013632 022737 000004 001430 CMP #4,ERRWCH ;HAS THERE BEEN 4 OF THEM?
1550 013640 001402 BEQ SYSERR ;YES PUT OUT SYSTEM ERROR MESSAGE.
1551 013642 000137 013166 JMP RETRY ;NO SO TRY AGAIN.
1552 013646 000137 013132 SYSERR: JMP SYSER
1553
1554 ;BUFFERS.
1555
1556 013652 000000 BUFF: .WORD 0
1557 013654 000030 RBUFF: .BLKW 30
1558
1559
1560
1561
1562
1563 .SBTTL RK05 CONTROL PANEL TEST
1564
1565 013734 BAD.ON:
1566 013734 104401 013742 TYPE ,65$ ;:TYPE ASCIZ STRING
1567 013740 000401 BR ,64$ ;:GET OVER THE ASCIZ
1568 ;:65$: .ASCIZ /?/
1569 ;:64$:
1570 013744 SECT.0:
1571 013744 104401 013752 TYPE ,65$ ;:TYPE ASCIZ STRING
1572 013750 000424 BR ,64$ ;:GET OVER THE ASCIZ
1573 ;:65$: .ASCIZ <15><12>/RK05 CONTROL PANEL TEST, WHICH DRIVE?/
1574 ;:64$:
1575 014022 104407 RDCHR
1576 014024 011600 MOV (SP),RO
1577 014026 104403 TYPOS
1578 014030 001 .BYTE 1
1579 014031 000 .BYTE 0
1580 014032 162700 000060 SUB #60,RO
1581 014036 100736 BMI BAD.ON
1582 014040 022700 000010 CMP #10,RO
1583 014044 003733 BLE BAD.ON
1584 014046 110037 001314 MOV8 RO,#DRIVE
1585 014052 000300 SWAB RO
1586 014054 006100 ROL RO
1587 014056 006100 ROL RO
1588 014060 006100 ROL RO
1589 014062 006100 ROL RO
1590 014064 006100 ROL RO
1591 014066 010037 001352 MOV RO,#DSKTMP
1592 014072 104401 014100 TYPE ,67$ ;:TYPE ASCIZ STRING
1593 014076 000414 BR ,66$ ;:GET OVER THE ASCIZ
1594 ;:67$: .ASCIZ <15><12>/MOUNT PACK ON DRIVE#/
1595 ;:66$:
1596 014130 013746 001314 MOV DRIVE,-(SP) ;:SAVE DRIVE FOR TYPEOUT
1597 014134 104403 TYPOS ;:GO TYPE--OCTAL ASCII
1598 014136 001 .BYTE 1 ;:TYPE 1 DIGIT(S)
1599 014137 000 .BYTE 0 ;:SUPPRESS LEADING ZEROS
1600 014140 104401 014146 TYPE ,69$ ;:TYPE ASCIZ STRING
1601 014144 000425 BR ,68$ ;:GET OVER THE ASCIZ

```

```

1603 014220 104401 014226 688: .ASCIZ (15)(12) PLACE DRIVE IN RUN ; SHOULD SEE THE RUN. /
1604 014220 104401 014226 688: TYPE 718 :: TYPE ASCIZ STRING
1605 014224 000423 014226 688: BR 705 :: GET OVER THE ASCIZ
1606 014274 104401 014302 718: .ASCIZ (15)(12) POWER, AND ON CYLINDER LAMPS LIGHT. /
1607 014274 104401 014302 718: TYPE 738 :: TYPE ASCIZ STRING
1608 014274 104401 014302 718: BR 725 :: GET OVER THE ASCIZ
1609 014300 000425 014302 718: .ASCIZ (15)(12) MAKE DRIVE WRITE ENABLE PRESS CONTINUE. /
1610 014354 000000 014354 738: .ASCIZ (15)(12) MAKE DRIVE WRITE ENABLE PRESS CONTINUE. /
1611 014354 000000 014354 738: BR 725 :: GET OVER THE ASCIZ
1612 014354 004737 016134 738: HALT PC,WRROCK : GO WRITE D'S, RD D'S, CHECK
1613 014356 104401 014370 738: JSR PC,WRROCK : GO WRITE D'S, RD D'S, CHECK
1614 014356 104401 014370 738: TYPE 755 :: TYPE ASCIZ STRING
1615 014356 000430 014370 738: BR 748 :: GET OVER THE ASCIZ
1616 014450 104401 014464 755: .ASCIZ (15)(12)(15)(12) WRITE PROTECT THE DRIVE THEN PRESS CONTINUE. /
1617 014450 000000 014464 755: HALT PC,WRPRO : GO TRY OVERWRITE
1618 014450 004737 016344 755: JSR PC,WRPRO : GO TRY OVERWRITE
1619 014450 104401 014464 755: TYPE 778 :: TYPE ASCIZ STRING
1620 014450 000426 014464 755: BR 768 :: GET OVER THE ASCIZ
1621 014540 000000 014540 778: .ASCIZ (15)(12)(15)(12) CLEAR WRITE PROTECT THEN PRESS CONTINUE. /
1622 014540 000000 014540 778: HALT PC,WRROCK : GO WRITE D'S, RD D'S, CHECK
1623 014540 004737 016134 778: JSR PC,WRROCK : GO WRITE D'S, RD D'S, CHECK
1624 014540 013777 001352 164622 778: MOV #0,DSKTMP,DRKDA : SET UP DRIVE
1625 014540 012777 000017 164606 778: MOV #17,DRKCS : FUNCTION WRITE PROTECT
1626 014540 004737 005574 164576 778: JSR PC,SMTME : GO WAIT TIME FOR RK11-C
1627 014540 005777 164576 15: STB DRKCS : IS CONTROL READY SET
1628 014540 000375 164576 15: BR #5 : IF NO, BRANCH
1629 014540 004737 016344 15: JSR PC,WRPRO : GO TRY OVERWRITE OF IERO'S
1630 014600 104401 014606 PRTTWO: TYPE 658 :: TYPE ASCIZ STRING
1631 014604 000431 014606 658: BR 648 :: GET OVER THE ASCIZ
1632 014670 104401 014676 658: .ASCIZ (15)(12)(15)(12) CAUTION! TRY TO OPEN THE DOOR, DO NOT FORCE. /
1633 014670 104401 014676 658: TYPE 678 :: TYPE ASCIZ STRING
1634 014674 000414 014676 658: BR 668 :: GET OVER THE ASCIZ
1635 014726 104401 014734 678: .ASCIZ (15)(12) DOOR SHOULD NOT OPEN! /
1636 014726 104401 014734 678: TYPE 698 :: TYPE ASCIZ STRING
1637 014732 000420 014734 678: BR 688 :: GET OVER THE ASCIZ
1638 014774 000000 014774 698: .ASCIZ (15)(12) PRESS CONTINUE WHEN FINISHED. /
1639 014774 000000 014774 698: HALT
1640 014776 104401 015004 718: TYPE 718 :: TYPE ASCIZ STRING
1641 015002 000426 015004 718: BR 705 :: GET OVER THE ASCIZ
1642 015060 104401 015066 718: .ASCIZ (15)(12)(15)(12) PUT DRIVE IN LOAD, WAIT FOR LOAD LIGHT. /
1643 015060 104401 015066 718: TYPE 738 :: TYPE ASCIZ STRING
1644 015064 000420 015066 718: BR 725 :: GET OVER THE ASCIZ
1645 015126 000000 015126 738: .ASCIZ (15)(12) PRESS CONTINUE WHEN FINISHED. /
1646 015126 000000 015126 738: HALT
1647 015130 013777 001352 164240 738: MOV #0,DSKTMP,DRKDA : SET UP DISK ADDRESS
1648 015130 012777 000015 164224 738: MOV #15,DRKCS : ISSUE A DRIVE RESET
1649 015144 004737 005574 738: JSR PC,SMTME : KILL TIME FOR RK11-C

```

```

1658 015175C 105777 164214 15: TSTB 2RKCS :CONTROL READY SET
1659 015175C 105777 164214 20: BPL 15 :IF NO, BRANCH
1660 015175C 105777 100000 164202 BIT 8BIT15,2RKER :DRE SET
1661 015175C 105777 25: BNE 25 :IF YES BRANCH
1662 015175C 104401 015174 TYPE 75S :TYPE ASCIZ STRING
1663 015175C 000422 BR 74S :GET OVER THE ASCIZ
1664 :75S: .ASCIZ <15><12> DRE=BIT15 OF RKER DID NOT SET
1665 74S:
1666 015234 032777 140000 164126 25: BIT 814000,2RKCS :DID HARD ERROR ON ERROR SET
1667 015234 001015 35: BNE 35 :BRANCH IF YES
1668 015234 104401 015252 TYPE 77S :TYPE ASCIZ STRING
1669 015250 000412 BR 76S :GET OVER THE ASCIZ
1670 :77S: .ASCIZ <15><12> ERROR DID NOT SET
1671 76S:
1672 015276 012777 000001 164064 35: MOV 81,2RKCS :ISSUE A CONTROL RESET
1673 015304 004737 005574 JSR PC,SMTME :WAIT TIME
1674 015310 105777 164054 45: TSTB 2RKCS :CONTROL READY SET
1675 015314 100375 45: BPL 45 :IF NO, BRANCH
1676 015316 032777 100000 164042 BIT 8BIT15,2RKER :DRE CLEAR
1677 015324 001425 55: BEQ 55 :IF YES BRANCH
1678 015326 104401 015334 TYPE 79S :TYPE ASCIZ STRING
1679 015332 000422 BR 78S :GET OVER THE ASCIZ
1680 :79S: .ASCIZ <15><12> CONTROL RESET DID NOT CLEAR 'DRE'
1681 78S:
1682 015400 032777 140000 163762 55: BIT 814000,2RKCS :ERROR BITS CLEAR
1683 015406 001431 X: BEQ X :IF YES BRANCH
1684 015410 104401 015416 TYPE 81S :TYPE ASCIZ STRING
1685 015414 000426 BR 80S :GET OVER THE ASCIZ
1686 :81S: .ASCIZ <15><12> CONTROL RESET DID NOT CLEAR 'ERROR', RKCS
1687 80S:
1688 015472 X:
1689 015472 104401 015500 TYPE 65S :TYPE ASCIZ STRING
1690 015476 000422 BR 64S :GET OVER THE ASCIZ
1691 :65S: .ASCIZ <15><12><15><12> OPEN THE DOOR, PUT DRIVE IN RUN
1692 64S:
1693 015544 104401 015552 TYPE 67S :TYPE ASCIZ STRING
1694 015550 000425 BR 66S :GET OVER THE ASCIZ
1695 :67S: .ASCIZ <15><12> CAUTION! IF RUN LIGHT ON ERROR! DEPRESS
1696 66S:
1697 015624 104401 015632 TYPE 69S :TYPE ASCIZ STRING
1698 015630 000426 BR 68S :GET OVER THE ASCIZ
1699 :69S: .ASCIZ <15><12> LOAD IMMEDIATELY, CONTINUE WHEN FINISHED
1700 68S:
1701 015706 000000 XX: HALT
1702 015710 104401 015716 TYPE 65S :TYPE ASCIZ STRING
1703 015714 000422 BR 64S :GET OVER THE ASCIZ
1704 :65S: .ASCIZ <15><12><15><12> REMOVE THE PACK, CLOSE THE DOOR
1705 64S:
1706 015762 104401 015770 TYPE 67S :TYPE ASCIZ STRING
1707 015766 000423 BR 66S :GET OVER THE ASCIZ
1708 :67S: .ASCIZ <15><12> PUT DRIVE IN RUN, DRIVE SHOULD NOT
1709 66S:
1710 016036 104401 016044 TYPE 69S :TYPE ASCIZ STRING
1711 016042 000423 BR 68S :GET OVER THE ASCIZ
1712 :69S: .ASCIZ <15><12> RUN... INTERLOCKS HAVE BEEN CHECKED
1713 68S:

```

```

1714 016112 104401 016120 TYPE 715 ::TYPE ASCIZ STRING
1715 016116 000404 BR 705 ::GET OVER THE ASCIZ
1716 016118 000404 ::715: .ASCIZ <15><12>/DONE!/
1717 016120 000137 001440 JMP 20START
1718 016122 005037 001362 WRRDCK: CLR 20PATTRN :MAKE A PATTERN OF ZERO'S
1719 016124 013777 001352 MOV 20DSKTMP,20RKDA :SET UP DRIVE ADDRESS
1720 016126 012777 000001 163230 MOV 01,20RKCS :ISSUE A CONTROL
1721 016128 004737 005574 163214 JSR PC,SMTME
1722 016130 105777 163204 55: TSTB 20RKCS :CONTROL READY SET
1723 016132 100375 BPL 55 :IF NO BRANCH
1724 016134 012777 001352 163202 MOV 20DSKTMP,20RKDA :SET UP DRIVE ADDRESS
1725 016136 012777 001362 163172 MOV 20PATTRN,20RKBA :GET BUSS ADDRESS
1726 016138 013777 001344 163162 MOV 20SECCNT,20RKWC :WORD COUNT 1 SECTOR
1727 016140 013777 001354 163152 MOV 20WRITCS,20RKCS :IBA + WRITE + GO
1728 016142 004737 005574 JSR PC,SMTME :KILL TIME FOR RK11-C
1729 016144 105777 163142 15: TSTB 20RKCS :CONTROL READY SET
1730 016146 100375 BPL 15 :IF NO BRANCH
1731 016148 012777 001352 163140 MOV 20DSKTMP,20RKDA :SET UP RK REGISTERS
1732 016150 012777 007336 163130 MOV 20RDBUFF,20RKBA :TO READ ONE SECTOR
1733 016152 012777 001344 163120 MOV 20SECCNT,20RKWC :TO THE READ BUFFER
1734 016154 012777 001356 163110 MOV 20READCS,20RKCS
1735 016156 004737 005574 JSR PC,SMTME :KILL TIME FOR RK11-C
1736 016158 105777 163100 25: TSTB 20RKCS :CONTROL READY SET
1737 016160 100375 BPL 25 :IF NO BRANCH
1738 016162 012704 007336 MOV 20RDBUFF,R4 :GET BUFFER TO R4
1739 016164 005005 CLR R5 :SET UP TO COMPARE
1740 016166 020524 35: CMP R5,(R4)+ :FOR ZERO'S
1741 016168 001414 BEQ 45 :IF OK, BRANCH
1742 016170 104401 016312 TYPE 655 ::TYPE ASCIZ STRING
1743 016172 000410 BR 645 ::GET OVER THE ASCIZ
1744 016332 645: ::55: .ASCIZ <15><12>/WRITE FAILED/
1745 016332 000700 BR WRRDCK :GO BACK TRY AGAIN
1746 016334 022704 010336 45: CMP 20MANSEL,R4 :DONE ALL CHECKS
1747 016336 001357 BNE 35 :IF NO BRANCH
1748 016338 000207 RTS PC :IF YES, RETURN
1749 016340 032777 000040 163012 WRRPRO: BIT 20BITS,20RKDS :BIT 5 ON
1750 016342 001021 BNE 15 :IF YES, BRANCH
1751 016344 104401 016362 TYPE 655 ::TYPE ASCIZ STRING
1752 016346 000416 BR 645 ::GET OVER THE ASCIZ
1753 016416 645: ::655: .ASCIZ <15><12>/WPS=BITS OF RKDS NOT SET
1754 016416 012737 177777 001362 15: MOV 20177777,20PATTRN :GO LOAD ALL
1755 016418 013777 001352 162744 MOV 20DSKTMP,20RKDA :RK REGISTERS
1756 016420 012777 001362 162734 MOV 20PATTRN,20RKBA :TO WRITE
1757 016422 013777 001344 162724 MOV 20SECCNT,20RKWC :ALL ONES (WITH WRITE LOCK)
1758 016424 013777 001354 162714 MOV 20WRITCS,20RKCS :OVER THE ZERO'S
1759 016426 004737 005574 JSR PC,SMTME :KILL TIME FOR RK11-C
1760 016428 105777 162704 25: TSTB 20RKCS :CONTROL READY SET?
1761 016430 100375 BPL 25 :IF NO BRANCH
1762 016432 032777 020000 162672 BIT 20BIT13,20RKER :WLO BIT SET
1763 016434 001032 BNE 35 :IF YES BRANCH
1764 016436 104401 016504 TYPE 675 ::TYPE ASCIZ STRING
1765 016438 000427 BR 665 ::GET OVER THE ASCIZ
1766 016502 665: ::675: .ASCIZ <15><12>/EXPECTED WLO=BIT13 OF RKER BUT DID NOT SET.

```

```

1770 016562
1771 016562 012777 000001 162600 66$:
1772 016570 004737 005574 3$: MOV #1,ARKCS ;DO A CONTROL RESET
1773 016574 105777 162570 4$: JSR PC,SMTME ;KILL TIME FOR RK11-C
1774 016600 100375 9PL 4$ ;CONTROL READY SET?
1775 016602 032777 020000 162556 BIT #8,T13,ARKER ;IF NO BRANCH
1776 016610 001431 BEQ R0CHK0 ;WLO BIT CLEAR
1777 016612 104401 016620 *TYPE 69$ ;IF YES, BRANCH
1778 016616 000426 BR 68$ ;TYPE ASCIZ STRING
1779 ;:69$: .ASCIZ <15><12>CONTROL RESET DID NOT CLEAR 'WLO' OF RKER/
1780 016674 68$:
1781 016674 013777 001352 162474 R0CHK0: MOV #RDSKTMP,ARKDA ;SET UP RK REGISTERS
1782 016702 012777 007336 162464 MOV #R0BUFF,ARKBA ;TO READ SECTOR 0,
1783 016710 013777 001344 162454 MOV #SECCNT,ARKWC ;CYLINDER 0, HEAD 0
1784 016716 013777 001356 162444 MOV #READCS,ARKCS ;TO ENSURE NO WRITE TOOK PLACE
1785 016724 004737 005574 JSR PC,SMTME ;KILL TIME
1786 016730 105777 162434 3$: *STB ARKCS
1787 016734 100375 3$ BPL 3$
1788 016736 012703 000005 MOV #5,R3
1789 016742 012704 007336 MOV #R0BUFF,R4 ;CHECK TO INSURE NO WRITE
1790 016746 005005 CLR R5 ;TOOK PLACE
1791 016750 020524 15: CMP R5,(R4)+ ;WITH WRITE LOCK
1792 016752 001474 BEQ 2$
1793 016754 005303 DEC R3 ;DEC THE ERROR COUNT
1794 016756 001475 BEQ 4$ ;IF ZERO BRANCH
1795 016760 104401 016766 *TYPE 65$ ;TYPE ASCIZ STRING
1796 016764 000422 BR 64$ ;GET OVER THE ASCIZ
1797 ;:65$: .ASCIZ <15><12>WRITE OCCURRED WITH WRITE PROTECT/
1798 017032 64$:
1799 017032 005744 TST -(R4)
1800 017034 104401 017042 *TYPE 67$ ;TYPE ASCIZ STRING
1801 017040 000410 BR 66$ ;GET OVER THE ASCIZ
1802 ;:67$: .ASCIZ <15><12>BUFFER ADDR=
1803 66$:
1804 017062 010446 MOV R4,-(SP)
1805 017064 104403 TYP0S
1806 017066 006 .BYTE 6
1807 017067 001 .BYTE 1
1808 017070 104401 017076 *TYPE 69$ ;TYPE ASCIZ STRING
1809 017074 000406 BR 68$ ;GET OVER THE ASCIZ
1810 ;:69$: .ASCIZ / EXPCTD=/
1811 68$:
1812 017112 010546 MOV R5,-(SP)
1813 017114 104404 TYP0N
1814 017116 104401 017124 *TYPE 71$ ;TYPE ASCIZ STRING
1815 017122 000406 BR 70$ ;GET OVER THE ASCIZ
1816 ;:71$: .ASCIZ / RECVD=/
1817 70$:
1818 017140 012446 MOV (R4)+,-(SP)
1819 017142 104404 TYP0N
1820 017144 022704 010336 2$: CMP #MANSEL,R4 ;FINISHED ALL CHECKS
1821 017150 001277 BNE 1$ ;IF NO, BRANCH
1822 017152 000207 4$: RTS PC ;RETURN

```

MAINDEC-11-DZRI-C MACY:1 27(1006) 04-OCT-76 14:01 PAGE 35
DZRI.D.P11 22-SEP-76 10:10 RKDS CONTROL PANEL TEST

F05

SEC 0057

1826

: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).

1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882

```

017154 000240          SECT.4: NOP
017156 012777 000001 162204 35:  MOV      #1,DRKCS
017164 105777 162200          *STB     DRKCS
017170 100375          BPL      35
017172 012700 020470          MOV      #DRIVO,R0
017176 005001          CLR      R1
017200 005002          CLR      R2

017202 010210          15:  MOV      R2,(R0)      ;SET UP ADDRESS TABLE
017204 010277 162166          MOV      R2,DRKDA    ;ADDRESS THE DRIVE
017210 105777 162150          TSTB     DRKDS       ;IS IT PRESENT?
017214 100021          BPL      25          ;NO

017216 104401 020324          TYPE     EM1         ;TYPE 'DRIVE'
017222 010146          MOV      R1,-(SP)    ;TYPE OUT DRIVE #
017224 104402          TYP0C
017226 104401 020335          TYPE     EM2         ;TYPE 'ON LINE'
017232 052710 00030C          BIS      #BIT6+BIT7,(R0) ;SET BITS INDICATING THIS
                                           ;DRIVE PRESENT

017236 012777 000015 162124          MOV      #15,DRKCS   ;ISSUE A DRIVE RESET
017244 004737 005574          JSR      PC,SMTME    ;ALLOW SOME TIME
017250 032777 000100 162106 45:  BIT      #RWS,DRKDS  ;WAIT FOR RWS RDY
017256 001774          BEQ     45

017260 005720          25:  TST      (R0)+
017262 005201          INC     R1
017264 062702 020000          ADD     #20000,R2    ;NXT DRIVE
017270 001344          BNE     15          ;ALL DONE?

017272 104401 001161          TYPE     .SCLF

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

1872 017276 012700 020470  BEGCT:  MOV      #DRIVO,R0      ;INITIALIZE POINTERS
1873 017302 005001          CLR      R1
1875 017304 011077 162066  BEGCT1: MOV      (R0),DRKDA     ;ADDRESS A DRIVE
1876 017310 042777 017777 162060  BIC      #17777,DRKDA    ;MASK OUT NON DR# BITS
1877 017316 105777 162042          TSTB     DRKDS         ;IS THIS DRIVE ON LINE?
1878 017322 100044          BPL      15            ;NO
1879          YES
1880 017324 105710          TSTB     (R0)         ;WAS IT 'ON LINE' LAST TIME?
1881 017326 100454          BMI     NXT1          ;YES, NO MESSAGE TO REPORT
1882 017330 052710 000200          BIS      #BIT7,(R0)    ;IT CHANGED FROM OFF LINE TO ON
    
```

```

1883 017334 104401 020324 TYPE EM1 :LINE, REPORT MESSAGE
1884 017340 010146 MOV R1,-(SP)
1885 017342 104402 TYPOC
1886 017344 104401 020335 TYPE EM2 :TYPE 'ON LINE'
1887 017350 032777 000040 162006 BIT #WPS,DRKDS ;WRITE ENABLED?
1888 017356 001417 BEG 25 ;YES, OK
1889 017360 104401 017366 TYPE 655 ;TYPE ASCIZ STRING
1890 017364 000414 BR 645 ;GET OVER THE ASCIZ
1891 :655: .ASCIZ <15><12>/EROR,NOT WRT ENABLED/
1892 017416 645:
1893 017416 012777 000017 161744 MOV #17,DRKCS ;WRITE PROT THE DISK
1894 017424 105777 161740 TSTB DRKCS
1895 017430 100375 BPL 35
1896 017432 000412 BR NXT1
1897
1898 017434 105710 15: TSTB (R0) ;WAS THIS DRIVE OFF LINE LAST
1899 017436 100010 BPL NXT1 ;TIME? BRNCH IF YES
1900 017440 104401 020324 TYPE EM1 ;IF NOT, REPORT THE CHANGE
1901 017444 010146 MOV R1,-(SP) ;TYPE DRIVE #
1902 017446 104402 TYPOC
1903 017450 104401 020347 TYPE EM3 ;TYPE 'OFF LINE'
1904 017454 042710 000200 BIC #BIT7,(R0) ;CLEAR BIT TO INDICATE THIS
1905 ;DRIVE 'OFF LINE'
1906
1907 :THIS CODE CHECKS 'WPS' BIT FOR EVERY DRIVE THAT IS IN 'DRY'
1908 ;CONDITION (ON LINE). IT REPORTS ANY CHANGE IN THE CONDITION OF
1909 ;THE 'WPS' BIT. IF THERE WAS NO CHANGE FROM LAST TIME NOTHING
1910 ;IS REPORTED. AT THE TIME OF ENTRY R0 POINTS TO DRIVE FLAG.
1911
1912 017460 105777 161700 NXT1: TSTB DRKDS ;IS THIS DRIVE PRESENT?
1913 ;RKDA CONTAINS THE DRV #
1914 017464 100033 BPL NXT2 ;NO, SKIP CHECKING
1915
1916 017466 032777 000040 161670 BIT #WPS,DRKDS ;WPS BIT SET?
1917 017474 001014 BNE 15 ;YES
1918 ;WPS BIT CLEAR
1919 017476 032710 000004 BIT #BIT2,(R0) ;WAS IT CLR LAST TIME ALSO?
1920 017502 001424 BEG NXT2 ;YES, NOTHING TO REPORT.
1921 ;WPS CHANGED FROM 'SET'
1922 017504 104401 020324 TYPE EM1 ;TO 'CLR', REPORT IT
1923 017510 010146 MOV R1,-(SP) ;TYPE DRIVE #
1924 017512 104402 TYPOC
1925 017514 042710 000004 BIC #BIT2,(R0) ;INDICATE THAT 'WPS' IS CLEAR
1926 017520 104401 020375 TYPE EM5 ;TYPE 'WPS CLEAR'
1927 017524 000413 BR NXT2
1928 ;WPS BIT IS SET
1929 017526 032710 000004 15: BIT #BIT2,(R0) ;WAS IT SET LAST TIME ALSO?
1930 017532 001010 BNE NXT2 ;YES, NOTHING TO REPORT.
1931 ;WPS CHANGED, FROM 'CLR' TO
1932 ;'SET', REPORT THIS CHANGE
1933 017534 104401 020324 TYPE EM1 ;TYPE 'DRIVE'
1934 017540 010146 MOV R1,-(SP) ;TYPE DRIVE #
1935 017542 104402 TYPOC
1936 017544 104401 020362 TYPE EM4 ;TYPE 'WPS SET'
1937 017550 052710 000004 BIC #BIT2,(R0) ;SET FLAG BIT INDICATING WPS SET
1938

```

```

1939 :THIS CODE PERFORMS A SEEK FUNCTION ON A DRIVE AND CHECKS IF
1940 :THE 'DPL' BIT SET AS A RESULT, (IF THE POWER WAS CUT OFF
1941 :FROM THE DRIVE). NOTE THAT ONLY THOSE DRIVES ARE
1942 :CHECKED WHICH WERE FOUND TO BE PRESENT AT BEGINNING (WHEN
1943 :THIS TEST WAS ENTERED). SEEK IS DONE TO CYLINDER 1.
1944 :AT THE TIME OF ENTRY RD POINTS TO THE DRIVE FLAG.
1945
1946 017354 032710 000100 NXT2: BIT #BIT6,(RD) ;WAS THIS DRIVE PRESENT AT BEGNG
1947 017560 001403 BEQ 45 ;NO
1948 017562 105777 161576 TSTB 2RKDS ;IS IT PRESENT NOW?
1949 017566 100402 BMI 35 ;YES
1950 017570 000137 020226 45: JMP DN1DRV ;IF NOT SKIP THIS CHECK
1951
1952 017574 052777 000040 161574 35: BIS #40,2RKDA ;RKDA ALREADY HAS THE DRV #
1953 017602 012777 000011 161560 MOV #11,2RKCS ;SET CYL 1 ADDRESS
1954 ;SEEK, GO
1955
1956 017610 105777 161554 15: TSTB 2RKCS ;WAIT FOR CONTROL RDY?
1957 017614 100375 BPL 15 ;SOMETHING WRONG IF CNTAL RDY
1958 ;DOES NOT COME BACK
1959 017616 032777 010000 161540 BIT #DPL,2RKDS ;DPL BIT SET?
1960 017624 001414 BEQ 25 ;NO
1961 ;YES, DPL SET
1962 017626 032710 000001 BIT #BIT0,(RD) ;WAS 'DPL' SET LAST TIME ALSO?
1963 017632 001167 BNE CLRDP1 ;YES, NOTHING TO REPORT.
1964 ;DPL CHANGED, GOT SET THIS
1965 017634 104401 020324 TYPE EM1 ;TIME, REPORT IT
1966 017640 010146 MOV R1,-(SP)
1967 017642 104402 TYPOC
1968 017644 104401 020413 TYPE EM6 ;TYPE 'POWER LO'
1969 017650 052710 000001 BIS #BIT0,(RD) ;SET FLAG BIT INDICATING THAT
1970 ;DPL SET THIS TIME
1971 017654 000556 BR CLRDP1
1972
1973 017656 032710 000001 25: BIT #BIT0,(RD) ;'DPL' BIT IS CLEAR
1974 017662 001410 BEQ WATSK ;WAS 'DPL' CLEAR LAST TIME ALSO?
1975 ;YES, NOTHING TO REPORT
1976 017664 104401 020324 TYPE EM1 ;REPORT THAT 'DPL' BIT CHANGED.
1977 017670 010146 MOV R1,-(SP) ;FROM SET TO CLEAR
1978 017672 104402 TYPOC
1979 017674 104401 020434 TYPE EM7 ;TYPE 'POWER UP'
1980 017700 042710 000001 BIC #BIT0,(RD) ;SET FLAG BIT INDICATING THAT DPL
1981 ;IS CLEAR THIS TIME
1982
1983 :THIS CODE WAITS FOR THE SEEK (DONE ABOVE) TO FINISH. WAITING
1984 :TIME IS APPROX. 50 MS (FOR THE WORST CASE). IF R/W/S RDY
1985 :DOES NOT SET WITHIN 50 MS, THEN IT IS ASSUMED THAT A 'SIN'
1986 :IS POSSIBLE AND THE PROGRAM WAITS FOR 1450 MS MORE, SO THAT
1987 :THE 'SIN' CAN SET. IF 'SIN' DOES NOT SET WITHIN THIS
1988 :TIME AN ERROR IS REPORTED:
1989 : SIN DIDN'T OCCUR
1990 :IF R/W/S RDY SETS WITHIN 50 MS THE PROGRAM PROCEEDS TO
1991 :CHECK THE NEXT DRIVE.
1992
1993 017704 012705 164220 WATSK: MOV #-6000,.R5 ;SET COUNT TO WAIT FOR
1994 ;50 MS

```

```

1995 017710 032777 000100 161446 18: BIT #RWS,ARKDS ;R/W/S. RDY SET?
1996 017716 001042 BNE 38 ;YES
1997 017720 005205 INC R5 ;WAIT
1998 017722 001372 BNE 18
1999 ;50 MS OVER, R/W/S RDY
2000 ;DIDN'T SET. WAIT FOR
2001 ;SIN TO SET.
2002 017724 005004 CLR R4
2003 017726 012705 MOV #17777,R5 ;SET UP COUNT
2004 017732 032777 001000 161424 28: BIT #SIN,ARKDS ;SIN SET?
2005 017740 001045 BNE SINST ;YES
2006 017742 005305 DEC R5 ;WAIT
2007 017744 001372 BNE 28
2008 017746 005704 TS+ R4
2009 017750 001002 BNE 48
2010 017752 005204 INC R4
2011 017754 000766 BR 28
2012 ;1500 MS ELAPSED, BUT SIN
2013 ;DIDN'T SET. ERROR!
2014 017756 48:
2015 017756 104401 017754 TYPE ,655 ;:TYPE ASCIZ STRING
2016 017762 000415 BR 645 ;:GET OVER THE ASCIZ
2017 ;:655: .ASCIZ <15><12>/SIN DIDN'T OCCUR, DRIVE/
2018 645:
2019 020016 MOV R1,-(SP) ;TYPE DRIVE #
2020 020020 TYPOC
2021 020022 000501 BR DNIDRV
2022 020024 032710 000002 38: BIT #BIT1,(R0) ;DID R/W/S RDY SET LAST TIME?
2023 020030 001476 BEQ DNIDRV ;YES
2024 020032 042710 000002 BIC #BIT1,(R0) ;CLR FLAG INDICATING THAT SEEK IS OK
2025 020036 104401 020324 TYPE ,EM1 ;REPORT THAT SEEK IS OK
2026 020042 010146 MOV R1,-(SP)
2027 020044 104402 TYPOC
2028 020046 104401 020455 TYPE ,EM9
2029 020052 000465 BR DNIDRV
2030
2031 ;IF SIN SET, DO DRIVE RESET AND CLEAR IT
2032
2033 SINST: BIT #BIT1,(R0) ;DID 'SIN' SET LAST TIME ALSO?
2034 020060 001010 BNE 48 ;YES. NOTHING TO REPORT
2035 020062 052710 000002 BIS #BIT1,(R0) ;SET FLAG INDICATING THAT
2036 020066 104401 020324 TYPE ,EM1 ;'SIN' SET, AND REPORT THE CHANGE
2037 020072 010146 MOV R1,-(SP)
2038 020074 104402 TYPOC
2039 020076 104401 020447 TYPE ,EMB ;TYPE 'SIN'
2040
2041 020102 017705 161270 48: MOV ARKDA,R5 ;SAVE RKDA
2042 020106 012777 000001 161254 MOV #1,ARKCS ;DO CONTROL RESET
2043 020114 105777 161250 18: TSTB ARKCS ;WAIT FOR CONTROL RDY
2044 020120 100375 BPL 18
2045 020122 010577 161250 MOV R5,ARKDA
2046 020126 012777 000015 161234 MOV #15,ARKCS ;DO DRIVE RESET. RKDA
2047 ;ALREADY HAS THE DRIVE #
2048 020134 105777 161230 28: TSTB ARKCS ;WAIT FOR CNTRL RDY
2049 020140 100375 BPL 28
2050 020142 005005 CLR R5

```

```

2051 020144 032777 000100 161212 3$: BIT #RWS,DRKDS ;R/W/S SET?
2052 020152 001025 BNE DNIDRV ;YES
2053 020154 005205 INC RS
2054 020156 001372 BNE 3$ ;WAIT FOR R/W/S RDY
2055 ;REPORT ERROR. R/W/S RDY CLR
2056 020160 104401 020166 TYPE 65$ ;TYPE ASCIZ STRING
2057 020164 000411 BR 64$ ;GET OVER THE ASCIZ
2058 ;65$: .ASCIZ <15><12>/RWS RDY NOT SET/
2059 64$:
2060
2061 020210 000406 BR DNIDRV
2062
2063 ;IF DPL SET CLEAR THE ERROR BY DOING CONTROL RESET.
2064
2065 020212 012777 000001 161150 CLRDPL: MOV #1,DRKCS ;CONTROL RESET
2066 020220 105777 161144 1$: TSTB DRKCS ;WAIT FOR CNTRL RDY
2067 020224 100375 BPL 1$
2068 ;AT THIS STAGE THE DRIVE (* IN RKDA) HAS BEEN CHECKED
2069 ;FOR DRY, WPS, DPL, & SIN. THE POINTERS ARE INCREMENTED
2070 ;AND THE SAME CHECKS WILL BE DONE ON THE NEXT
2071 ;DRIVE & THEN THE NEXT ONE & SO ON. NOTE THAT
2072 ;THIS SUB-PROGRAM KEEPS ON CYCLING THROUGH
2073 ;ALL THE DRIVES. AT THE TIME OF ENTRY (HERE)
2074 ;RD POINTS TO THE FLAG FOR THE DRIVE THAT WAS
2075 ;JUST CHECKED. BEFORE GOING ON TO THE NEXT
2076 ;DRIVE THE HEADS ARE BROUGHT BACK TO CYLINDER
2077 ;D (FOR THE NEXT CYCLE).
2078
2079 020226 011077 161144 DNIDRV: MOV (RD),DRKDA ;GET DRIVE #
2080 020232 105777 161126 TSTB DRKDS ;DRIVE PRESENT?
2081 020236 100017 BPL 3$ ;NO
2082 020240 042777 017777 161130 BIC #1777,DRKDA ;CYL ADRES = C
2083 020246 012777 000011 161114 MOV #11,DRKCS ;GO, SEEK
2084
2085 020254 105777 161110 1$: TSTB DRKCS ;WAIT FOR CNTRL RDY
2086 020260 100375 BPL 1$
2087
2088 020262 004737 005574 JSR PC,SMTME
2089 020266 032777 000100 161070 4$: BIT #RWS,DRKDS
2090 020274 001774 BEQ 4$
2091
2092 020276 005720 3$: TST (RD)+ ;INCREMENT POINTERS TO
2093 020300 005201 INC R1 ;NEXT DRIVE
2094 020302 020127 000010 CMP R1,#10 ;ALL DONE, THIS CYCLE?
2095 020306 001402 BEQ 2$ ;YES
2096 020310 000137 017304 JMP BEGCT1 ;GO DO NEXT DRIVE
2097 020314 000137 017276 2$: JMP BEGCT ;RESTART THE CYCLE OVER
2098 ;AGAIN
2099
2100
2101
2102 020320 000000 SHFCNT: .WORD 0
2103 020322 000000 DRVCNT: .WORD 0
2104 ;MESSAGES
2105 020324 005015 051104 053111 EMI: .ASCIZ <15><12>/DRIVE /
2106 020332 020105 000

```


.SBTTL HEAD ALIGNMENT ROUTINE

2153
2154
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

```

:HEAD ALIGNMENT ROUTINE
:THIS MAINTAINANCE ROUTINE IS HELPFUL IN HEAD ALIGNMENT. UPON ENTRY
:THE QUESTION - DRIVE? - IS ASKED. THE USER SHOULD REPLY WITH THE
:DRIVE NUMBER THAT IS TO BE ALIGNED. IF THE DRIVE IS AN RK-05F
:THE LETTER 'F' IS ADDED AS A SUFFIX. FOR SELECTING SURFACE 0
:PUT SWO=0, FOR SELECTING SURFACE 1 PUT SWO=1. SET SWI =1 TO SELECT
:CYLINDER 64. SET SWI=0 TO SELECT CYLINDER 105.
:IF THE DRIVE IS AN RK-05F, CYLINDER 64 BECOMES CYLINDER 130
:OF THE EVEN DRIVE, AND CYLINDER 105 BECOMES CYLINDER 5 OF THE ODD DRIVE
:THE HEADS ARE PLACED ON THE SELECTED CYLINDER AND DATA IS READ
:CONTINUOSLY FROM THE CYLINDER (SECTOR 0)
: THE UPPER OR LOWER HEAD AND 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?).
    
```

```

020510 000240
020512 104401 020520
020516 000426
020574
020574 104401 020602
020600 000432
020666
020666 104401 020674
020672 000427
020752
020752 104401 021370
020756 005037 021364
020762 104410
020764 012601
020766 112100
020770 162700 000060
020774 002766
020776 022700 000067
021002 002763
021004 000241
021006 006000
021010 006000
021012 006000
021014 006000
021016 010037 020470
021022 112100
021024 001412
021026 020027 000106
021032 001347
021034 105711
021036 001345
021040 042737 020000 020470
021046 005237 021364
021052 013700 020470
    
```

```

SECT.5: NOP
        TYPE      65$           ;;TYPE ASCIZ STRING
        BR        64$           ;;GET OVER THE ASCIZ
        65$: .ASCIZ <15><12>/SET SWO=0 FOR SURFACE 0, SWO=1 FOR SUR 1./
        64$:
        TYPE      67$           ;;TYPE ASCIZ STRING
        BR        66$           ;;GET OVER THE ASCIZ
        67$: .ASCIZ <15><12>/SET SWI=0 FOR CYLINDER 105, SWI=1 FOR CYLINDER 64/
        66$:
        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-05F
        RDLIN      ;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        $$             ;ALL DONE IF C.R.
        CMP        RO,#'F         ;IS IT F?
        BNE        HDALGN        ;NO, SO ERROR
        TSTB      (R1)           ;NEXT CHAR MUST BE C.R.
        BNE        HDALGN        ;ELSE, ERROR
        BIC        #BIT13,DRIVO    ;USE EVEN DRIVE IF RK-05F
        INC        FFLAG          ;SHOW F TYPE DRIVE
        MOV        DRIVO,RO        ;DRIVE ADDR TO RO
        $$:
    
```

2209	021056	017737	160056	021366		MOV	0SWR,SWTCH	:HOLD SWITCHES
2210	021064	042737	177775	021366		BIC	#177775,SWTCH	:WANT SW1 ONLY
2211	021072	001005				BNE	7\$:SW1 SET, SO LOW CYLINDER
2212	021074	005737	021364			TST	FFLAG	:F DRIVE?
2213	021100	001402				BEQ	7\$:NO
2214	021102	052700	020000			BIS	#BIT13,RO	:ODD DRIVE IF HIGH TRACK OF F
2215	021106	010077	160264		7\$:	MOV	RO,0RKDA	:ADDRESS DRIVE
2216	021112	012777	000017	160250		MOV	#17,0RKCS	:WRITE PROTECT
2217	021120	105777	160244		8\$:	TSTB	0RKCS	
2218	021124	100375				BPL	8\$:WAIT FOR DRIVE READY
2219	021126	012777	000001	160234		MOV	#1,0RKCS	:RESET CONTROLLER
2220	021134	105777	160230		9\$:	TSTB	0RKCS	
2221	021140	100375				BPL	9\$:WAIT FOR READY
2222	021142	005737	021364			TST	FFLAG	:F DRIVE?
2223	021146	001410				BEQ	13\$:NO
2224	021150	012701	000240			MOV	#5,*40,R1	:TRACK 5 OF HIGH
2225	021154	005737	021366			TST	SWTCH	:SW1 SET?
2226	021160	001412				BEQ	10\$:YES SO TEST TRACK 8 OF DRIVE HIGH
2227	021162	062701	010100			ADD	#130,*40,R1	:TRACK 130. IF SW1 SET
2228	021166	000407				BR	10\$	
2229	021170	012701	004000		13\$:	MOV	#64,*40,R1	:CYLINDER 64 IF NOT F
2230	021174	005737	021366			TST	SWTCH	:SW1 SET?
2231	021200	001002				BNE	10\$:YES, SO CYLINDER 64
2232	021202	062701	002440			ADD	#41,*40,R1	:CYLINDER 105
2233	021206	005777	160156		10\$:	TST	0RKCS	:ANY ERROR?
2234	021212	100006				BPL	11\$:NO, CONTINUE
2235	021214	012777	000001	160146		MOV	#1,0RKCS	:RESET
2236	021222	105777	160142		12\$:	TSTB	0RKCS	
2237	021226	100375				BPL	12\$:WAIT FOR READY
2238	021230	017702	157704		11\$:	MOV	0SWR,R2	:SWITCH REG TO R2
2239	021234	042702	177775			BIC	#177775,R2	:SW1 ONLY
2240	021240	020237	021366			CMP	R2,SWTCH	:ANY CHANGE SINCE LAST?
2241	021244	001302				BNE	5\$:YES, GO SET-UP ADDR AGAIN
2242	021246	010077	160124		6\$:	MOV	RO,0RKDA	:ADDRESS THE DRIVE
2243	021252	012777	000017	160110		MOV	#17,0RKCS	:WRITE PROTECT THE DRIVE
2244	021260	105777	160104			TSTB	0RKCS	:WAIT FOR CONTROL RDY
2245	021264	100375				BPL	.-4	
2246	021266	042700	000020		4\$:	BIC	#20,RO	:CLEAR TRACK ADDR
2247	021272	032777	000001	157640		BIT	#1,0SWR	:SW0 SET?
2248	021300	001402				BEQ	2\$:NO TEST TRACK 0
2249	021302	052700	000020			BIS	#20,RO	:TEST TRACK 1
2250	021306	042700	017740		2\$:	BIC	#17740,RO	:CLEAR CYLINDER ADDR
2251	021312	050100				BIS	R1,RO	:PUT CYLINDER ADDR IN ADDR
2252	021314	010077	160056			MOV	RO,0RKDA	:ADRES THE DRIVE
2253	021320	012777	177400	160044		MOV	#-400,0RKWC	:READ 1 SECTOR
2254	021326	012777	007336	160040		MOV	#RDBUFF,0RKBA	:INTO THIS BUFFER
2255	021334	012777	000005	160026		MOV	#5,0RKCS	:READ, GO
2256								
2257	021342	105777	160022		3\$:	TSTB	0RKCS	:DONE?
2258	021346	100375				BPL	3\$:NO
2259								
2260	021350	032777	177774	157562		BIT	#177774,0SWR	:EXIT OUT?
2261	021356	001713				BEQ	10\$:NO, CONTINUE ON THIS DRIVE
2262	021360	000137	020752			JMP	H0ALGN	:YES, GET NEW DRIVE
2263								
2264	021364	000000						

FFLAG: 0

001366 000000
001370 005015
001376 037505
051104 053111
000

SWTCH: 0
EPID: .ASCIZ (15) (12) DRIVE

.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 (DRCP
: POWER) IS GIVEN. AFTER RECEIVING THIS MESSAGE THE USER SHOULD
: DRCP POWER FROM THE DRIVE. ON SENSING A LOSS OF POWER, 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.

021402

.EVEN

001400 005000
001404 005037
001410 010077
001414 105777
001420 100406
001426 005237
001432 062700
001438 001366
001444 000762
001450 104401
001456 013746
001462 104402
001468 012777
001474 105777
001480 100376
001486 005005
001466 013701
001472 013702
001478 013703
001484 013704
001490 010011
001496 010537
001502 012713
001508 012712
001514 012714
001520 105714
001526 100376
001532
001538 005205
001544 020527
001550 000C20
001556 001362

SECT.6: CLR R0 ; INITIALIZE DRIVE #
CLR DRIVO
BS: MOV R0, DRKDA ; IS IT PRESENT?
TSTB DRKDS ; IF NOT SKIP
BMI 95
INC DRIVO
105: ADD #20000, R0
BNE 95
BR SECT.6
95: TYPE EM1 ; GET DRIVE #
MOV DRIVO, -(SP)
MOV #1, DRKCS ; CONTROL RESET
TSTB DRKCS
BPL -4
CLR R5 ; INITIALIZE PATTERN TO BE WRITTEN
; 0 ON CYL 0, 1 ON CYL 1, ETC
MOV RKDA, R1
MOV RKWC, R2
MOV RKBA, R3
MOV RKCS, R4
MOV R0, DR1
15: MOV R5, BUFR ; FILL THE PATTERN IN DATA BUFFER.
MOV #BUFR, DR3 ; BUS ADRES
MOV #-14000, DR2 ; WRITE 1 CYL (256X12X2 WORDS)
MOV #4003, DR4 ; WRITE GO, IBA SET
TSTB DR4 ; WAIT FOR CONTROL READY
BPL -2
; DONE
INC R5 ; WRITTEN ALL 15 CYLINDERS?
CMP R5, #20 ; IF NOT, GO BAK
BNE 15

157712

```

021544 010005      MOV      R0,R5      ;DRIVE #
021546 000500      BIS      #500,R5    ;CYL 10
021548 000012 022046    MOV      #12, R5    ;PATTERN TO BE WRITTEN
021550 010511      MOV      R5, R1     ;ADRES THE DISK
021552 012713      MOV      #14000, R2 ;WORD COUNT= 1 CYLINDER
021554 012713      MOV      #BUFR, R3   ;BUS ADRES
021556 012713      MOV      #4003, R4  ;WRITE, GO, IBA
021558 032777 157560 25:    BIT      #RWS, R4   ;WAIT FOR THE HEADS TO SETTLE
021560 001774      BEQ     25         ;ON CYL 10
021562 104401 021614      TYPE   655        ;TYPE ASCIZ STRING
021564 000406      BR      645       ;GET OVER THE ASCIZ
;:655: .ASCIZ <DROP POWER>
021630 645:
021630 000407      BR      55
021632 010511 35:    MOV      R5, R1     ;ACRES THE DISK
021634 012713      MOV      #14000, R2 ;WORD COUNT= 1 CYLINDER
021636 012713      MOV      #BUFR, R3   ;BUS ADRES
021638 012713      MOV      #4003, R4  ;WRITE, GO, IBA
021640 105714 55:    TSTB   R4         ;WAIT FOR CONTROL READY
021642 000376      BR      -2
021644 005714      TST    R4         ;WAIT FOR DRIVE POWER TO GO DOWN,
;:OTHERWISE, KEEP ON WRITING ON CYL 10
021656 100365      BPL     35
;:IF DRIVE POWER LOSS WAS SENSED,
;:ASK TO PUT POWER ON.
021662 104401 021666      TYPE   675
021664 000406      BR      665
;:675: .ASCIZ <15> <12> POWER ON>
;:GET OVER THE ASCIZ
021702 105777 157456      TSTB   R4KDS     ;WAIT FOR DRIVE READY
021704 100375      BPL     -4
021706 012714 000001      MOV      #1, R4    ;CONTROL RESET, CLEAR ERROR
021708 105714      TSTB   R4
021710 100376      BPL     -2
021712 010077 157452      MOV      R0, R4KDA ;INITIALIZE PATTERN
021714 005005      CLR     R5
021716 010537 022046 65:    MOV      R5, BUFR
021718 012713      MOV      #BUFR, R3
021720 012713      MOV      #14000, R2 ;WRITE CHECK, GO, IBA
021722 012712 164000      MOV      #4007, R4
021724 012714      TSTB   R4
021726 105714      BPL     -2
021728 005714      TST    R4         ;ANY ERROR?
021730 100023      BPL     75         ;NO
021732 104401 021764      TYPE   695
021734 000416      BR      685
;:695: .ASCIZ <15> <12> ERROR, ON PWR-UP, RKDA=/
;:GET OVER THE ASCIZ
022020 011146 75:    MOV      R1, -(SP)
022022 104402      TYPOC
022024 005205      INC     R5
022026 020527 000012      CMP     R5, #12   ;CYLINDER 10
022028 001774      BEQ     75         ;BR IF YES, INCREMENT AGAIN
022030 020527 000020      CMP     R5, #20

```

022040 001332
022042 000137 021422
022046 000000

BNE 68
JMP 108
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

022050 105737 001157
022054 100002
022056 000000
022060 000407
022062 010046
022064 017600 000002
022070 112046
022072 001005
022074 005726
022076 012600
022100 062716 000002
022104 000002
022106 122716 000011
022112 001430
022114 122716 000200
022120 001006
022122 005726
022124 104401
022126 001161
022130 105037 022264
022134 000755
022136 004737 022220
022142 123726 001156
022146 001350
022150 013746 001154

022154 105366 000001
022160 002770
022162 004737 022220
022166 105337 022264
022172 000770

STYPE: TSTB \$TFPLG :: IS THERE A TERMINAL?
BPL 18 :: BR IF YES
HALT :: HALT HERE IF NO TERMINAL
BR 38 :: LEAVE
15: MOV RD, -(SP) :: SAVE RD
MOV 22(SP), RD :: GET ADDRESS OF ASCIZ STRING
25: MOVB (RD)+, -(SP) :: PUSH CHARACTER TO BE TYPED ONTO STACK
BNE 48 :: BR IF IT ISN'T THE TERMINATOR
TST (SP)+ :: IF TERMINATOR POP IT OFF THE STACK
60\$: MOV (SP)+, RD :: RESTORE RD
35: ADD #2, (SP) :: ADJUST RETURN PC
RTI :: RETURN
45: CMPB #HT, (SP) :: BRANCH IF <HT>
BEG 88
CMPB #CRLF, (SP) :: BRANCH IF NOT <CRLF>
BNE 55
TST (SP)+ :: POP <CR><LF> EQUIV
TYPE TYPE A CR AND LF
CLRB \$CHARCNT :: CLEAR CHARACTER COUNT
BR 25 :: GET NEXT CHARACTER
55: JSR PC, STYPEC :: GO TYPE THIS CHARACTER
65: CMPB \$FILLC, (SP)+ :: IS IT TIME FOR FILLER CHARS.?
BNE 25 :: IF NO GO GET NEXT CHAR.
MOV \$NULL, -(SP) :: GET # OF FILLER CHARS. NEEDED
AND THE NULL CHAR.
75: DECB 1(SP) :: DOES A NULL NEED TO BE TYPED?
BLT 65 :: BR IF NO--GO POP THE NULL OFF OF STACK
JSR PC, STYPEC :: GO TYPE A NULL
DECB \$CHARCNT :: DO NOT COUNT AS A COUNT
BR 75 :: LOOP

F06

NOEC-11-DZKAI-0
DZKAI.P11 22-SEP-76

MACY11 27(1006) 10:10

04-OCT-76 14:01 PAGE 49
BINARY TO OCTAL (ASCII) AND TYPE

SEC 0370

```

010546      MOV      R5, -(SP)          ;;SAVE R5
0113704     MOVVB   $OMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
0113704     NEG      R4              ;;
0113704     ADD      #6,R4          ;;SUBTRACT IT FOR MAX. ALLOWED
0113704     MOVVB   R4,$OMODE        ;;SAVE IT FOR USE
0113704     MOVVB   $OFILL,R4       ;;GET THE ZERO FILL SWITCH
0113704     MOV      12(SP),R5      ;;PICKUP THE INPUT NUMBER
0113704     CLR      R3            ;;CLEAR THE OUTPUT WORD
0113704     ROL     R5             ;;ROTATE MSB INTO "C"
0113704     BR      35            ;;GO DO MSB
0113704     ROL     R5            ;;FORM THIS DIGIT
0113704     ROL     R5            ;;
0113704     ROL     R5            ;;
0113704     MCV     R5,R3          ;;
0113704     ROL     R3            ;;GET LSB OF THIS DIGIT
0113704     DECB   $OMODE          ;;TYPE THIS DIGIT?
0113704     BPL     75            ;;BR IF NO
0113704     BIC     #177770,R3     ;;GET RID OF JUNK
0113704     ONE    45            ;;TEST FOR 0
0113704     TST    R4             ;;SUPPRESS THIS 0?
0113704     BEQ    55            ;;BR IF YES
0113704     INC    R4             ;;DON'T SUPPRESS ANYMORE 0'S
0113704     BIS    #'0,R3         ;;MAKE THIS DIGIT ASCII
0113704     BIS    #' ,R3         ;;MAKE ASCII IF NOT ALREADY
0113704     MOVVB   R3,$S         ;;SAVE FOR TYPING
0113704     TYPE   $S             ;;GO TYPE THIS DIGIT
0113704     DECB   $OCNT          ;;COUNT BY 1
0113704     BGT    25            ;;BR IF MORE TO DO
0113704     BLT    65            ;;BR IF DONE
0113704     INC    R4             ;;INSURE LAST DIGIT ISN'T A BLANK
0113704     BR      25            ;;GO DO THE LAST DIGIT
0113704     MOV     (SP)+,R5       ;;RESTORE R5
0113704     MOV     (SP)+,R4       ;;RESTORE R4
0113704     MOV     (SP)+,R3       ;;RESTORE R3
0113704     MOV     2(SP),4(SP)    ;;SET THE STACK FOR RETURNING
0113704     MOV     (SP)+,(SP)
0113704     RTI                    ;;RETURN
0113704     .BYTE   0              ;;STORAGE FOR ASCII DIGIT
0113704     .BYTE   0              ;;TERMINATOR FOR TYPE ROUTINE
0113704     $OCNT: .BYTE 0          ;;OCTAL DIGIT COUNTER
0113704     $OFILL: .BYTE 0         ;;ZERO FILL SWITCH
0113704     $OMODE: .WORD  J         ;;NUMBER OF DIGITS TO TYPE
0113704     .SBTTL TTY INPUT ROUTINE

;*****
;ENABL  LSB
$TKCNT: .WORD  0          ;;NUMBER OF ITEMS IN QUEUE
$TKQIN: .WORD  0          ;;INPUT POINTER
$TKQOUT: .WORD  0         ;;OUTPUT PCINTER
$TKQSRT: .BLKB 1         ;;TTY KEYBOARD QUEUE
$TKQEND=.
.EVEN

;*TK INITIALIZE ROUTINE
;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT

```

2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600

022526 005037 022516
022532 012737 022524 022520
022540 013737 022520 022522
022546 012737 022576 000063
022554 012737 000200 000062
022562 005777 156360
022566 012777 000100 156350
022574 000207

022576 117746 156344
022582 042716 177600
022586 021627 000007
022592 001004
022594 022737 000176 001140
022598 001500

022624
022624 022737 000001 022516
022632 001004
022634 104401 023646
022640 005726
022642 000451
022644 021627 000023
022650 001021
022652 005077 156266
022656 005726
022660 105777 156260
022664 100375
022666 117746 156254
022672 042716 177600
022676 022627 000021
022702 001366
022704 012777 000100 156232
022712 000002
022714 005237 022516
022720 021627 000140
022724 002405
022726 021627 000175
022732 003002
022734 042716 000040
022740 112677 177554
022744 005237 022520
022750 023727 022520 022525
022756 001003
022760 012737 022524 022520
022766 000002

```

:
: *CALL:
: * JSR PC,$TKINT
: * RETURN
:
$TKINT: CLR $TKCNT ;; CLEAR COUNT OF ITEMS IN QUEUE
MOV $STKQSR,$TKQIN ;; MOVE THE STARTING ADDRESS OF THE
MOV $TKQIN,$TKQOUT ;; QUEUE INTO THE INPUT & OUTPUT POINTERS.
MOV $STKSRV,$TKVEC ;; INITIALIZE THE KEYBOARD VECTOR
MOV $200,$TKVEC+2 ;; "BR" LEVEL 4
TST $TKB ;; CLEAR DONE FLAG
MOV $100,$TKS ;; ENABLE TTY KEYBOARD INTERRUPT
RTS PC ;; RETURN TO CALLER

:
: *TK SERVICE ROUTINE
: * THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
: * BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
: * IT IN THE QUEUE.
:
$TKSRV: MOVB $TKB,-(SP) ;; PICKUP THE CHARACTER
BIC #0177,(SP) ;; STRIP THE JUNK
1$: CMP (SP),#7 ;; IS IT A CONTROL G?
BNE 2$ ;; BRANCH IF NO
CMP $SWREG,$SWR ;; IS SOFT-SWR SELECTED?
BEQ 6$ ;; GO TO SWR CHANGE

2$:
CMP #1,$TKCNT ;; IS THE QUEUE FULL?
BNE 3$ ;; BRANCH IF NO
TYPE $BELL ;; RING THE TTY BELL
TST (SP)+ ;; CLEAN CHARACTER OFF OF STACK
BR 5$ ;; EXIT
3$: CMP (SP),#23 ;; IS IT A CONTROL-S?
BNE 32$ ;; BRANCH IF NO
CLR $TKS ;; DISABLE TTY KEYBOARD INTERRUPTS
TST (SP)+ ;; CLEAN CHAR OFF STACK
4$: TSTB $TKS ;; WAIT FOR A CHAR
BPL 31$ ;; LOOP UNTIL ITS THERE
MOVB $TKB,-(SP) ;; GET THE CHARACTER
BIC #0177,(SP) ;; MAKE IT 7-BIT ASCII
CMP (SP)+,#21 ;; IS IT A CONTROL-Q?
BNE 31$ ;; BRANCH IF NO
MOV $100,$TKS ;; REENABLE TTY KEYBOARD INTERRUPTS
RTI ;; RETURN
32$: INC $TKCNT ;; COUNT THIS CHARACTER
CMP (SP),#140 ;; IS IT UPPER CASE?
BLT 4$ ;; BRANCH IF YES
CMP (SP),#175 ;; IS IT A SPECIAL CHAR?
BGT 4$ ;; BRANCH IF YES
BIC #40,(SP) ;; MAKE IT UPPER CASE
4$: MOVB (SP)+,$TKQIN ;; AND PUT IT IN QUEUE
INC $TKQIN ;; UPDATE THE POINTER
CMP $TKQIN,$TKQEND ;; GO OFF THE END?
BNE 5$ ;; BRANCH IF NO
MOV $STKQSR,$TKQIN ;; RESET THE POINTER
5$: RTI ;; RETURN

```

```

2601
2602
2603
2604
2605
2606
2607 022770 022737 000176 001140 $CKSWR: CMP #SWREG,SWR ;; IS THE SOFT-SWR SELECTED
2608 022776 001104 BNE 15$ ;; EXIT IF NOT
2609 023000 105777 156140 TSTB $STKS ;; IS A CHAR WAITING?
2610 023004 100101 BPL 15$ ;; IF NOT, EXIT
2611 023006 117746 156134 MOVB $STKB,-(SP) ;; YES
2612 023012 042716 177600 BIC #1C177,(SP) ;; MAKE IT 7-BIT ASCII
2613 023016 021627 000007 CMP (SP),#7 ;; IS IT A CONTROL-G?
2614 023022 001300 BNE 2$ ;; IF NOT, PUT IT IN THE TTY QUEUE
2615
2616
2617
2618
2619
2620
2621 023024 123727 001134 000001 6$: CMPB $AUTOB,#1 ;; ARE WE RUNNING IN AUTO-MODE?
2622 023032 001674 BEQ 2$ ;; BRANCH IF YES
2623 023034 005726 TST (SP)+ ;; CLEAR CONTROL-G OFF STACK
2624 023036 004737 022526 JSR PC,$TKINT ;; FLUSH THE TTY INPUT QUEUE
2625 023042 005077 156076 CLR $STKS ;; DISABLE TTY KEYBOARD INTERRUPTS
2626 023046 112737 000001 001135 MOVB #1,$INTAG ;; SET INTERRUPT MODE INDICATOR
2627
2628 023054 104401 023657 $GTSWR: TYPE , $CNTLG ;; ECHO THE CONTROL-G (↑G)
2629 023060 104401 023664 TYPE $MSWR ;; TYPE CURRENT CONTENTS
2630 023064 013746 000176 MOV $SWREG,-(SP) ;; SAVE SWREG FOR TYPEOUT
2631 023070 104402 TYPCO ;; GO TYPE--OCTAL ASCII(ALL DIGITS)
2632 023072 104401 023675 TYPE , $MNEW ;; PROMPT FOR NEW SWR
2633 023076 005046 19$: CLR -(SP) ;; CLEAR COUNTER
2634 023100 005046 CLR -(SP) ;; THE NEW SWR
2635 023102 105777 156036 7$: TSTB $STKS ;; CHAR THERE?
2636 023106 100375 BPL 7$ ;; IF NOT TRY AGAIN
2637
2638 023110 117746 156032 MOVB $STKB,-(SP) ;; PICK UP CHAR
2639 023114 042716 177600 BIC #1C177,(SP) ;; MAKE IT 7-BIT ASCII
2640
2641
2642
2643 023120 021627 000025 9$: CMP (SP),#25 ;; IS IT A CONTROL-U?
2644 023124 001005 BNE 10$ ;; BRANCH IF NOT
2645 023126 104401 023652 TYPE , $CNTLU ;; YES, ECHO CONTROL-U (↑U)
2646 023132 062706 000006 20$: ADD #6,SP ;; IGNORE PREVIOUS INPUT
2647 023136 000757 BR 19$ ;; LET'S TRY IT AGAIN
2648
2649
2650 023140 021627 000015 10$: CMP (SP),#15 ;; IS IT A <CR>?
2651 023144 001022 BNE 16$ ;; BRANCH IF NO
2652 023146 005766 000004 TST 4(SP) ;; YES, IS IT THE FIRST CHAR?
2653 023152 001403 BEQ 11$ ;; BRANCH IF YES
2654 023154 016677 000002 155756 MOV 2(SP),$SWR ;; SAVE NEW SWR
2655 023162 062706 000006 11$: ADD #6,SP ;; CLEAR UP STACK
2656 023166 104401 001161 14$: TYPE , $CRLF ;; ECHO <CR> AND <LF>

```

```

2657 023172 123727 001135 000001      CMPB    $INTAG,#1      ;;RE-ENABLE TTY KBC INTERRUPTS
2658 023200 001003      BNE     15$           ;;BRANCH IF NOT
2659 023252 012777 000100 155734      MOV     #100,$STKS    ;;RE-ENABLE TTY KBC INTERRUPTS
2660 023270 000002      RTI                    ;;RETURN
2661 023212 004737 022220      JSR     PC,$TYPEC    ;;ECHO CHAR
2662 023216 021627 000060      CMP     (SP),#60      ;;CHAR < 0?
2663 023222 002420      BLT     18$           ;;BRANCH IF YES
2664 023224 021627 000067      CMP     (SP),#67      ;;CHAR > 7?
2665 023230 003015      BGT     18$           ;;BRANCH IF YES
2666 023232 042726 000060      BIC     #60,(SP)+     ;;STRIP-OFF ASCII
2667 023236 005766 000002      TST     2(SP)         ;;IS THIS THE FIRST CHAR
2668 023242 001403      BEQ     17$           ;;BRANCH IF YES
2669 023244 006316      ASL     (SP)          ;;NO, SHIFT PRESENT
2670 023246 006316      ASL     (SP)          ;;CHAR OVER TO MAKE
2671 023250 006316      ASL     (SP)          ;;ROOM FOR NEW ONE.
2672 023252 005266 000002      INC     2(SP)         ;;KEEP COUNT OF CHAR
2673 023256 056616 177776      BIS     -2(SP),(SP)   ;;SET IN NEW CHAR
2674 023262 000707      BR      7$           ;;GET THE NEXT ONE
2675 023264 104401 001160      18$:    TYPE    $QUES  ;;TYPE ?<CR><LF>
2676 023270 000720      BR      20$         ;;SIMULATE CONTROL-U
2677      .DSABL  LSB

```

```

2678
2679
2680
2681      ;;*****
2682      ;;THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
2683      ;;CALL:
2684      ;;      RDCHR      ;;GET A CHARACTER FROM THE QUEUE
2685      ;;      RETURN HERE ;;CHARACTER IS ON THE STACK
2686      ;;      ;;WITH PARITY BIT STRIPPED OFF
2687

```

```

2688 023272 011646 000004 000002 $RDCHR: MOV     (SP),-(SP)    ;;PUSH DOWN THE PC AND
2689 023274 016666 000004      MOV     4(SP),2(SP)  ;;THE PS
2690 023302 005066 000004      CLR     4(SP)        ;;GET READY FOR A CHARACTER
2691 023306 005046      CLR     -(SP)        ;;PUT NEW PS ON STACK
2692 023310 012746 023316      MOV     #64$,-(SP)   ;;PUT NEW PC ON STACK
2693 023314 000002      RTI                    ;;POP NEW PC AND PS
2694 023316
2695 023316 005737 022516      64$:   TST     $STKCNT   ;;WAIT ON A CHARACTER
2696 023322 001775      1$:    BEQ     1$           ;;
2697 023324 005337 022516      DEC     $STKCNT      ;;DECREMENT THE COUNTER
2698 023330 117766 177166 000004      MOVB   $STKQOUT,4(SP) ;;GET ONE CHARACTER
2699 023336 005237 022522      INC     $STKQOUT     ;;UPDATE THE POINTER
2700 023342 023727 022522 022525      CMP     $STKQOUT,$STKQEND ;;DID IT GO OFF OF THE END?
2701 023350 001003      BNE     2$           ;;BRANCH IF NO
2702 023352 012737 022524 022522      MOV     #$STQSRT,$STKQOUT ;;RESET THE POINTER
2703 023360 000002      2$:    RTI                    ;;RETURN

```

```

2704      ;;*****
2705      ;;THIS ROUTINE WILL INPUT A STRING FROM THE TTY
2706      ;;CALL:
2707      ;;      RDLIN     ;;INPUT A STRING FROM THE TTY
2708      ;;      RETURN HERE ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
2709      ;;      ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
2710
2711 023362 010346 $PDLIN: MOV     R3,-(SP)  ;;SAVE R3
2712 023364 005046      CLR     -(SP)        ;;CLEAR THE RUBOUT KEY

```

2713	023366	012703	023616	15:	MOV	#\$TTYIN,R3	:: GET ADDRESS
2714	023372	022703	023646	25:	CMP	#\$TTYIN+30,R3	:: BUFFER FULL?
2715	023376	101456			BLOS	45	:: BR IF YES
2716	023400	104407			RDCHR		:: GO READ ONE CHARACTER FROM THE TTY
2717	023402	112613			MOVW	(SP)+,R3	:: GET CHARACTER
2718	023404	122713	000177	105:	CMPB	#177,(R3)	:: IS IT A RUBOUT
2719	023410	001023			BNE	55	:: BR IF NO
2720	023412	005716			TST	(SP)	:: IS THIS THE FIRST RUBOUT?
2721	023414	001007			BNE	65	:: BR IF NO
2722	023416	112737	000134 023614		MOVW	#'\,95	:: TYPE A BACK SLASH
2723	023424	104401	023614		TYPE	95	
2724	023430	012716	177777		MOV	#-1,(SP)	:: SET THE RUBOUT KEY
2725	023434	005303		65:	DEC	R3	:: BACKUP BY ONE
2726	023436	020327	023616		CMP	R3, \$TTYIN	:: STACK EMPTY?
2727	023442	103434			BLO	45	:: BR IF YES
2728	023444	111337	023614		MOVW	(R3),95	:: SETUP TO TYPEOUT THE DELETED CHAR.
2729	023450	104401	023614		TYPE	95	:: GO TYPE
2730	023454	000746			BR	25	:: GO READ ANOTHER CHAR.
2731	023456	005716		55:	TST	(SP)	:: RUBOUT KEY SET?
2732	023460	001406			BEQ	75	:: BR IF NO
2733	023462	112737	000134 023614		MOVW	#'\,95	:: TYPE A BACK SLASH
2734	023470	104401	023614		TYPE	95	
2735	023474	005016			CLR	(SP)	:: CLEAR THE RUBOUT KEY
2736	023476	122713	000025	75:	CMPB	#25,(R3)	:: IS CHARACTER A CTRL U?
2737	023502	001003			BNE	85	:: BR IF NO
2738	023504	104401	023652		TYPE	\$CNTLU	:: TYPE A CONTROL "U"
2739	023510	000726			BR	15	:: GO START OVER
2740	023512	122713	000022	85:	CMPB	#22,(R3)	:: IS CHARACTER A "↑R"?
2741	023516	001011			BNE	35	:: BRANCH IF NO
2742	023520	105013			CLRB	(R3)	:: CLEAR THE CHARACTER
2743	023522	104401	001161		TYPE	.\$CRLF	:: TYPE A "CR" & "LF"
2744	023526	104401	023616		TYPE	.\$TTYIN	:: TYPE THE INPUT STRING
2745	023532	000717			BR	25	:: GO PICKUP ANOTHER CHARACTER
2746	023534	104401	001160	45:	TYPE	.\$QUES	:: TYPE A '?'
2747	023540	000712			BR	15	:: CLEAR THE BUFFER AND LOOP
2748	023542	111337	023614	35:	MOVW	(R3),95	:: ECHO THE CHARACTER
2749	023546	104401	023614		TYPE	95	
2750	023552	122723	000015		CMPB	#15,(R3)+	:: CHECK FOR RETURN
2751	023556	001305			BNE	25	:: LOOP IF NOT RETURN
2752	023560	105063	177777		CLRB	-1(R3)	:: CLEAR RETURN (THE 15)
2753	023564	104401	001162		TYPE	.\$LF	:: TYPE A LINE FEED
2754	023570	005726			TST	(SP)+	:: CLEAN RUBOUT KEY FROM THE STACK
2755	023572	012603			MOV	(SP)+,R3	:: RESTORE R3
2756	023574	011646			MOV	(SP)-,(SP)	:: ADJUST THE STACK AND PUT ADDRESS OF THE
2757	023576	016666	000004 000002		MOV	4(SP),2(SP)	:: FIRST ASCII CHARACTER ON IT
2758	023604	012766	023616 000004		MOV	#\$TTYIN,4(SP)	
2759	023612	000002			RTI		:: RETURN
2760	023614	000		95:	.BYTE	0	:: STORAGE FOR ASCII CHAR. TO TYPE
2761	023615	000			.BYTE	0	:: TERMINATOR
2762	023616	000030			.\$TTYIN:	.\$BLKB 30	:: RESERVE 30 BYTES FOR TTY INPUT
2763	023646	177607	000377		.\$BELL:	.\$ASCIZ <207><377><377>	:: CODE FOR BELL
2764	023652	052536	005015 000		.\$CNTLU:	.\$ASCIZ /↑U/<15><12>	:: CONTROL "U"
2765	023657	136	006507 000012		.\$CNTLG:	.\$ASCIZ /↑G/<15><12>	:: CONTROL "G"
2766	023664	005015	053523 020122		.\$MSWR:	.\$ASCIZ <15><12>/SWR = /	
2767	023672	020075	000				
2768	023675	040	047040 053505		.\$MNEW:	.\$ASCIZ / NEW = /	

2769 023702 036440 000040

.SBTTL READ AN OCTAL NUMBER FROM THE TTY

2770
2771
2772
2773
2774
2775
2776
2777
2778
2779
2780
2781
2782
2783
2784
2785
2786
2787
2788
2789
2790
2791
2792
2793
2794
2795
2796
2797
2798
2799
2800
2801
2802
2803
2804
2805
2806
2807
2808
2809
2810
2811
2812
2813
2814
2815
2816
2817
2818
2819
2820
2821
2822
2823
2824

011646
016666 000004 000002
010046
010146
010246
104410
012600
005001
005002
112046
001412
006301
006102
006301
006102
006301
006102
042716 177770
062601
000764
005726
010166 000012
010237 024006
012602
012601
012600
000002
000000

010046
016600 000002
005740
111000
006300
016000 024044
000200

*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
*CHANGE IT TO BINARY.
*CALL:
* RDOCT ;:READ AN OCTAL NUMBER
* RETURN HERE ;:LOW ORDER BITS ARE ON TOP OF THE STACK
* ;:HIGH ORDER BITS ARE IN \$HIOCT

\$RDOCT: MOV (SP),-(SP) ;:PROVIDE SPACE FOR THE
MOV 4(SP),2(SP) ;:INPUT NUMBER
MOV RC,-(SP) ;:PUSH R0 ON STACK
MOV R1,-(SP) ;:PUSH R1 ON STACK
MOV R2,-(SP) ;:PUSH R2 ON STACK
1\$: RDLIN ;:READ AN ASCII LINE
MOV (SP)+,R0 ;:GET ADDRESS OF 1ST CHARACTER
CLR R1 ;:CLEAR DATA WORD
CLR R2
2\$: MOVB (R0)+,-(SP) ;:PICKUP THIS CHARACTER
BEQ 3\$;:IF ZERO GET OUT
ASL R1 ;:*2
ROL R2 ;:*4
ASL R1 ;:*8
ROL R2
BIC #1C7,(SP) ;:STRIP THE ASCII JUNK
ADD (SP)+,R1 ;:ADD IN THIS DIGIT
BR 2\$;:LOOP
3\$: TST (SP)+ ;:CLEAN TERMINATOR FROM STACK
MOV R1,12(SP) ;:SAVE THE RESULT
MOV R2,\$HIOCT
MOV (SP)+,R2 ;:POP STACK INTO R2
MOV (SP)+,R1 ;:POP STACK INTO R1
MOV (SP)+,R0 ;:POP STACK INTO R0
RTI ;:RETURN
\$HIOCT: .WORD 0 ;:HIGH ORDER BITS GO HERE
.SBTTL TRAP DECODER

*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
*GO TO THAT ROUTINE.

\$TRAP: MOV R0,-(SP) ;:SAVE R0
MOV 2(SP),R0 ;:GET TRAP ADDRESS
TST -(R0) ;:BACKUP BY 2
MOVB (R0),R0 ;:GET RIGHT BYTE OF TRAP
ASL R0 ;:POSITION FOR INDEXING
MOV \$TRPAD(R0),R0 ;:INDEX TO TABLE
RTS R0 ;:GO TO ROUTINE

2825
2826
2827
2828
2829
2830
2831
2832
2833
2834
2835
2836
2837
2838
2839
2840
2841
2842
2843
2844
2845
2846
2847
2848
2849
2850
2851
2852
2853
2854
2855
2856
2857
2858
2859
2860
2861
2862
2863
2864
2865
2866
2867
2868
2869
2870
2871
2872
2873
2874
2875
2876
2877
2878
2879
2880

024032 011646
024034 016666 000004 000002
024042 000002

024044 024032
024046 022050
024050 022314
024052 022270
024054 022330

024056 023060

024060 022770
024062 023272
024064 023362
024066 023706

024070 012737 024230 000024
024076 012737 000340 000026
024104 010046
024106 010146
024110 010246
024112 010346
024114 010446
024116 010546
024120 017746 155014
024124 010637 024234
024130 012737 024142 000024
024136 000000
024140 000776

024142 012737 024230 000024
024150 013706 024234
024154 005037 024234
024160 005237 024234
024164 001375
024166 012677 154746
024172 012605
024174 012604
024176 012603
024200 012602
024202 012601

```
:::THIS IS USE TO HANDLE THE "GETPRI" MACRO
$TRAP2: MOV (SP),-(SP) ;;MOVE THE PC DOWN
MOV 4(SP),2(SP) ;;MOVE THE PSW DOWN
RTI ;;RESTORE THE PSW

.SBTTL TRAP TABLE
:::THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
:::BY THE "TRAP" INSTRUCTION.

: ROUTINE
:-----
$TRPAD: .WORD $TRAP2
$TYPE ;;CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE
$TYPOC ;;CALL=TYPOC TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
$TYPOS ;;CALL=TYPOS TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
$TYPON ;;CALL=TYPON TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)

$GTSWR ;;CALL=GTSWR TRAP+5(104405) GET SOFT-SWR SETTING

$CKSWR ;;CALL=CKSWR TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR
$RDCHR ;;CALL=RDCHR TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
$RDLIN ;;CALL=RDLIN TRAP+10(104410) TTY TYPEIN STRING ROUTINE
$RDOCT ;;CALL=RDOCT TRAP+11(104411) READ AN OCTAL NUMBER FROM TTY

.SBTTL POWER DOWN AND UP ROUTINES

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

:::*****
:POWER UP ROUTINE
$PWRUP: MOV $SILLUP,@PWRVEC ;;SET FOR FAST DOWN
MOV $SAVR6,SP ;;GET SP
CLR $SAVR6 ;;WAIT LOOP FOR THE TTY
1$: INC $SAVR6 ;;WAIT FOR THE INC
BNE 1$ ;;OF WORD
MOV (SP)+,@SWR ;;POP STACK INTO @SWR
MOV (SP)+,R5 ;;POP STACK INTO R5
MOV (SP)+,R4 ;;POP STACK INTO R4
MOV (SP)+,R3 ;;POP STACK INTO R3
MOV (SP)+,R2 ;;POP STACK INTO R2
MOV (SP)+,R1 ;;POP STACK INTO R1
```

```

2881 024204 012600      MOV      (SP)+,R0      ::POP STACK INTO R0
2882 024206 012737 024370 000024      MOV      @SPWRDN,@PWRVEC  ::SET UP THE POWER DOWN VECTOR
2883 024214 012737 000340 000026      MOV      @340,@PWRVEC+2  ::PRIO:7
2884 024222 104401      TYPE      ::REPORT THE POWER FAILURE
2885 024224 024236      SPWRMG: .WORD  $POWER  ::POWER FAIL MESSAGE PCINTER
2886 024226 000002      RTI
2887 024230 000000      $ILLUP: HALT      ::THE POWER UP SEQUENCE WAS STARTED
2888 024232 000776      BR      .-2      ::BEFORE THE POWER DOWN WAS COMPLETE
2889 024234 000000      $SAVR6: 0
2890 024236 005015 047520 042527      $POWER: .ASCIZ <15><12>"POWER"  ::PUT THE SP HERE
2891 024244 000122
2892 000001      .EVEN
2893      .END

```

AUTSL2	002624	CYCLE	004154	EXECJT	005362	PR01	003266	SECCNT	001344
BA	001406	CYCLE2	004200	EXIT	003346	PR02	003140	SECCND	003432
BADINP	003254	CYCL2	004170	EXITA	003634	PRTTWO	014600	SECCNE	006732
BADONE	010340	CYL	013524	EXITB	003642	PRC	= 000000	SECSYS	003030
BAD.IN	011752	CYLADR	005262	EXITX	003410	PR1	= 000040	SECTBL	001305
BAD.ON	013734	CYLAD2	005256	EXIT2	006056	PR2	= 000100	SECT.0	013744
BASE	001266	CYLBAS	001315	EXITFR2	003352	PR3	= 000140	SECT.1	011762
BASINC	005340	CYLCNT	001342	EXTR	001420	PR4	= 000200	SECT.2	010350
BEGCT	017276	CYLOFF	005272	FAIL	013276	PR5	= 000240	SECT.3	002622
BEGCT1	017304	CYLTBL	001274	FAILED	013260	PR6	= 000300	SECT.4	017154
BEGIN	002602	DA	001410	FFLAG	021364	PR7	= 000340	SECT.5	020510
BIT0	= 000001	DDISP	= 177570	FIL.ON	003652	PS	= 177776	SECT.6	021402
BIT00	= 000001	DISPLA	001142	GDI	012506	PSW	= 177776	SEEKI	001402
BIT01	= 000002	DISPRE	000174	GO	003272	PWRVEC	= 000024	SEEKO	001404
BIT02	= 000004	DNIDRV	020226	GOOT	012462	RBA	001414	SEKSET	011550
BIT03	= 000010	DO2	004230	GO1	003304	RBUFF	013654	SHFCNT	020320
BIT04	= 000020	DPL	= 010000	GO2	003316	RDBUFF	007336	SHF1	013570
BIT05	= 000040	DRACTV	001164	GO3	003324	RDCHK	006430	SHF2	013566
BIT06	= 000100	DRCNT1	001311	GTSWR	= 104405	RDCHKO	016674	SHF2	013564
BIT07	= 000200	DRIVE	001314	HDALGN	020752	RDCHR	= 104407	SIN	= 001000
BIT08	= 000400	DRIVC	020470	HDRFLG	001320	RDLIN	= 104410	SINST	020054
BIT09	= 001000	DRIV1	020472	HT	= 000011	RDLINK	005612	SMTME	005574
BIT1	= 000002	DRIV2	020474	IDEX	001324	RDOCT	= 104411	STACK	= 001100
BIT10	= 002000	DRIV3	020476	ILEGAL	004122	RDTBL	001226	START	001440
BIT11	= 004000	DRIV4	020500	INITIL	004504	RD1	005636	STARTR	001434
BIT12	= 010000	DRIV5	020502	INITI2	004516	RD2	005650	STFLG	001325
BIT13	= 020000	DRIV6	020504	INIT.2	010446	RD3	005656	STKLMT	= 177774
BIT14	= 040000	DRIV7	020506	INSYS2	003506	RD4	005720	STOP	013556
BIT15	= 100000	DRVCNT	020322	IO	012512	RD5	005726	STORE	004052
BIT2	= 000004	DRVO	001206	IOTVEC	= 000020	RD6	005732	STRT1	001754
BIT3	= 000010	DSKADR	001334	KYTEMP	001330	RD7	006032	SUR	013550
BIT4	= 000020	DSKTMP	001352	LDFLG	004204	READCS	001356	SWR	001140
BIT5	= 000040	DSWR	= 177570	LDSEEK	011564	RESTRT	006212	SWREG	000176
BIT6	= 000100	ECNT	001321	LF	= 000012	RESVEC	= 000010	SWTCH	021366
BIT7	= 000200	EMTVEC	= 000030	LFLF	= 105212	RETRF2	007250	SW0	= 000001
BIT8	= 000400	EM1	020324	LOGA	001166	RETRN2	006426	SW00	= 000001
BIT9	= 001000	EM10	021370	MANSEL	010336	RETRN3	005354	SW01	= 000002
BPTVEC	= 000014	EM2	020335	MASK	005214	RETRN4	004016	SW02	= 000004
BUFF	013652	EM3	020347	MASKER	004010	RETRY	013166	SW03	= 000010
BUFF	022046	EM4	020362	MODE	001312	RFCERR	013234	SW04	= 000020
BUSADR	001336	EM5	020375	MORE	013040	RFERR	013216	SW05	= 000040
CHECK1	005430	EM6	020413	MOUNT	004276	RKBA	001374	SW06	= 000100
CHKCN	001350	EM7	020434	MSKTBL	001256	RKCS	001370	SW07	= 000200
CKSWR	= 104406	EM8	020447	NEXT	012376	RKDA	001376	SW08	= 000400
CLRDPL	020212	EM9	020455	NG	002574	RKDS	001364	SW09	= 001000
CNTSIN	001322	ERRCHK	006062	NXT1	017460	RKER	001366	SW1	= 000002
COM	012300	ERRFLG	001360	NXT2	017554	RKWC	001372	SW10	= 002000
COMMON	012274	ERRRF	001424	OSPFLG	001326	RWC	001416	SW11	= 004000
COMMND	001316	ERRRFC	001426	PASTBL	001246	RWS	= 000100	SW12	= 010000
CONTIN	011364	ERRVEC	= 000004	PATTRN	001362	RWSRDY	011336	SW13	= 020000
CONTRL	001332	ERRWCH	001430	PIRQ	= 177772	R6	= %000006	SW14	= 040000
CR	= 000015	ERRWCS	001432	PIRQVE	= 000240	R7	= %000007	SW15	= 100000
CRLF	= 000200	ERRWF	001422	PRCNUM	001313	SEC	013536	SW2	= 000004

012 0001
013 0002
014 0003
015 0004
016 0005
017 0006
018 0007
019 0008
020 0009
021 0010
022 0011
023 0012
024 0013
025 0014
026 0015
027 0016
028 0017
029 0018
030 0019
031 0020
032 0021
033 0022
034 0023
035 0024
036 0025
037 0026
038 0027
039 0028
040 0029
041 0030
042 0031
043 0032
044 0033
045 0034
046 0035
047 0036
048 0037
049 0038
050 0039
051 0040
052 0041
053 0042
054 0043
055 0044
056 0045
057 0046
058 0047
059 0048
060 0049
061 0050
062 0051
063 0052
064 0053
065 0054
066 0055
067 0056
068 0057
069 0058
070 0059
071 0060
072 0061
073 0062
074 0063
075 0064
076 0065
077 0066
078 0067
079 0068
080 0069
081 0070
082 0071
083 0072
084 0073
085 0074
086 0075
087 0076
088 0077
089 0078
090 0079
091 0080
092 0081
093 0082
094 0083
095 0084
096 0085
097 0086
098 0087
099 0088
100 0089
101 0090
102 0091
103 0092
104 0093
105 0094
106 0095
107 0096
108 0097
109 0098
110 0099
111 0100
112 0101
113 0102
114 0103
115 0104
116 0105
117 0106
118 0107
119 0108
120 0109
121 0110
122 0111
123 0112
124 0113
125 0114
126 0115
127 0116
128 0117
129 0118
130 0119
131 0120
132 0121
133 0122
134 0123
135 0124
136 0125
137 0126
138 0127
139 0128
140 0129
141 0130
142 0131
143 0132
144 0133
145 0134
146 0135
147 0136
148 0137
149 0138
150 0139
151 0140
152 0141
153 0142
154 0143
155 0144
156 0145
157 0146
158 0147
159 0148
160 0149
161 0150
162 0151
163 0152
164 0153
165 0154
166 0155
167 0156
168 0157
169 0158
170 0159
171 0160
172 0161
173 0162
174 0163
175 0164
176 0165
177 0166
178 0167
179 0168
180 0169
181 0170
182 0171
183 0172
184 0173
185 0174
186 0175
187 0176
188 0177
189 0178
190 0179
191 0180
192 0181
193 0182
194 0183
195 0184
196 0185
197 0186
198 0187
199 0188
200 0189
201 0190
202 0191
203 0192
204 0193
205 0194
206 0195
207 0196
208 0197
209 0198
210 0199
211 0200
212 0201
213 0202
214 0203
215 0204
216 0205
217 0206
218 0207
219 0208
220 0209
221 0210
222 0211
223 0212
224 0213
225 0214
226 0215
227 0216
228 0217
229 0218
230 0219
231 0220
232 0221
233 0222
234 0223
235 0224
236 0225
237 0226
238 0227
239 0228
240 0229
241 0230
242 0231
243 0232
244 0233
245 0234
246 0235
247 0236
248 0237
249 0238
250 0239
251 0240
252 0241
253 0242
254 0243
255 0244
256 0245
257 0246
258 0247
259 0248
260 0249
261 0250
262 0251
263 0252
264 0253
265 0254
266 0255
267 0256
268 0257
269 0258
270 0259
271 0260
272 0261
273 0262
274 0263
275 0264
276 0265
277 0266
278 0267
279 0268
280 0269
281 0270
282 0271
283 0272
284 0273
285 0274
286 0275
287 0276
288 0277
289 0278
290 0279
291 0280
292 0281
293 0282
294 0283
295 0284
296 0285
297 0286
298 0287
299 0288
300 0289
301 0290
302 0291
303 0292
304 0293
305 0294
306 0295
307 0296
308 0297
309 0298
310 0299
311 0300
312 0301
313 0302
314 0303
315 0304
316 0305
317 0306
318 0307
319 0308
320 0309
321 0310
322 0311
323 0312
324 0313
325 0314
326 0315
327 0316
328 0317
329 0318
330 0319
331 0320
332 0321
333 0322
334 0323
335 0324
336 0325
337 0326
338 0327
339 0328
340 0329
341 0330
342 0331
343 0332
344 0333
345 0334
346 0335
347 0336
348 0337
349 0338
350 0339
351 0340
352 0341
353 0342
354 0343
355 0344
356 0345
357 0346
358 0347
359 0348
360 0349
361 0350
362 0351
363 0352
364 0353
365 0354
366 0355
367 0356
368 0357
369 0358
370 0359
371 0360
372 0361
373 0362
374 0363
375 0364
376 0365
377 0366
378 0367
379 0368
380 0369
381 0370
382 0371
383 0372
384 0373
385 0374
386 0375
387 0376
388 0377
389 0378
390 0379
391 0380
392 0381
393 0382
394 0383
395 0384
396 0385
397 0386
398 0387
399 0388
400 0389
401 0390
402 0391
403 0392
404 0393
405 0394
406 0395
407 0396
408 0397
409 0398
410 0399
411 0400
412 0401
413 0402
414 0403
415 0404
416 0405
417 0406
418 0407
419 0408
420 0409
421 0410
422 0411
423 0412
424 0413
425 0414
426 0415
427 0416
428 0417
429 0418
430 0419
431 0420
432 0421
433 0422
434 0423
435 0424
436 0425
437 0426
438 0427
439 0428
440 0429
441 0430
442 0431
443 0432
444 0433
445 0434
446 0435
447 0436
448 0437
449 0438
450 0439
451 0440
452 0441
453 0442
454 0443
455 0444
456 0445
457 0446
458 0447
459 0448
460 0449
461 0450
462 0451
463 0452
464 0453
465 0454
466 0455
467 0456
468 0457
469 0458
470 0459
471 0460
472 0461
473 0462
474 0463
475 0464
476 0465
477 0466
478 0467
479 0468
480 0469
481 0470
482 0471
483 0472
484 0473
485 0474
486 0475
487 0476
488 0477
489 0478
490 0479
491 0480
492 0481
493 0482
494 0483
495 0484
496 0485
497 0486
498 0487
499 0488
500 0489
501 0490
502 0491
503 0492
504 0493
505 0494
506 0495
507 0496
508 0497
509 0498
510 0499
511 0500
512 0501
513 0502
514 0503
515 0504
516 0505
517 0506
518 0507
519 0508
520 0509
521 0510
522 0511
523 0512
524 0513
525 0514
526 0515
527 0516
528 0517
529 0518
530 0519
531 0520
532 0521
533 0522
534 0523
535 0524
536 0525
537 0526
538 0527
539 0528
540 0529
541 0530
542 0531
543 0532
544 0533
545 0534
546 0535
547 0536
548 0537
549 0538
550 0539
551 0540
552 0541
553 0542
554 0543
555 0544
556 0545
557 0546
558 0547
559 0548
560 0549
561 0550
562 0551
563 0552
564 0553
565 0554
566 0555
567 0556
568 0557
569 0558
570 0559
571 0560
572 0561
573 0562
574 0563
575 0564
576 0565
577 0566
578 0567
579 0568
580 0569
581 0570
582 0571
583 0572
584 0573
585 0574
586 0575
587 0576
588 0577
589 0578
590 0579
591 0580
592 0581
593 0582
594 0583
595 0584
596 0585
597 0586
598 0587
599 0588
600 0589
601 0590
602 0591
603 0592
604 0593
605 0594
606 0595
607 0596
608 0597
609 0598
610 0599
611 0600
612 0601
613 0602
614 0603
615 0604
616 0605
617 0606
618 0607
619 0608
620 0609
621 0610
622 0611
623 0612
624 0613
625 0614
626 0615
627 0616
628 0617
629 0618
630 0619
631 0620
632 0621
633 0622
634 0623
635 0624
636 0625
637 0626
638 0627
639 0628
640 0629
641 0630
642 0631
643 0632
644 0633
645 0634
646 0635
647 0636
648 0637
649 0638
650 0639
651 0640
652 0641
653 0642
654 0643
655 0644
656 0645
657 0646
658 0647
659 0648
660 0649
661 0650
662 0651
663 0652
664 0653
665 0654
666 0655
667 0656
668 0657
669 0658
670 0659
671 0660
672 0661
673 0662
674 0663
675 0664
676 0665
677 0666
678 0667
679 0668
680 0669
681 0670
682 0671
683 0672
684 0673
685 0674
686 0675
687 0676
688 0677
689 0678
690 0679
691 0680
692 0681
693 0682
694 0683
695 0684
696 0685
697 0686
698 0687
699 0688
700 0689
701 0690
702 0691
703 0692
704 0693
705 0694
706 0695
707 0696
708 0697
709 0698
710 0699
711 0700
712 0701
713 0702
714 0703
715 0704
716 0705
717 0706
718 0707
719 0708
720 0709
721 0710
722 0711
723 0712
724 0713
725 0714
726 0715
727 0716
728 0717
729 0718
730 0719
731 0720
732 0721
733 0722
734 0723
735 0724
736 0725
737 0726
738 0727
739 0728
740 0729
741 0730
742 0731
743 0732
744 0733
745 0734
746 0735
747 0736
748 0737
749 0738
750 0739
751 0740
752 0741
753 0742
754 0743
755 0744
756 0745
757 0746
758 0747
759 0748
760 0749
761 0750
762 0751
763 0752
764 0753
765 0754
766 0755
767 0756
768 0757
769 0758
770 0759
771 0760
772 0761
773 0762
774 0763
775 0764
776 0765
777 0766
778 0767
779 0768
780 0769
781 0770
782 0771
783 0772
784 0773
785 0774
786 0775
787 0776
788 0777
789 0778
790 0779
791 0780
792 0781
793 0782
794 0783
795 0784
796 0785
797 0786
798 0787
799 0788
800 0789
801 0790
802 0791
803 0792
804 0793
805 0794
806 0795
807 0796
808 0797
809 0798
810 0799
811 0800
812 0801
813 0802
814 0803
815 0804
816 0805
817 0806
818 0807
819 0808
820 0809
821 0810
822 0811
823 0812
824 0813
825 0814
826 0815
827 0816
828 0817
829 0818
830 0819
831 0820
832 0821
833 0822
834 0823
835 0824
836 0825
837 0826
838 0827
839 0828
840 0829
841 0830
842 0831
843 0832
844 0833
845 0834
846 0835
847 0836
848 0837
849 0838
850 0839
851 0840
852 0841
853 0842
854 0843
855 0844
856 0845
857 0846
858 0847
859 0848
860 0849
861 0850
862 0851
863 0852
864 0853
865 0854
866 0855
867 0856
868 0857
869 0858
870 0859
871 0860
872 0861
873 0862
874 0863
875 0864
876 0865
877 0866
878 0867
879 0868
880 0869
881 0870
882 0871
883 0872
884 0873
885 0874
886 0875
887 0876
888 0877
889 0878
890 0879
891 0880
892 0881
893 0882
894 0883
895 0884
896 0885
897 0886
898 0887
899 0888
900 0889
901 0890
902 0891
903 0892
904 0893
905 0894
906 0895
907 0896
908 0897
909 0898
910 0899
911 0900
912 0901
913 0902
914 0903
915 0904
916 0905
917 0906
918 0907
919 0908
920 0909
921 0910
922 0911
923 0912
924 0913
925 0914
926 0915
927 0916
928 0917
929 0918
930 0919
931 0920
932 0921
933 0922
934 0923
935 0924
936 0925
937 0926
938 0927
939 0928
940 0929
941 0930
942 0931
943 0932
944 0933
945 0934
946 0935
947 0936
948 0937
949 0938
950 0939
951 0940
952 0941
953 0942
954 0943
955 0944
956 0945
957 0946
958 0947
959 0948
960 0949
961 0950
962 0951
963 0952
964 0953
965 0954
966 0955
967 0956
968 0957
969 0958
970 0959
971 0960
972 0961
973 0962
974 0963
975 0964
976 0965
977 0966
978 0967
979 0968
980 0969
981 0970
982 0971
983 0972
984 0973
985 0974
986 0975
987 0976
988 0977
989 0978
990 0979
991 0980
992 0981
993 0982
994 0983
995 0984
996 0985
997 0986
998 0987
999 0988
1000 0989
1001 0990
1002 0991
1003 0992
1004 0993
1005 0994
1006 0995
1007 0996
1008 0997
1009 0998
1010 0999
1011 1000
1012 1001
1013 1002
1014 1003
1015 1004
1016 1005
1017 1006
1018 1007
1019 1008
1020 1009
1021 1010
1022 1011
1023 1012
1024 1013
1025 1014
1026 1015
1027 1016
1028 1017
1029 1018
1030 1019
1031 1020
1032 1021
1033 1022
1034 1023
1035 1024
1036 1025
1037 1026
1038 1027
1039 1028
1040 1029
1041 1030
1042 1031
1043 1032
1044 1033
1045 1034
1046 1035
1047 1036
1048 1037
1049 1038
1050 1039
1051 1040
1052 1041
1053 1042
1054 1043
1055 1044
1056 1045
1057 1046
1058 1047
1059 1048
1060 1049
1061 1050
1062 1051
1063 1052
1064 1053
1065 1054
1066 1055
1067 1056
1068 1057
1069 1058
1070 1059
1071 1060
1072 1061
1073 1062
1074 1063
1075 1064
1076 1065
1077 1066
1078 1067
1079 1068
1080 1069
1081 1070
1082 1071
1083 1072
1084 1073
1085 1074
1086 1075
1087 1076
1088 1077
1089 1078
1090 1079
1091 1080
1092 1081
1093 1082
1094 1083
1095 1084
1096 1085
1097 1086
1098 1087
1099 1088
1100 1089
1101 1090
1102 1091
1103 1092
1104 1093
1105 1094
1106 1095
1107 1096
1108 1097
1109 1098
1110 1099
1111 1100
1112 1101
1113 1102
1114 1103
1115 1104
1116 1105
1117 1106
1118 1107
1119 1108
1120 1109
1121 1110
1122 1111
1123 1112
1124 1113
1125 1114
1126 1115
1127 1116
1128 1117
1129 1118
1130 1119
1131 1120
1132 1121
1133 1122
1134 1123
1135 1124
1136 1125
1137 1126
1138 1127
1139 1128
1140 1129
1141 1130
1142 1131
1143 1132
1144 1133
1145 1134
1146 1135
1147 1136
1148 1137
1149 1138
1150 1139
1151 1140
1152 1141
1153 1142
1154 1143
1155 1144
1156 1145
1157 1146
1158 1147
1159 1148
1160 1149
1161 1150
1162 1151
1163 1152
1164 1153
1165 1154
1166 1155
1167 1156
1168 1157
1169 1158
1170 1159
1171 1160
1172 1161
1173 1162
1174 1163
1175 1164
1176 1165
1177 1166
1178 1167
1179 1168
1180 1169
1181 1170
1182 1171
1183 1172
1184 1173
1185 1174
1186 1175
1187 1176
1188 1177
1189 1178
1190 1179
1191 1180
1192 1181
1193 1182
1194 1183
1195 1184
1196 1185
1197 1186
1198 1187
1199 1188
1200 1189
1201 1190
1202 1191
1203 1192
1204 1193
1205 1194
1206 1195
1207 1196
1208 1197
1209 1198

C07

Supplier runtime 12 Seconds, 55 KCS, 331 disk reads, 4 disk writes, 79 pages

Report generated on 04-09-76 12:20:29 Monitor IPC-0 602 (670) #END#

000000001111111111222222222233333333444444445555555566666666777777778888888899999999
000000001111111111222222222233333333444444445555555566666666777777778888888899999999