RP11E

FUNCT LOGIC + READ/WRITE

MD-11-DZRPY-C

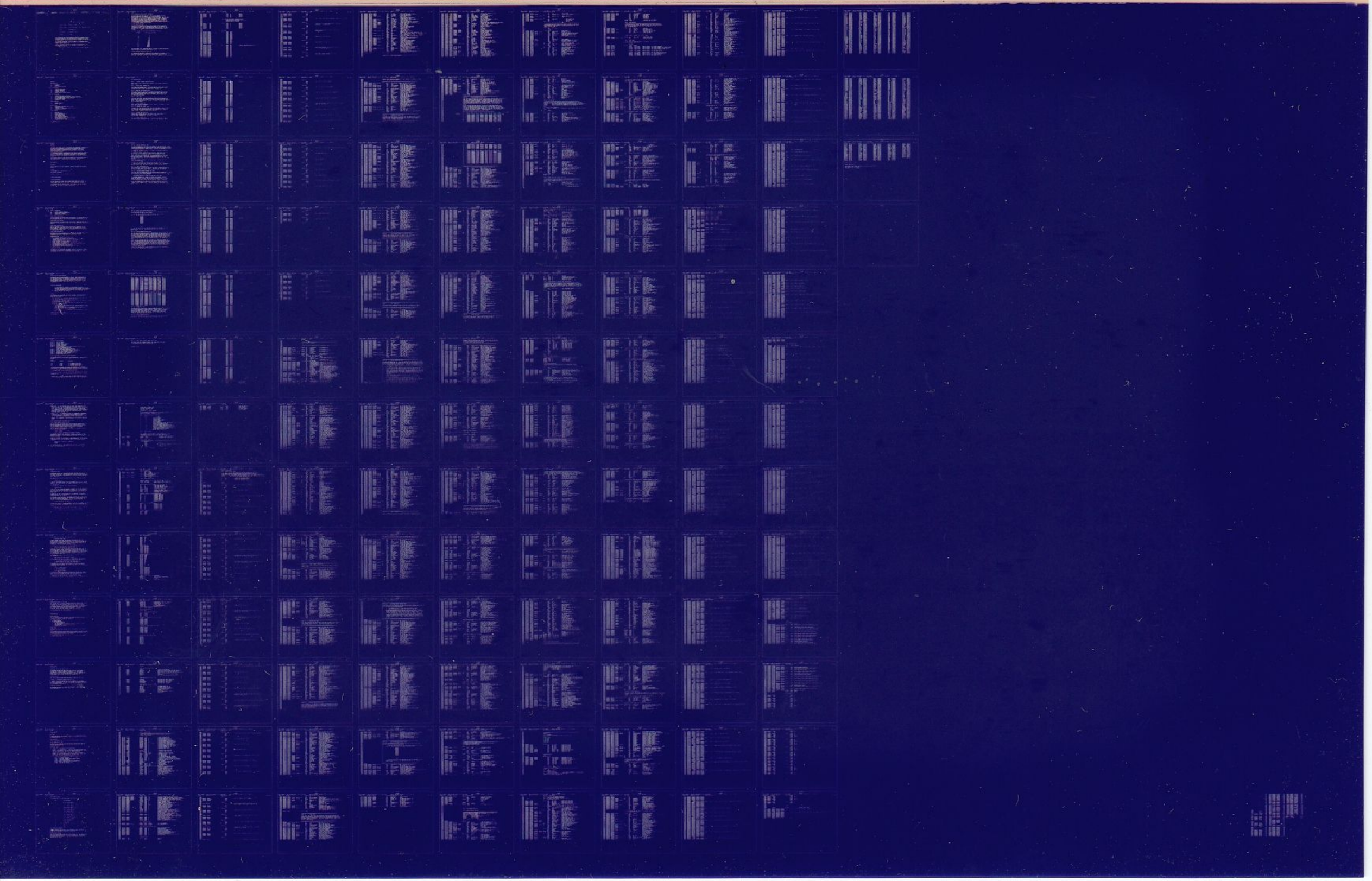
EP-DZRPY-C-DL-C

FICHE 1 OF 1

DEC 1977

digital

MADE IN USA



.REM %

I D E N T I F I C A T I O N

PRODUCT CODE: MAINDEC-11-DZRPY-C-D

PRODUCT NAME: RP11E FUNCTIONAL LOGIC & READ/WRITE TEST

DATE: SEPTEMBER 1977

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: C. HESS

CONTENTS

1. ABSTRACT

THE RP11E FUNCTIONAL LOGIC & READ/WRITE TEST PERFORMS A CURSORY CHECK OF THE RP11E CONTROLLER LOGIC AND PERFORMS DATA STORAGE/RETRIEVAL TESTING OF THE SUBSYSTEM. INCLUDED IN THE PROGRAM ARE UTILITY TESTS WHICH ASSIST IN PERFORMING HEAD ALIGNMENT, WHICH CHECK SUBSYSTEM POWER FAIL OPERATION, AND WHICH CHECK THE RP11E AND DRIVE CONTROL SWITCHES.

THE PROGRAM CONTAINS A 'CONVERSATION MODE' WHICH ALLOWS THE OPERATOR TO MODIFY THE OPERATION PARAMETERS FOR THE DATA STORAGE/RETREIVAL TEST. THE PROGRAM WILL USE MEMORY MANAGEMENT IN THE OPERATION OF THE DATA TEST IF THE PROGRAM IS BEING RUN ON A MEMORY MANAGEMENT SYSTEM.

THIS PROGRAM COMPLETES THE TESTING STARTED IN THE DISKLESS CONTROLLER TEST (MD-11-DZRPW). TESTING IN THE FUNCTIONAL LOGIC PROGRAM IS PERFORMED WITH 1 - 8 RP02, RPR02, OR RP03 DISK DRIVES.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11 COMPUTER WITH 12K OF MEMORY; CONSOLE TELETYPE; RP11E DISK CONTROLLER; 1 TO 8 RP02, RPR02, OR RP03 DISK DRIVES WITH FORMATTED PACKS.

2.2 STORAGE

THIS PROGRAM WILL LOAD AND RUN IN 12K.

2.3 PRELIMINARY PROGRAMS

MAINDEC-11-DZRPW - RP11E DISKLESS LOGIC TEST

3. STARTING PROCEDURES

3.1 LOADING PROCEDURES

THE PROGRAM MAY BE LOADED FROM EITHER PAPER TAPE OR AN 'XXDP' DEVICE. REFER TO EITHER THE STANDARD PROCEDURES FOR LOADING 'ABS' FORMAT PAPER TAPES OR TO THE 'XXDP' SYSTEM REFERENCE DOCUMENT.

THIS PROGRAM MAY BE INCLUDED IN AN 'XXDP' CHAIN. IF THE PROGRAM IS PART OF A CHAIN AND IS LOADED FROM AN RP02, RPR02, OR AN RP03, THE PROGRAM WILL NOT USE DRIVE 0 FOR TESTING.

3.2 STARTING ADDRESSES

200 NORMAL STARTING ADDRESS
204 SELECT OPERATING PARAMETERS
210 SELECT RP11 ADDRESS
214 COMBINATION OF 204 AND 210

3.2.1 START FROM LOCATION 200

WHEN THE PROGRAM IS STARTED FROM LOCATION 200, ALL ONLINE DRIVES WILL BE USED (SEE SECTION 3.1 FOR DRIVE 0 EXCEPTION) USING TESTS 0 - 14. ALL PARAMETERS FOR TEST 14 WILL BE RETURNED TO THEIR DEFAULT VALUES.

3.2.2 START FROM LOCATION 204

WHEN THE PROGRAM IS STARTED FROM LOCATION 204, PRAMETERS FOR TEST 14 ARE RESET TO THEIR DEFAULT VALUES, AND THE PROGRAM ENTERS CONVERSATIONAL MODE.

3.2.3 START FROM LOCATION 210

WHEN THE PROGRAM IS STARTED FROM LOCATION 210, OPERATION IS THE SAME AS THE START FROM 200, EXCEPT THAT THE PROGRAM ASKS FOR AN RP11E ADDRESS. THE OPERATOR MAY ENTER A NEW ADDRESS VALUE OR HE MAY RETAIN THE CURRENT VALUE. THE PARAMETERS FOR TEST 14 ARE RESET TO THEIR DEFAULT VALUES.

3.2.4 START FROM LOCATION 214

PROGRAM START FROM LOCATION 214 IS THE SAME AS THE START FROM LOCATION 204 EXCEPT THAT THE PROGRAM ASKS FOR A NEW RP11E ADDRESS. THE PARAMETERS FOR TEST 14 ARE RESET TO THEIR DEFAULT VALUES.

3.3 OPERATOR ACTION
BEFOR STARTING REFER TO SECTION 4.1

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3.1)
2. CHANGE TELETYPE KEYBOARD AND PRINTER ADDRESSES AS REQUIRED (SEE SECTION 4.2)
3. PLACE A FORMATTED PACK ON DRIVE(S) TO BE USED
4. BRING DRIVE(S) TO ONLINE STATE.
5. PLACE 'FORMAT' AND 'MAINTENANCE' SWITCHES ON THE RP11E CONTROL PANEL TO 'NORMAL'
6. SET CONTROLLER AND DRIVE WRITE LOCKOUT SWITCHES AS REQUIRED
7. LOAD ADDRESS 200, 204, 210, OR 214.
8. SET CPU SWITCH REGISTER SWITCHES (SEE SECTION 4.1)
9. PRESS START ON THE CPU CONTROL PANEL

3.4 'CONTROL C' OPTION

THE OPERATOR MAY TERMINATE THE CURRENT TEST AT ANY TIME BY TYPING A 'CONTROL C' ON THE TELETYPE KEYBOARD. TYPING A 'CONTROL C' CAUSES THE PROGRAM TO ENTER THE 'CONVERSATION MODE' ROUTINE. ALL THE PARAMETERS FOR TEST 14 WILL BE RETURNED TO THEIR DEFAULT VALUES.

3.5 DEFAULT RP11E UNIBUS ADDRESS

THE DEFAULT RP11E UNIBUS ADDRESS IS 176710. WHEN THE PROGRAM IS STARTED FROM EITHER LOCATION 200 OR LOCATION 204, ADDRESS 176710 WILL BE CHECKED AND IF NO RESPONSE IS RECEIVED FROM THAT ADDRESS THE PROGRAM WILL ENTER THE RP11E ADDRESS CHANGE ROUTINE. ADDRESS CHANGE OPERATION WILL BE IDENTICAL TO THE OPERATION WHEN THE PROGRAM IS STARTED FROM EITHER LOCATION 210 OR 214. (REFER TO SECTION 4.4)

4. OPERATING PROCEDURES

IF THE DIAGNOSTIC IS RUN ON A CPU WITHOUT A SWITCH REGISTER THEN A SOFTWARE SWITCH REGISTER IS USED WHICH ALLOWS THE USER THE SAME SWITCH OPTIONS AS THE HARDWARE SWITCH REGISTER. IF THE HARDWARE SWITCH REGISTER DOES NOT EXIST OR IF ONE DOES AND IT CONTAINS ALL ONES (177777) THEN THE SOFTWARE SWITCH REGISTER (LOC. 176) IS USED.

CONTROL:

THIS PROGRAM ALSO SUPPORTS THE DYNAMIC LOADING OF THE SOFTWARE SWITCH REGISTER (LOC. 176) FROM THE TTY. THIS CAN BE ACCOMPLISHED BY DOING THE FOLLOWING:

1) TYPE CONTROL G (↑G); THIS WILL ALLOW THE TTY TO ENTER DATA INTO LOC. 176 AT SELECTED POINTS WITHIN THE PROGRAM.

2) THE MACHINE WILL THEN TYPE: SWR=XXXXXXNEW= (XXXXXX IS THE OCTAL CONTENTS OF THE SOFTWARE SWITCH REGISTER.)

3) AFTER THE ''NEW='' HAS BEEN TYPED THEN THE OPERATOR CAN DO ONE OF THE FOLLOWING AT THE TTY:

   A) TYPE A NUMBER TO BE LOADED INTO LOC. 176 FOLLOWED BY A <CR>. (ONLY NUMBERS BETWEEN 0-7 WILL BE ACCEPTED AND ONLY 6 NUMBERS WILL BE ALLOWED)
   IF A <CR> IS THE FIRST KEY DEPRESSED THE SOFTWARE SWITCH REGISTER CONTENTS WILL NOT BE CHANGED.

   B) IF A CONTROL U (↑U) IS DEPRESSED THEN THE PROGRAM WILL SEND YOU BACK TO STEP 2.

4.1 OPERATIONAL SWITCH SETTINGS

WITH SW<15:0>=1, THE PROGRAM WILL TYPEOUT ERRORS AND CONTINUE WITH THE TEST.

THE SWITCH SETTINGS ARE:

```
SW<15>=1...HALT ON ERROR
SW<14>=1...LOOP ON TEST
SW<13>=1...INHIBIT ERROR TYPEOUTS
SW<11>=1...INHIBIT ITERATIONS
SW<10>=1...RING BELL ON ERROR
SW<09>=1...LOOP ON ERROR
SW<07>=1...DON'T TYPE DRIVE STATUS WHEN PROGRAM STARTED OR WHEN
           ENTERING CONVERSATION MODE.
           CHANGE HEAD SELECTION IN TEST 16.
SW<06>=1...DON'T USE MEMORY MANAGEMENT IN TEST 14
SW<05>=1...DISPLAY ALL READ RETRY ATTEMPTS IN TEST 14
SW<04>=1...TYPE ALL DATA COMPARE ERRORS IN TESTS 13 & 14
SW<03>=1...INHIBIT DATA COMPARE IN TESTS 13 & 14
SW<02>=1...BYPASS READ & DATA COMPARE IN TEST 14
SW<01>=1...BYPASS WRITE CHECK IN TEST 14
SW<00>=1...BYPASS WRITE IN TEST 14
```

4.2 TELETYPE KEYBOARD AND PRINTER UNIBUS ADDRESSES

THE PROGRAM ASSUMES THE FOLLOWING UNIBUS ADDRESSES. THESE ADDRESSES MAY BE CHANGED AT THE INDICATED LOCATION BEFORE STARTING THE PROGRAM.

MEMORY LOCATION	CONTENTS	FUNCTION
1136	177560	TTY KEYBOARD STATUS REG
1140	177562	TTY KEYBOARD BUFFER REG
1142	177564	TTY PRINTER STATUS REGISTER
1144	177566	TTY PRINTER BUFFER REG

(THE RP11E UNIBUS ADDRESS IS CHANGED IN RESPONSE TO THE PROGRAM'S TYPED REQUEST IF THE PROGRAM IS STARTED FROM EITHER LOCATION 210 OR 214 OR IF THE THERE IS NO RESPONSE WHEN LOCATION 176710 IS ADDRESSED.)

4.3 TTY KEYBOARD ENTRY CONTROL CODES AND CONVENTIONS

THE FOLLOWING CONTROL CODES AND CONVENTIONS ARE USED BY THE PROGRAM:

CARRIAGE RETURN (CR) - ALL ENTRIES MUST BE TERMINATED BY A CARRIAGE RETURN.

CONTROL U (↑U) - IF THIS CONTROL CODE IS TYPED BEFORE AN ENTRY IS TERMINATED, THE ENTRY IS CLEARED AND ANOTHER VALUE MAY BE ENTERED. A ↑U IS TYPED BY PRESSING THE 'CONTROL' AND 'U' KEYS TOGETHER.

RUBOUT (RO) - IF THIS CONTROL CODE IS TYPED, THE CHARACTER TYPED BEFORE THE 'RO' WILL BE DELETED FROM THE ENTRY. EACH TIME A 'RO' IS TYPED, THE LAST CHARACTER IN THE INPUT ENTRY IS DELETED. WHEN A 'RO' IS TYPED, THE ENTRY ROUTINE WILL TYPE A BACKSLASH ('\') CHARACTER AND THE CHARACTER BEING DELETED. WHEN A CHARACTER OTHER THAN A 'RO' IS TYPED, THE ENTRY ROUTINE WILL TYPE ANOTHER BACKSLASH CHARACTER AND TYPE THE NEW CHARACTER ENTERED. THE 'RO' CANNOT BE USED TO DELETE A 'CR'.

CONTROL C (↑C) - THE OPERATION OF THE '↑C' CHARACTER IS DESCRIBED IN SECTION 3.4. '↑C' MAY ALSO BE USED TO ABORT THE CONVERSATION MODE AND TO RETURN TO THE BEGINNING OF CONVERSATION MODE. '↑C' IS GENERATED BY PRESSING THE 'CONTROL' KEY AND THE 'C' KEY.

IF ONLY A 'CR' IS ENTERED IN RESPONSE TO AN ENTRY REQUEST FROM THE PROGRAM, THE PROGRAM WILL USE THE DEFAULT VALUE FOR THE REQUESTED ENTRY.

LEADING ZEROS ARE NOT REQUIRED ON ANY ENTRY.

4.4 RP11E UNIBUS ADDRESS CHANGE ROUTINE

WHEN THE PROGRAM IS STARTED FROM LOCATION 200 OR LOCATION 204, THE PROGRAM WILL ASSUME THAT THE RP11E IS AT UNIBUS ADDRESS 176710. WHEN THE PROGRAM IS STARTED FROM EITHER LOCATION 210 OR LOCATION 214, THE PROGRAM ENTERS A ROUTINE WHICH ALLOWS THE OPERATOR TO CHANGE THE RP11E UNIBUS ADDRESS TO BE USED BY THE PROGRAM.

WHEN THE ADDRESS CHANGE ROUTINE IS ENTERED, THE PROGRAM TYPES THE PRESENT RP11E UNIBUS ADDRESS:

```
RPADR = AAAAAA/
```

NOTE: 'AAAAAA' REPRESENTS THE PRESENT ADDRESS

THE ADDRESS TYPED OUT MAY BE USED BY TYPING A 'CR' ONLY. A NEW ADDRESS IS ENTERED BY TYPING THE NEW ADDRESS AND TERMINATING THE ENTRY WITH A 'CR'. (THE NEW ADDRESS WILL BE USED UNTIL ANOTHER ADDRESS IS ENTERED.)

AFTER THE OPERATOR HAS RESPONDED TO THE ADDRESS ENTRY REQUEST, THE PROGRAM WILL VERIFY THAT THE RP11E ADDRESS RESPONDS. IF NO RESPONSE IS RECEIVED FROM THE RP11E ADDRESS, THE FOLLOWING ERROR MESSAGE IS TYPED:

```
RP11E DIDN'T RESPOND TO ADDRESSING
RPADR
AAAAA
```

THE PROGRAM RETURNS TO THE ADDRESS ENTRY ROUTINE. THE PROGRAM WILL NOT CONTINUE UNTIL THE OPERATOR ENTERS AN ADDRESS FOR WHICH THERE IS A UNIBUS RESPONSE.

4.5 CONVERSATION MODE

THE PROGRAM CONTAINS A 'CONVERSATION MODE' ROUTINE WHICH ALLOWS THE OPERATOR TO SELECT AND EXECUTE INDIVIDUAL TESTS AND TO SPECIFY THE DRIVE TO BE USED. THE CONVERSATION MODE ALSO ALLOWS THE OPERATOR TO CHANGE THE OPERATING PARAMETERS OF THE 'DATA STORAGE/RETRIEVAL TEST' (TEST 14).

WHEN THE PROGRAM ENTERS CONVERSATION MODE, THE FOLLOWING MESSAGE IS TYPED:

ENTER TEST #:

A 'CR' MAY BE ENTERED OR A VALID TEST NUMBER (E.G., 0 - 17) MAY BE ENTERED. THE DEFAULT ENTRY ('CR') WILL CAUSE THE PROGRAM TO SELECT TESTS 0 - 14. IF A SINGLE TEST NUMBER IS ENTERED, ONLY THAT TEST WILL BE EXECUTED.

THE PROGRAM WILL THEN ASK FOR A DRIVE ASSIGNMENT.

ENTER DRIVE #:

A 'CR' ENTRY WILL CAUSE THE PROGRAM TO SELECT ALL ONLINE DRIVES FOR TESTING. IF A VALID DRIVE NUMBER IS ENTERED (0 - 7), THE PROGRAM WILL CHECK THE STATE OF THE DRIVE; IF THE DRIVE IS NOT ONLINE, THE DRIVE'S STATUS WILL BE TYPED AND THE PROGRAM WILL ASK FOR ANOTHER DRIVE ENTRY.

IF TEST 15, 16, OR 17 IS SELECTED, A 'CR' RESPONSE TO THE DRIVE NUMBER REQUEST WILL NOT BE ACCEPTED; A SPECIFIC DRIVE NUMBER MUST BE ENTERED FOR TEST 15, 16, OR 17.

IF TESTS 0 - 13, 15, 16 OR 17 HAVE BEEN SELECTED, OR IF THE ENTIRE TEST SEQUENCE WERE SPECIFIED, THE PROGRAM WILL BEGIN EXECUTING THE THE TEST OR SEQUENCE IMMEDIATELY. IF TEST 14 IS SPECIFIED, THE PROGRAM WILL ASK IF THE OPERATOR WANTS TO CHANGE PARAMETERS FOR TEST 14. THE FOLLOWING MESSAGE IS TYPED.

CHANGE PARAMETERS FOR TEST 14 ?

EITHER A 'CR' OR A 'Y' MAY BE ENTERED. IF A 'CR' IS ENTERED, THE PROGRAM WILL BYPASS THE TEST 14 PARAMETER ENTRY ROUTINE AND START TEST 14. IF A 'Y' IS ENTERED, THE PROGRAM STARTS THE TEST 14 PARAMETER CHANGE DIALOG.

THE PROGRAM WILL THEN TYPE THE WORD LENGTH FOR ALL TEST 14 TRANSFERS.

MAXIMUM WORD COUNT FOR TEST 14 (IN OCTAL): SSSSSS

NOTE: 'SSSSSS' = THE OCTAL VALUE OF THE WORD COUNT

THE PROGRAM WILL THEN ASK FOR A NEW WORD COUNT.

ENTER NEW WORD COUNT:

EITHER A VALUE (LESS THAN THE MAXIMUM TYPED ABOVE) OR A 'CR'
MAY BE ENTERED. IF A VALUE IS ENTERED, THE NEW VALUE WILL BE USED
FOR ALL DATA TRANSFER OPERATIONS IN TEST 14. WHEN THE PROGRAM IS
RESTARTED OR A 'CONTROL C' IS TYPED, THE WORD COUNT VALUE WILL BE
RETURNED TO THE SYSTEM DEFAULT VALUE. IF A 'CR' IS TYPED, THE
PROGRAM WILL USE THE DEFAULT WORD COUNT VALUE.

THE PROGRAM WILL THEN ASK FOR A PATTERN SELECTION WORD ENTRY.

ENTER PATTERN SELECTION CODE FOR TEST 14:

ENTER THE OCTAL VALUE OF THE PATTERN SELECTION WORD OR A 'CR' FOR
THE DEFAULT VALUE. REFER TO SECTION 7.9 FOR A DESCRIPTION OF
THE PATTERN SELECTION WORD AND ITS USE IN TEST 14. IF TESTING IS TO
BE AT A SPECIFIC DISK LOCATION, ENTER THE CODE FOR A SINGLE PATTERN.
(IF TESTING IS AT A SPECIFIC ADDRESS, THE PROGRAM WILL NOT ROTATE THE
DATA PATTERNS BUT WILL USE THE LOWEST NUMBERED PATTERN SPECIFIED
BY THE PATTERN SELECT WORD.)

THE PROGRAM WILL ASK IF THE OPERATOR WANTS TO TEST AT A SPECIFIC
DISK ADDRESS.

USE A SPECIFIC DISK ADDRESS FOR TEST 14 ?

ENTER EITHER A 'CR' OR A 'Y'. IF A 'Y' IS ENTERED, THE PROGRAM
WILL TYPE THE FOLLOWING REMINDER MESSAGE ABOUT DATA PATTERN USAGE.

(A SINGLE DATA PATTERN MUST HAVE BEEN SPECIFIED
IF NOT, PATTERN 0 (ALL ZEROS) WILL BE USED)

THE PROGRAM WILL ASK FOR CYLINDER, TRACK, AND SECTOR ADDRESSES.
AN ADDRESS MUST BE ENTERED, THE PROGRAM WILL NOT ACCEPT DEFAULT
ENTRIES FOR THESE ADDRESSES. THE ENTRY REQUESTS FOR THE ADDRESSES ARE
GIVEN BELOW.

ENTER CYLINDER ADDRESS:

ENTER TRACK ADDRESS:

ENTER SECTOR ADDRESS:

IF A SPECIFIC ADDRESS IS ENTERED FOR TESTING, TESTING WILL TAKE
PLACE ONLY AT THAT ADDRESS WITH THE WORD COUNT AND PATTERN ENTERED
ABOVE. THE TEST WILL MAKE ONE PASS AND GO TO END OF TEST. SW<14>
SHOULD BE SET TO INHIBIT THE END OF TEST MESSAGE TYPEOUT.

TO TERMINATE THE TEST OR SEQUENCE, EITHER TYPE A 'CONTROL C' OR HALT
THE PROCESSOR AND RESTART AT THE APPROPRIATE STARTING ADDRESS.

4.6 OPERATION OF TESTS 15, 16, & 17

TEST 15 AND TEST 17 CONTAIN IMBEDDED OPERATING INSTRUCTIONS. SETUP AND OPERATING INSTRUCTIONS ARE TYPED OUT AS THE TEST PROCEEDS.

TO OPERATE TEST 16, 'HEAD ALIGNMENT ROUTINE', REFER TO THE APPLICABLE MAINTENANCE MANUALS FOR THE DRIVE SETUP PROCEDURES. THE PROGRAM WILL TYPEOUT THE CONTROLLER SETUP PROCEDURES. TO CHANGE HEAD SELECTION, SW<07> MUST BE TOGGLED (TURNED ON, THEN OFF).

TO SETUP THE RP11E FOR HEAD ALIGNMENT, CONNECT A JUMPER BETWEEN PINS J13M2 AND J13C2 ON THE BACKPLANE.

5. ERROR HANDLING

5.1 ERROR REPORTING

WHEN THE PROGRAM DETECTS AN ERROR, THE ERROR ROUTINE IS CALLED; IF SW<13> IS NOT SET, THE ERROR MESSAGE FOR THAT ERROR WILL BE TYPED. EACH ERROR TYPEOUT CONTAINS THE FOLLOWING:

A. AN ERROR MESSAGE
B. A DATA HEADER LINE
C. A DATA LINE CONTAINING
    1. THE TEST NUMBER
    2. THE ADDRESS (PROGRAM COUNTER CONTENTS) WHERE THE ERROR CALL WAS MADE
    3. CONTENTS OF THE APPROPRIATE REGISTERS
    4. OTHER SPECIFIC DATA

5.2 ERROR TYPES

THE ERRORS DETECTED ARE DIVIDED INTO 2 CATEGORIES: FATAL AND NON-FATAL.

FATAL ERRORS ARE DRIVE UNSAFE ERRORS WHICH OCCUR ON THE DRIVE BEING USED FOR TESTING OR DRIVES WHICH GO OFFLINE DURING TESTING. IF EITHER OF THESE TWO CONDITIONS OCCUR, THE PROGRAM WILL TERMINATE TESTING USING THE DRIVE WHICH GAVE THE FATAL ERROR.

ALL OTHER ERROR TYPES ARE NON-FATAL, E.G., TESTING CONTINUES AFTER THE ERROR IS REPORTED.

6. RESTRICTIONS

6.1 SUBSYSTEMS WITH MIXED DRIVE TYPES

THE PROGRAM ASSUMES THAT ONLY ONE KIND OF DRIVE IS ATTACHED TO A SYSTEM (E.G. RP02'S & RPR02'S OR RP03'S). IF BOTH RP02'S (AND RPR02'S) OR RP03'S ARE ON THE SAME SYSTEM, THE PROGRAM WILL ASSUME THAT ALL OF THE DRIVES ARE RP03'S. TO TEST DRIVES ON A MIXED SYSTEM, TEST ALL OF THE SAME KIND OF DRIVE AT ONE TIME. THE OTHER DRIVES ON THE CONTROLLER MUST BE POWERED DOWN.

6.2 FORMATTED PACKS

EACH OF THE DRIVES BEING USED MUST HAVE A FORMATTED DISK PACK.

6.3 'CONTROL C' TERMINATION DURING TESTS 5 & 13

IF TESTS 5 OR 13 ARE TERMINATED WITH A 'CONTROL C', CYLINDER 0, TRACK 0, SECTOR 0 MAY BE LEFT WITH EITHER AN INVALID HEADER OR AN INVALID DATA FIELD. IF TEST 5 IS STOPPED, THE HEADER FOR SECTOR 0 MAY STILL BE MISFORMATTED; IF TEST 13 IS STOPPED, SECTOR 0 MAY BE STILL RECORDED IN 10/15 MODE.

6.4 CONTROLLER INTERRUPT

THE PROGRAM DOES NOT USE CONTROLLER INTERRUPT. THE PROGRAM WAITS FOR THE CONTROLLER 'READY' OR DRIVE 'READY' BITS TO SET TO INDICATE ORDER TERMINATION.

7. MISCELLANEOUS

7.1 RUN TIME

TO MAKE ONE PASS OF THE PROGRAM (TESTS 0 - 14) TAKES APPROXIMATELY 20 MINUTES FOR EACH RP02 OR RPR02 USED AND APPROXIMATELY 40 MINUTES FOR EACH RP03 USED.

7.2 STACK POINTER

STACK IS INITIALLY SET TO 1100 AND EXTENDS DOWNWARD IN MEMORY.

7.3 END OF PASS/END OF TEST

THE PROGRAM WILL PERFORM ONE PASS FOR EACH DRIVE BEING USED FOR TESTING. END OF TEST WILL OCCUR WHEN EACH DRIVE BEING USED FOR TESTING HAS HAD ONE PASS.

7.4 SUBROUTINE CALLS

THE SUBROUTINE CALLS USED BY THE PROGRAM ARE:

A. 'SCOPE' (IOT INSTRUCTION). THIS CALL IS PLACED BETWEEN EACH TEST IN THE PROGRAM. THIS ROUTINE ESTABLISHES THE TEST ITERATION COUNT AND LOOP ON TEST AND LOOP ON ERROR ADDRESSES.

B. 'ERROR' (EMT INSTRUCTION). THIS CALL IS USED TO REPORT ALL ERRORS. A CALL TO THIS ROUTINE IS FOLLOWED BY A NUMBER WHICH IDENTIFIES THE ERROR MESSAGE WHICH WILL BE TYPED.

THE TRAP INSTRUCTION IS USED FOR THE FOLLOWING SUBROUTINE CALLS:

TYPE - TTY TYPEOUT ROUTINE
TYPOC - TYPE OCTAL NUMBER (WITH LEADING ZEROS)
TYPOS - TYPE OCTAL NUMBER (NO LEADING ZEROS)
TYPDS - TYPE DECIMAL NUMBER
RDCHR - READ A CHARACTER FROM THE TTY KEYBOARD
RDLIN - READ A LINE FROM THE TTY KEYBOARD
SAVREG - ROUTINE TO SAVE R0 - R5
RESREG - ROUTINE TO RESTORE R0 - R5

7.5 DISK SURFACE USAGE

THE PROGRAM USES THE FOLLOWING DISK LOCATIONS FOR TESTING.

TEST	DISK ADDRESS
0	CYL 0, TRK 0, SEC 0
1	CYL 0, TRK 0, SEC 0
2	CYL 0, TRK 0, SEC 0
3	CYL 0, TRK 0, SEC 0 MAXIMUM CYL, TRK 19, SEC 9
4	CYL 0, TRK 0, SEC 0
5	CYL 0, TRK 0, SEC 0
6	CYL 192, TRK 0, SEC 0
7	CYL 0, TRK 0, SEC 0
10	CYL 0, TRK 0, SEC 0 - 9
11	CYL 0, TRK 0 - 19, SEC 0
12	CYL 0, TRK 0, SEC 0
13	CYL 0, TRK 0, SEC 0
14	ENTIRE PACK
15	MAXIMUM CYL, TRK 0, SEC 0
16	N/A
17	NONE

7.6 TESJ EXECUTION SEQUENCE

THE TEST SEQUENCE IS TESTS 0 - 14. TESTS 15, 16, & 17 ARE ONLY PERFORMED IF DIRECTED BY THE OPERATOR FROM CONVERSATION MODE. TESTS 15, 16, OR 17 CANNOT BE EXECUTED IN SEQUENCE.

7.7 DRIVE STATUS TYPEOUT

WHEN THE PROGRAM IS STARTED FROM LOCATION 200, 204, 210, OR 214 OR ENTERS THE CONVERSATION ROUTINE FROM A 'CONTROL C', THE PROGRAM WILL TYPE THE STATUS OF DRIVES 0 - 7. THIS DRIVE STATUS TYPEOUT CAN BE INHIBITED BY SETTING SW<07> BEFORE THE PROGRAM IS STARTED OR A 'CONTROL C' IS TYPED.

7.8 DATA COMPARSION USING MEMORY MANAGEMENT

IF THE SYSTEM HAS MORE THAN 28K, TEST 14 WILL ROTATE THE READ BUFFERS THROUGH ALL OF MEMORY. IF A DATA COMPARSION ERROR IS DETECTED, THE PROGRAM WILL TYPEOUT THE 'GOOD' AND THE 'BAD' DATA AND THE V I R T U A L ADDRESS OF THE COMPARSION ERROR LOCATION. THE CONTENTS OF THE APR'S IN USE WILL ALSO BE TYPED IF MEMORY MANAGEMENT IS ENABLED. THE PROGRAM PERFORMS ALL DATA COMPARSIONS FROM VIRTUAL LOCATIONS 60000 - 137776 USING APR'S 3, 4, & 5. THE PHYSICAL ADDRESS OF THE COMPARSION ERROR CAN BE FORMED BY ADDING THE APPLICABLE 'KIPAR' REGISTER CONTENTS (THE REGISTERS ARE DISPLAYED) TO THE VIRTUAL ADDRESS OF THE ERROR.

7.9 PATTERN SELECTION WORD

THE DATA PATTERNS USED BY TEST 14 ARE SELECTED FROM A SELECTION WORD WHICH IS LOADED BY THE PROGRAM OR LOADED BY THE OPERATOR IN CONVERSATION MODE. THE PATTERN SELECTION WORD CONTAINS A BIT FOR EACH PATTERN TO BE USED; BIT00 REPRESENTS PATTERN 0, BIT01 REPRESENTS PATTERN 1, ETC. REFER TO SECTION 8.13 FOR THE CONTENTS OF EACH PATTERN.

THE OCTAL VALUES FOR EACH DATA PATTERN ARE AS FOLLOWS.

PATTERN #	OCTAL SELECTION CODE
0	1
1	2
2	4
3	10
4	20
5	40
6	100
7	200
8	400
9	1000
10	2000
11	4000
12	10000
13	20000
14	40000
15	100000

THE INDIVIDUAL OCTAL CODES ARE 'OR ED' TOGETHER TO FORM THE PATTERN SELECTION WORD. TO SELECT PATTERNS 0, 6, & 13, THE PATTERN SELECTION WORD COULD BE THE FOLLOWING OCTAL VALUE: 20101.

7.10 CONTROLLER/DRIVE OPERATION TIME LIMIT

ALL CONTROLLER AND DRIVE OPERATRIONS ARE TIMED BY THE PROGRAM. IF AN OPERATION EXCEEDS THE TIME LIMIT, THE PROGRAM WILL REPORT THE TIMEOUT, RESET THE CONTROLLER, TERMINATE THE CURRENT TEST, AND GO TO THE NEXT TEST. THE TIME LIMIT IS 150 MS TO 1.5 SEC (DEPENDING ON THE PROCESSOR).

8. TEST DESCRIPTIONS

8.1 TEST 0 - 'CLEAR' COMMAND TERMINATION TEST

VERIFY THAT 'CLEAR' TERMINATES AN OPERATION AND CAUSES CONTROLLER READY TO SET.

8.2 TEST 1 - WRITE CHECK COMMAND TEST

THIS TEST VERIFIES THE WRITE CHECK LOGIC AND TO VERIFY THAT A WRITE CHECK ERROR CAN BE DETECTED. 'FLOATING ONES' AND 'FLOATING ZEROS' PATTERNS ARE USED TO TEST THE WRITE CHECK COMPARE LOGIC.

8.3 TEST 2 - PARTIAL SECTOR WRITE TEST

CHECK THE ABILITY OF THE RP11E TO CLEAR THE REMAINDER OF A SECTOR ON A PARTIAL WRITE OPERATION. A SECTOR OF ALL ONES IS WRITTEN AND THEN A TWO WORD WRITE OPERATION IS PERFORMED. THE SECTOR IS THEN READ BACK AND VERIFIED. THE FIRST TWO WORDS SHOULD BE ONES AND THE REST SHOULD BE ZEROS.

8.4 TEST 3 - 'EOP' TEST

TEST THAT 'EOP' SETS WHEN THE CONTROLLER TRIES TO WRITE BEYOND THE LIMITS OF THE DRIVE. THE FIRST SECTOR OF THE PACK IS WRITTEN WITH ZEROS; THEN A TWO SECTOR WRITE OF ALL ONE'S IS ISSUED FOR THE MAXIMUM BE SET. THE FIRST SECTOR OF THE PACK IS CHECKED TO MAKE SURE THAT IS STILL CONTAINS ZEROS.

8.5 TEST 4 - 'PROG' ERROR TEST

VERIFY THAT THE RP11E GENERATES A 'PROG' ERROR IF A COMMAND IS ISSUED WHILE THE CONTROLLER IS BUSY.

8.6 TEST 5 - 'HEADER NOT FOUND' TEST

MISFORMAT THE FIRST SECTOR ON THE PACK AND THEN READ IT BACK. VERIFY THAT READ AND WRITE HEADER OPERATIONS WILL TRANSFER DATA CORRECTLY. ISSUE A WRITE COMMAND TO SECTOR ZERO; THIS SHOULD CAUSE 'HEADER NOT FOUND' TO SET. THE TEST SECTOR IS REFORMATTED.

8.7 TEST 6 - COMMAND BUFFERING TEST

ISSUE A SEEK COMMAND AND WAIT FOR DONE TO SET. THEN ISSUE A READ COMMAND WHILE THE HEADS ARE STILL MOVING. THE RP11E SHOULD HOLD THE READ COMMAND UNTIL THE SEEK IS COMPLETE.

8.8 TEST 7 - 'NXME' ERROR TEST

THE NON-EXISTENT MEMORY ERROR BIT IS TESTED BY ATTEMPTING TO READ 1 WORD INTO LOCATION 760000. 'NXME', 'HE', & 'ERR' SHOULD ALL BE SET.

8.9 TEST 10 - SECTOR ADDRESSING TEST

THE SECTOR ADDRESSING TEST TESTS THE CAPABILTIY OF THE RP11E TO LOCATE A SECTOR BY USING THE 'SOT' COUNTER (FOR 'HDR' OPERATIONS) AND TO LOCATE A SECTOR BY COMPARING THE HEADER CONTENTS AGAINST THE CYLINDER AND TRACK/SECTOR ADDRESS REGISTERS (NORMAL MODE). THE TEST IS PERFORMED IN 3 PARTS.

1. WRITE ALL HEADERS ON TRACK 0, CYLINDER 0 IN ASCENDING SEQUENCE FROM INDEX (0, 1, 2 ... 9).

2. READ EACH HEADER, BEGINNING WITH SECTOR 0, AND VERIFY THAT THE SECTOR FIELD IN THE HEADER IS CORRECT FOR THE HEADER EXPECTED.

VERIFY THAT THE 'SOT' COUNTER AND THE SECTOR ADDRESS IN 'RPDA' ARE EQUAL AND ARE THE EXPECTED VALUE. THE EXPECTED VALUE IS 1 GREATER THAN THE SECTOR READ. THE SECTOR ADDRESS REGISTER AND THE 'SOT' WILL BE INCREMENTED BY THE TIME THE PROGRAM IS ABLE TO READ THE REGISTER.

WRITE THE SECTOR'S ADDRESS IN THE DATA FIELD. (FOR SECTOR 0, AN OCTAL 12 IS WRITTEN.)

CONTINUE THIS SEQUENCE FOR THE REMAINING SECTORS ON THE TRACK.

3. READ THE DATA FROM EACH SECTOR USING A NORMAL (I.E., NON-HEADER) READ AND VERIFY THAT THE DATA IS CORRECT FOR THE EXPECTED SECTOR.

8.10 TEST 11 - TRACK ADDRESSING TEST

TRACK ADDRESSING IS TESTED BY WRITING THE TRACK'S ADDRESS AS DATA IN SECTOR 0 OF EACH TRACK IN CYLINDER 0. SECTOR 0 IS THEN READ BACK AND THE DATA IS CHECKED TO VERIFY THAT THE PROPER HEAD WAS SELECTED.

8.11 TEST 12 - EXTENDED MEMORY ADDRESS BIT TEST

THIS TEST TESTS THE OPERATION OF THE EXTENDED MEMORY ADDRESS BITS. IF THE SYSTEM DOES NOT HAVE MEMORY MANAGEMENT OR HAS MEMORY MANAGEMENT AND ONLY 32K, THE TEST WILL NOT BE PERFORMED.

1. THE PROGRAM WRITES A 2 WORD TEST SECTOR OF ALL ONES.

2. EXTENDED ADDRESS BIT 'MEX00' IS TESTED BY CLEARING LOCATION 200000 AND READING THE TEST SECTOR INTO LOCATION 177776. LOCATION 200000 IS CHECKED TO VERIFY THAT THE DATA IS CORRECT (ONES). THE PROGRAM VERIFIES THAT 'MEX00' HAS SET AND THAT 'MEX01' IS NOT SET.

3. IF THE SYSTEM HAS AT LEAST 64K, 'MEX01' IS TESTED. LOCATION 400000 IS CLEARED AND THE TEST SECTOR IS READ INTO LOCATION 377776 ('MEX00' IS SET FOR THE READ). LOCATION 400000 IS CHECKED FOR THE PROPER CONTENTS (ONES). 'MEX00' SHOULD HAVE RESET AND 'MEX01' SHOULD BE SET.

8.12 TEST 13 - RP11E DATA BUFFER REGISTER BIT TEST

THE DATA BUFFER REGISTER TEST VERIFIES THAT ALL 36 BITS OF THE DATA BUFFER REGISTER CAN BE SET AND CLEARED.

1. THE TEST WRITES THE FOLLOWING TEST PATTERN IN CYLINDER 0, TRACK 0, SECTOR 0 USING 10/15 MODE ('MODE' BIT SET).

```
052525
052525
000005
125252
125252
125252
000012
052525
052525
000005
125252
  .
  .
  .
ETC
```

2. THE TEST SECTOR IS THEN READ AND THE DATA IS COMPARED. AT THE COMPLETION OF THE TEST, THE PROGRAM RESTORES THE TEST SECTOR TO PDP-11 MODE.

8.13 TEST 14 - DATA STORAGE/RETRIEVAL TEST

THE DATA TEST PERFORMS DATA STORAGE AND RETREIVAL TESTING USING THE CONTROLLER AND THE CURRENTLY SELECTED DRIVE. UNLESS ALTERED BY THE OPERATOR, THE TEST WILL WRITE AND WRITE CHECK THE ENTIRE DISK PACK USING ALL 16 DATA PATTERNS IN A ROTATING MANNER. THE BUFFER SIZE USED FOR THE WRITE AND WRITE CHECK ORDERS IS DETERMINED BY THE AVAILABLE MEMORY ON THE SYSTEM (MINUS THE SPACE REQUIRED BY THE PROGRAM LOADER). THE PROGRAM WILL USE THE AVAILABLE MEMORY SIZE OR 8K, WHICHEVER IS LESS, AS THE BUFFER SIZE.

AFTER THE PACK HAS BEEN WRITTEN, THE TEST WILL READ AND COMPARE THE DATA. THE BUFFER SIZE USED FOR THE READ IS THE SAME SIZE AS THAT USED FOR THE WRITE/WRITE CHECK SEGEMENT OF THE TEST. FOR THE READ SEGEMENT OF THE TEST, THE PROGRAM ROTATES THE READ BUFFER THROUGH MEMORY SO THAT ALL OF AVAILABLE MEMORY IS USED BY THE READ AND COMPARE OPERATIONS. THE TEST USES MEMORY MANAGEMENT ON SYSTEMS WITH MORE THAN 28K. (MEMORY MANAGEMENT WILL NOT BE USED IF THE TEST IS RUN ON MEMORY MANAGEMENT SYSTEMS WITH 28K OR LESS.)

IF AN ERROR OCCURS DURING A READ OPERATION (EXCEPT DATA COMPARSION ERRORS), THE PROGRAM WILL RETRY THE READ ORDER UP TO 3 TIMES. WRITE AND WRITE CHECK ORDERS ARE NOT RETRIED.

THE DATA PATTERNS USED BY THE DATA TEST ARE GIVEN BELOW:

PAT 0	PAT 1	PAT 2	PAT 3	PAT 4	PAT 5	PAT 6	PAT 7
000000	000001	177776	000000	000000	052525	007417	026455
000000	000003	177774	000000	010421	052525	007417	026455
000000	000007	177770	000000	021042	052525	007417	026455
000000	000017	177760	177777	031463	125252	170360	151322
000000	000037	177740	177777	042104	125252	170360	151322
000000	000077	177700	177777	052525	125252	170360	151322
000000	000177	177600	000000	063146	052525	007417	026455
000000	000377	177400	000000	073567	052525	007417	026455
000000	000777	177000	177777	104210	125252	170360	151322
000000	001777	176000	177777	114631	125252	170360	151322
000000	003777	174000	000000	125252	052525	007417	026455
000000	007777	170000	177777	135673	125252	170360	151322
000000	017777	160000	000000	146314	052525	007417	026455
000000	037777	140000	177777	156735	125252	170360	151322
000000	077777	100000	177777	167356	125252	170360	151322
000000	177777	000000	000000	177777	052525	007417	026455

PAT 8	PAT 9	PAT 10	PAT 11	PAT 12	PAT 13	PAT 14	PAT 15
077577	000001	177776	172666	077777	177777	000000	177777
077577	000002	177775	155555	137777	177777	000000	177777
077577	000004	177773	172666	157777	177777	177777	000000
077577	000010	177767	155555	167777	177777	177777	000000
077577	000020	177757	172666	173777	177777	177777	000000
077577	000040	177737	155555	175777	177777	177777	000000
077577	000100	177677	172666	176777	177777	177777	000000
077577	000200	177577	155555	177377	177777	177777	000000
077577	000400	177377	172666	177577	177777	177777	000000
077577	001000	176777	155555	177677	177777	177777	000000
077577	002000	175777	172666	177737	177777	177777	000000
077577	004000	173777	155555	177757	177777	177777	000000
077577	010000	167777	172666	177767	177777	177777	000000
077577	020000	157777	155555	177773	177777	177777	000000
077577	040000	137777	172666	177775	177777	177777	000000
077577	100000	077777	155555	177776	177777	177777	000000

8.14 TEST 15 - POWER FAIL TEST

TEST THE ABILITY OF THE RP11E TO SENSE POWER FAILURE AND FOR THE DRIVES TO RETRACT THEIR HEADS. WHEN POWER IS TURNED ON AGAIN THE CYLINDER ADDRESS IS TESTED FOR ZERO. AFTER TYPING THE MESSAGE REQUESTING POWER TO BE TURNED OFF, THE PROGRAM GOES INTO A LOOP READING FROM THE SELECTED DISK DRIVE. AFTER POWER IS RESTORED, MEMORY IS CHECKED TO SEE THAT THE DISK DID NOT TRANSFER ANY DATA TO MEMORY WHILE POWER WAS GOING DOWN.

8.15 TEST 16 - HEAD ALIGNMENT ROUTINE

THIS ROUTINE PERFORMS HEAD SELECTION AS DIRECTED FROM THE KEYBOARD; ALIGNMENT OF THE SELECTED HEAD MAY BE CHECKED OR ADJUSTED.

8.16 TEST 17 - RP11E/DRIVE CONTROL PANEL SWITCH TEST

THIS TEST TESTS THE 'ENABLE/DISABLE' AND 'READ ONLY' SWITCHES ON THE DRIVE CONTROL PANEL AND TESTS THE 'WRITE LOCKOUT' AND 'LOA' SWITCHES ON THE RP11E CONTROL PANEL.

9. PROGRAM LISTING

%

```
                        .TITLE  MD-11-DZRPY-C, RP11-E FUNCTIONAL LOGIC & R/W TEST
                        ;*COPYRIGHT (C) 1975,1977
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY C. HESS/F. ROEMER
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*
                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*       SWITCH              USE
                        ;*       ------              ---
                        ;*         15                HALT ON ERROR
                        ;*         14                LOOP ON TEST
                        ;*         13                INHIBIT ERROR TYPEOUTS
                        ;*         11                INHIBIT ITERATIONS
                        ;*         10                BELL ON ERROR
                        ;*          9                LOOP ON ERROR
                        ;*          7                DON'T TYPE DRIVE STATUS WHEN PROGRAM STARTED OR WHEN
                        ;*                           ENTERING CONVERSATION MODE
                        ;*                           CHANGE HEAD SELECTION IN TEST 16
                        ;*          6                DON'T USE MEMORY MANAGEMENT IN TEST 14
                        ;*          5                DISPLAY ALL READ RETRY ATTEMPTS IN TEST 14
                        ;*          4                PRINT ALL COMPARE ERRORS IN TESTS 13 & 14
                        ;*          3                INHIBIT DATA COMPARE IN TESTS 13 & 14
                        ;*          2                BYPASS READ AND DATA COMPARE IN TEST 14
                        ;*          1                BYPASS WRITE CHECK IN TEST 14
                        ;*          0                BYPASS WRITE IN TEST 14

                        .SBTTL  TRAP CATCHER

000000                          .=0
                        ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                        ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                        ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
000174                          .=174
000174  000000  DISPREG: .WORD  0                ;;SOFTWARE DISPLAY REGISTER
000176  000000  SWREG:   .WORD  0                ;;SOFTWARE SWITCH REGISTER

                        .SBTTL  ACT11 HOOKS

                        ;;*********************************************************************
                        ;HOOKS REQUIRED BY ACT11
000200                          $SVPC=.                 ;SAVE PC
000046                          .=46
000046  017050                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
000052                          .=52
000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
000200                          .=$SVPC                 ;; RESTORE PC

                        .SBTTL  STARTING ADDRESSES

000200                  .=200
```

```
                                ;*200 = NORMAL START
000200  000137  003654                  JMP     START1
                                ;*204 = SELECT OPERATING PARAMETERS
000204  000137  003676                  JMP     START2
                                ;*210 = SELECT RP11E ADDRESS
000210  000137  003644                  JMP     START3
                                ;*214 = COMBINATION OF 204 AND 210
000214  000137  003666                  JMP     START4

                                .SBTTL  BASIC DEFINITIONS

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

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

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

                                ;*PRIORITY LEVEL DEFINITIONS
000000                          PR0=    0               ;;PRIORITY LEVEL 0
000040                          PR1=    40              ;;PRIORITY LEVEL 1
000100                          PR2=    100             ;;PRIORITY LEVEL 2
000140                          PR3=    140             ;;PRIORITY LEVEL 3
000200                          PR4=    200             ;;PRIORITY LEVEL 4
000240                          PR5=    240             ;;PRIORITY LEVEL 5
000300                          PR6=    300             ;;PRIORITY LEVEL 6
000340                          PR7=    340             ;;PRIORITY LEVEL 7

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                          SW15=   100000
040000                          SW14=   40000
020000                          SW13=   20000
010000                          SW12=   10000
004000                          SW11=   4000
```

```
                    002000          SW10=   2000
                    001000          SW09=   1000
                    000400          SW08=   400
                    000200          SW07=   200
                    000100          SW06=   100
                    000040          SW05=   40
                    000020          SW04=   20
                    000010          SW03=   10
                    000004          SW02=   4
                    000002          SW01=   2
                    000001          SW00=   1
                                    .EQUIV  SW09,SW9
                                    .EQUIV  SW08,SW8
                                    .EQUIV  SW07,SW7
                                    .EQUIV  SW06,SW6
                                    .EQUIV  SW05,SW5
                                    .EQUIV  SW04,SW4
                                    .EQUIV  SW03,SW3
                                    .EQUIV  SW02,SW2
                                    .EQUIV  SW01,SW1
                                    .EQUIV  SW00,SW0

                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
                    100000          BIT15=  100000
                    040000          BIT14=  40000
                    020000          BIT13=  20000
                    010000          BIT12=  10000
                    004000          BIT11=  4000
                    002000          BIT10=  2000
                    001000          BIT09=  1000
                    000400          BIT08=  400
                    000200          BIT07=  200
                    000100          BIT06=  100
                    000040          BIT05=  40
                    000020          BIT04=  20
                    000010          BIT03=  10
                    000004          BIT02=  4
                    000002          BIT01=  2
                    000001          BIT00=  1
                                    .EQUIV  BIT09,BIT9
                                    .EQUIV  BIT08,BIT8
                                    .EQUIV  BIT07,BIT7
                                    .EQUIV  BIT06,BIT6
                                    .EQUIV  BIT05,BIT5
                                    .EQUIV  BIT04,BIT4
                                    .EQUIV  BIT03,BIT3
                                    .EQUIV  BIT02,BIT2
                                    .EQUIV  BIT01,BIT1
                                    .EQUIV  BIT00,BIT0

                                    ;*BASIC "CPU" TRAP VECTOR ADDRESSES
                    000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
                    000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                    000014          TBITVEC=14              ;;"T" BIT
                    000014          TRTVEC= 14              ;;TRACE TRAP
                    000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
```

```
                        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  MEMORY MANAGEMENT DEFINITIONS

                        ;*KT11 VECTOR ADDRESS

                        MMVEC=  250

                        ;*KT11 STATUS REGISTER ADDRESSES

                        SR0=    177572
                        SR1=    177574
                        SR2=    177576
                        SR3=    172516

                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

                        KIPDR0= 172300
                        KIPDR1= 172302
                        KIPDR2= 172304
                        KIPDR3= 172306
                        KIPDR4= 172310
                        KIPDR5= 172312
                        KIPDR6= 172314
                        KIPDR7= 172316

                        ;*KERNEL "I" PAGE ADDRESS REGISTERS

                        KIPAR0= 172340
                        KIPAR1= 172342
                        KIPAR2= 172344
                        KIPAR3= 172346
                        KIPAR4= 172350
                        KIPAR5= 172352
                        KIPAR6= 172354
                        KIPAR7= 172356

                        ;RP11E REGISTER INDEX EQUATES

                        RPDS=00
                        RPER=02
                        RPCS=04
                        RPWC=06
                        RPBA=10
                        RPCA=12
                        RPDA=14
                        RPM1=16
                        RPM2=20
                        RPM3=22
                        SUCA=24
```

```
                        SILO=26

                        ;RP11 OP CODE DEFINITIONS

000001                  CLEAR=1                 ;CLEAR THE CONTROLLER
000003                  WRTSEK=3                ;WRITE WITH IMPLIED SEEK AND HEAD SELECTION
000005                  RDSEK=5                 ;READ WITH IMPLIED SEEK AND HEAD SELECT
000007                  WRTCHK=7                ;WRITE CHECK WITH IMPLIIED SEEK AND HEAD SELECT
000011                  SEEK=11                 ;SEEK
000013                  WRITE=13                ;WRITE (NO IMPLIED SEEK OR HEAD SELECT)
000015                  HOMSEK=15               ;HOME SEEK
000017                  READ=17                 ;READ (NO IMPLIED SEEK OR HEAD SELECT)


                        ;RPDS REGISTER BIT DEFINITIONS

000400                  SUWP=400                ;SELECTED UNIT WRITE PROTECTED
001000                  SUFU=1000               ;SELECTED UNIT FILE UNSAFE
004000                  SUSI=4000               ;SELECTED UNIT SEEK INCOMPLETE
020000                  SURP03=20000            ;SELECTED UNIT IS AN RP03
040000                  SUOL=40000              ;SELECTED UNIT IS ONLINE
100000                  SURDY=100000            ;SELECTED UNIT IS READY

                        ;RPCS REGISTER BIT DEFINITIONS

000020                  MEX0=20                 ;EXTENDED MEMORY BIT #1
000040                  MEX1=40                 ;EXTENDED MEMORY BIT #2
000100                  IDE=100                 ;INTERRUPT ON DONE ENABLE BIT
000200                  RDY=200                 ;RP11 CONTROLLER READY BIT
004000                  HDR=4000                ;HEADER OPERATION BIT
010000                  MODE=10000              ;MODE BIT
020000                  AIE=20000               ;ATTENTION INTERRUPT ENABLE BIT
040000                  HE=40000                ;HARD ERROR BIT
100000                  ERR=100000              ;ERROR BIT
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  $CMTAG:                         ;;START OF COMMON TAGS
001100  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001102  000             $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
001103  000             $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
001104  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001110  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001114  000             $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001115  001             $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001116  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001122  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001124  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001126  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001130  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001132  000000                  .WORD   0
001134  000             $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001135  000             $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001136  000000                  .WORD   0
001140  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001142  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001144  177560          $TKS:   177560                  ;;TTY KBD STATUS
001146  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001150  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001152  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001154  000             $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001155  002             $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001156  012             $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001157  000             $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001160  000000          $TMPO:  .WORD   0               ;;USER DEFINED
001162  000000          $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
001164  000000          $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
001166  177607  000377  $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
001172  077             $QUES:  .ASCII  /?/             ;;QUESTION MARK
001173  015             $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
001174  000012          $LF:    .ASCIZ  <12>            ;;LINE FEED
                        ;;*****************************************************************
001176  000000          CHKDRV: .WORD   0               ;ADDRESS OF DRIVE BEING USED
001200  000000          BYPASS: .WORD   0               ;TEST EXIT ADDRESS
001202  000000          FLAG:   .WORD   0               ;INTERNAL PROGRAM FLAG WORD
001204  000000          PATNUM: .WORD   0               ;CURRENTLY SELECTED PATTERN
001206  000000          CNTRLC: .WORD   0               ;CONVERSATION MODE FLAG
001210  000000          MAXCYL: .WORD   0               ;MAXIMUM CYLINDER ADDRESS FOR THE DRIVE IN USE
001212  001750          TIMOUT: .WORD   1000.           ;TIME ALLOWED FOR AN OPERATION
001214  000000          ICNT:   .WORD   0               ;TEST ITERATION COUNT
001216  000000          BUSADR: .WORD   0               ;CHANGE RP11 ADDRESS FLAG
001220  000000          STALLT: .WORD   0               ;STALL VALUE
001222  000000          MMACTV: .WORD   0               ;MEMORY MANAGEMENT IS ACTIVE FLAG
```

```
1238  001224  000000  000000          ACTMEM: .WORD   0,0             ;PHYSICAL ADDRESS OF NEXT READ BUFFER
1239  001230  000000  000000          MAXMEM: .WORD   0,0             ;ADDRESS OF LAST MEMORY BANK (FILLED BY $SIZE)
1240  001234  000000  000000          HIMEM:  .WORD   0,0             ;PHYSICAL ADDRESS OF LAST MEMORY BANK (FOR TEST 14)
1241  001240  000000  000000          LOMEM:  .WORD   0,0             ;PHYSICAL ADDRESS OF FIRST BUFFER LOCATION
1242  001244  000000  000000          PAKSIZ: .WORD   0,0             ;SECTORS ON THE PACK ON THE DRIVE BEING TESTED
1243  001250  000000                  INCSEC: .WORD   0               ;SECTOR INCREMENT VALUE
1244  001252  000000                  INCTRK: .WORD   0               ;TRACK INCREMENT VALUE
1245  001254  000000                  BLKS14: .WORD   0               ;NUMBER SECTORS IN TRANSFER IN TEST 14
1246  001256  000000                  WC14:   .WORD   0               ;WORD COUNT FOR TEST 14
1247  001260  000000                  OPRSEL: .WORD   0               ;OPERATOR SELECTED ADDRESS FLAG
1248  001262  020000                  LMTBK:  .WORD   8192.           ;LARGEST TRANSFER PERMITTED
1249  001264  000000                  MAXWC:  .WORD   0               ;MAXMUM WORD COUNT ALLOWED ON THIS SYSTEM
1250  001266  000000                  SCYL:   .WORD   0               ;OPERATOR SELECTED CYLINDER
1251  001270  000000                  STRK:   .WORD   0               ;OPERATOR SELECTED TRACK
1252  001272  000000                  SSEC:   .WORD   0               ;OPERATOR SELECTED SECTOR
1253  001274  000000                  RDERR:  .WORD   0               ;READ RETRY COUNTER
1254  001276  000000                  TSTNMS: .WORD   0               ;TEST SELECTION WORD.  BIT 0 = TEST 0, ETC.
1255  001300  000000                  PATTRN: .WORD   0               ;PATTERN SELECTION WORD FOR TEST 14
1256  001302  000000                  MEMSIZ: .WORD   0               ;ADDRESS OF HIGHEST NON-MEMORY MANAGEMENT
1257                                                                  ;MEMORY LOCATION
1258  001304  000000                  DRVTYP: .WORD   0               ;CONTAINS A BIT FOR EACH RP03 DRIVE
1259                                                                  ;ON THE SYSTEM.  BIT00 = DRIVE 0, ETC.
1260  001306  000000                  DRVMSK: .WORD   0               ;DRIVE TEST SELECTION MASK
1261  001310  000000                  DRVBAD: .WORD   0               ;CONTAINS A BIT FOR THE DRIVE IF THE DRIVE
1262                                                                  ;GOES OFFLINE OR BECOMES UNSAFE DURING TESTING.
1263                                                                  ;BIT00 = DRIVE 0, ETC.
1264  001312  000000                  DRVSEL: .WORD   0               ;CONTAINS A BIT FOR EACH DRIVE TO BE TESTED
1265  001314  000000                  RETRY:  .WORD   0               ;RETRY COUNTER FOR TEST 14
1266  001316  000003                  LRETRY: .WORD   3               ;RETRY LIMIT FOR TEST 14
1267  001320  000003                  CMP.CT: .WORD   3               ;NUMBER OF COMPARE ERROR TYPEOUTS
1268  001322  000000                  LDRFLG: .WORD   0               ;WHEN =0 LOADERS ARE AT TOP OF MEMORY
1269
1270                                  ;RP11 ADDRESS VALUES
1271
1272  001324  176710                  RPADR:  .WORD   176710          ;RP11 BUS ADDRESS
1273  001326  000254  000256          RPVEC:  .WORD   254,256         ;RP11 VECTOR ADDRESS
1274  001332  000240                  RPPRIO: .WORD   <5*32.>          ;RP11 PRIORITY
1275
1276                                  ;SAVE THE RP11E REGISTERS HERE
1277
1278  001334  000000                  $RPDS:  .WORD   0               ;DRIVE STATUS REGISTER
1279  001336  000000                  $RPER:  .WORD   0               ;ERROR REGISTER
1280  001340  000000                  $RPCS:  .WORD   0               ;COMMAND & STATUS REGISTER
1281  001342  000000                  $RPWC:  .WORD   0               ;WORD COUNT REGISTER
1282  001344  000000                  $RPBA:  .WORD   0               ;BUFFER ADDRESS REGISTER
1283  001346  000000                  $RPCA:  .WORD   0               ;CURRENT CYLINDER ADDRESS REGISTER
1284  001350  000000                  $RPDA:  .WORD   0               ;TRACK-SECTOR ADDRESS REGISTER
1285  001352  000000                  $RPM1:  .WORD   0               ;MAINTENANCE REGISTER #1
1286  001354  000000                  $SUCA:  .WORD   0               ;SELECTED UNIT CYLINDER ADDRESS REGISTER
1287  001356  000000                  $SILO:  .WORD   0               ;SILO REGISTER
1288
1289                                  ;ATTENTION BITS
1290
1291  001360      001                 ATABIT: .BYTE   1               ;DRIVE 0
1292  001361      002                         .BYTE   2               ;DRIVE 1
1293  001362      004                         .BYTE   4               ;DRIVE 2
```

```
                                .BYTE   10              ;DRIVE 3
                                .BYTE   20              ;DRIVE 4
                                .BYTE   40              ;DRIVE 5
                                .BYTE   100             ;DRIVE 6
                                .BYTE   200             ;DRIVE 7


                        ;DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
                                ;DRVSTA=0 DRIVE NONEXISTENT OR OFFLINE
                                ;DRVSTA>0 DRIVE IS ONLINE
                                ;DRVSTA<0 DRIVE IS UNSAFE

001370  000            DRVSTA: .BYTE   0               ;DRIVE 0
001371  000                    .BYTE   0               ;DRIVE 1
001372  000                    .BYTE   0               ;DRIVE 2
001373  000                    .BYTE   0               ;DRIVE 3
001374  000                    .BYTE   0               ;DRIVE 4
001375  000                    .BYTE   0               ;DRIVE 5
001376  000                    .BYTE   0               ;DRIVE 6
001377  000                    .BYTE   0               ;DRIVE 7


                        ;BIT TABLE

001400  000001         BITS:  .WORD   BIT00
001402  000002                .WORD   BIT01
001404  000004                .WORD   BIT02
001406  000010                .WORD   BIT03
001410  000020                .WORD   BIT04
001412  000040                .WORD   BIT05
001414  000100                .WORD   BIT06
001416  000200                .WORD   BIT07
001420  000400                .WORD   BIT08
001422  001000                .WORD   BIT09
001424  002000                .WORD   BIT10
001426  004000                .WORD   BIT11
001430  010000                .WORD   BIT12
001432  020000                .WORD   BIT13
001434  040000                .WORD   BIT14
001436  100000                .WORD   BIT15


001440  001500         PAT.PT: .WORD   PAT0            ;TABLE OF POINTERS WHICH POINT TO THE
001442  001540                 .WORD   PAT1            ;PATTERNS USED BY THE DATA TEST
001444  001600                 .WORD   PAT2
001446  001640                 .WORD   PAT3
001450  001700                 .WORD   PAT4
001452  001740                 .WORD   PAT5
001454  002000                 .WORD   PAT6
001456  002040                 .WORD   PAT7
001460  002100                 .WORD   PAT8
001462  002140                 .WORD   PAT9
001464  002200                 .WORD   PAT10
001466  002240                 .WORD   PAT11
001470  002300                 .WORD   PAT12
001472  002340                 .WORD   PAT13
001474  002400                 .WORD   PAT14
001476  002440                 .WORD   PAT15
```

```
1350
1351                                    ;PATTERNS 0 THRU 15
1352
1353  001500  000000                    PAT0:   .WORD   000000          ;PATTERN 0
1354  001502  000000                            .WORD   000000
1355  001504  000000                            .WORD   000000
1356  001506  000000                            .WORD   000000
1357  001510  000000                            .WORD   000000
1358  001512  000000                            .WORD   000000
1359  001514  000000                            .WORD   000000
1360  001516  000000                            .WORD   000000
1361  001520  000000                            .WORD   000000
1362  001522  000000                            .WORD   000000
1363  001524  000000                            .WORD   000000
1364  001526  000000                            .WORD   000000
1365  001530  000000                            .WORD   000000
1366  001532  000000                            .WORD   000000
1367  001534  000000                            .WORD   000000
1368  001536  000000                            .WORD   000000
1369
1370  001540  000001                    PAT1:   .WORD   000001          ;PATTERN 1
1371  001542  000003                            .WORD   000003
1372  001544  000007                            .WORD   000007
1373  001546  000017                            .WORD   000017
1374  001550  000037                            .WORD   000037
1375  001552  000077                            .WORD   000077
1376  001554  000177                            .WORD   000177
1377  001556  000377                            .WORD   000377
1378  001560  000777                            .WORD   000777
1379  001562  001777                            .WORD   001777
1380  001564  003777                            .WORD   003777
1381  001566  007777                            .WORD   007777
1382  001570  017777                            .WORD   017777
1383  001572  037777                            .WORD   037777
1384  001574  077777                            .WORD   077777
1385  001576  177777                            .WORD   177777
1386
1387  001600  177776                    PAT2:   .WORD   177776          ;PATTERN 2
1388  001602  177774                            .WORD   177774
1389  001604  177770                            .WORD   177770
1390  001606  177760                            .WORD   177760
1391  001610  177740                            .WORD   177740
1392  001612  177700                            .WORD   177700
1393  001614  177600                            .WORD   177600
1394  001616  177400                            .WORD   177400
1395  001620  177000                            .WORD   177000
1396  001622  176000                            .WORD   176000
1397  001624  174000                            .WORD   174000
1398  001626  170000                            .WORD   170000
1399  001630  160000                            .WORD   160000
1400  001632  140000                            .WORD   140000
1401  001634  100000                            .WORD   100000
1402  001636  000000                            .WORD   000000
1403
1404  001640  000000                    PAT3:   .WORD   000000          ;PATTERN 3
1405  001642  000000                            .WORD   000000
```

```
1406  001644  000000                      .WORD   000000
1407  001646  177777                      .WORD   177777
1408  001650  177777                      .WORD   177777
1409  001652  177777                      .WORD   177777
1410  001654  000000                      .WORD   000000
1411  001656  000000                      .WORD   000000
1412  001660  177777                      .WORD   177777
1413  001662  177777                      .WORD   177777
1414  001664  000000                      .WORD   000000
1415  001666  177777                      .WORD   177777
1416  001670  000000                      .WORD   000000
1417  001672  177777                      .WORD   177777
1418  001674  177777                      .WORD   177777
1419  001676  000000                      .WORD   000000
1420
1421  001700  000000              PAT4:   .WORD   000000          ;PATTERN 4
1422  001702  010421                      .WORD   010421
1423  001704  021042                      .WORD   021042
1424  001706  031463                      .WORD   031463
1425  001710  042104                      .WORD   042104
1426  001712  052525                      .WORD   052525
1427  001714  063146                      .WORD   063146
1428  001716  073567                      .WORD   073567
1429  001720  104210                      .WORD   104210
1430  001722  114631                      .WORD   114631
1431  001724  125252                      .WORD   125252
1432  001726  135673                      .WORD   135673
1433  001730  146314                      .WORD   146314
1434  001732  156735                      .WORD   156735
1435  001734  167356                      .WORD   167356
1436  001736  177777                      .WORD   177777
1437
1438  001740  052525              PAT5:   .WORD   052525          ;PATTERN 5
1439  001742  052525                      .WORD   052525
1440  001744  052525                      .WORD   052525
1441  001746  125252                      .WORD   125252
1442  001750  125252                      .WORD   125252
1443  001752  125252                      .WORD   125252
1444  001754  052525                      .WORD   052525
1445  001756  052525                      .WORD   052525
1446  001760  125252                      .WORD   125252
1447  001762  125252                      .WORD   125252
1448  001764  052525                      .WORD   052525
1449  001766  125252                      .WORD   125252
1450  001770  052525                      .WORD   052525
1451  001772  125252                      .WORD   125252
1452  001774  125252                      .WORD   125252
1453  001776  052525                      .WORD   052525
1454
1455  002000  007417              PAT6:   .WORD   007417          ;PATTERN 6
1456  002002  007417                      .WORD   007417
1457  002004  007417                      .WORD   007417
1458  002006  170360                      .WORD   170360
1459  002010  170360                      .WORD   170360
1460  002012  170360                      .WORD   170360
1461  002014  007417                      .WORD   007417
```

```
 1462   002016   007417                          .WORD   007417
 1463   002020   170360                          .WORD   170360
 1464   002022   170360                          .WORD   170360
 1465   002024   007417                          .WORD   007417
 1466   002026   170360                          .WORD   170360
 1467   002030   007417                          .WORD   007417
 1468   002032   170360                          .WORD   170360
 1469   002034   170360                          .WORD   170360
 1470   002036   007417                          .WORD   007417
 1471
 1472   002040   026455                  PAT7:   .WORD   026455          ;PATTERN 7
 1473   002042   026455                          .WORD   026455
 1474   002044   026455                          .WORD   026455
 1475   002046   151322                          .WORD   151322
 1476   002050   151322                          .WORD   151322
 1477   002052   151322                          .WORD   151322
 1478   002054   026455                          .WORD   026455
 1479   002056   026455                          .WORD   026455
 1480   002060   151322                          .WORD   151322
 1481   002062   151322                          .WORD   151322
 1482   002064   026455                          .WORD   026455
 1483   002066   151322                          .WORD   151322
 1484   002070   026455                          .WORD   026455
 1485   002072   151322                          .WORD   151322
 1486   002074   151322                          .WORD   151322
 1487   002076   026455                          .WORD   026455
 1488
 1489   002100   077577                  PAT8:   .WORD   077577          ;PATTERN 8 (WORST CASE PATTERN)
 1490   002102   077577                          .WORD   077577
 1491   002104   077577                          .WORD   077577
 1492   002106   077577                          .WORD   077577
 1493   002110   077577                          .WORD   077577
 1494   002112   077577                          .WORD   077577
 1495   002114   077577                          .WORD   077577
 1496   002116   077577                          .WORD   077577
 1497   002120   077577                          .WORD   077577
 1498   002122   077577                          .WORD   077577
 1499   002124   077577                          .WORD   077577
 1500   002126   077577                          .WORD   077577
 1501   002130   077577                          .WORD   077577
 1502   002132   077577                          .WORD   077577
 1503   002134   077577                          .WORD   077577
 1504   002136   077577                          .WORD   077577
 1505
 1506   002140   000001                  PAT9:   .WORD   000001          ;PATTERN 9
 1507   002142   000002                          .WORD   000002
 1508   002144   000004                          .WORD   000004
 1509   002146   000010                          .WORD   000010
 1510   002150   000020                          .WORD   000020
 1511   002152   000040                          .WORD   000040
 1512   002154   000100                          .WORD   000100
 1513   002156   000200                          .WORD   000200
 1514   002160   000400                          .WORD   000400
 1515   002162   001000                          .WORD   001000
 1516   002164   002000                          .WORD   002000
 1517   002166   004000                          .WORD   004000
```

```
 1518  002170  010000                    .WORD   010000
 1519  002172  020000                    .WORD   020000
 1520  002174  040000                    .WORD   040000
 1521  002176  100000                    .WORD   100000
 1522
 1523  002200  177776          PAT10:    .WORD   177776          ;PATTERN 10
 1524  002202  177775                    .WORD   177775
 1525  002204  177773                    .WORD   177773
 1526  002206  177767                    .WORD   177767
 1527  002210  177757                    .WORD   177757
 1528  002212  177737                    .WORD   177737
 1529  002214  177677                    .WORD   177677
 1530  002216  177577                    .WORD   177577
 1531  002220  177377                    .WORD   177377
 1532  002222  176777                    .WORD   176777
 1533  002224  175777                    .WORD   175777
 1534  002226  173777                    .WORD   173777
 1535  002230  167777                    .WORD   167777
 1536  002232  157777                    .WORD   157777
 1537  002234  137777                    .WORD   137777
 1538  002236  077777                    .WORD   077777
 1539
 1540  002240  172666          PAT11:    .WORD   172666          ;PATTERN 11
 1541  002242  155555                    .WORD   155555
 1542  002244  172666                    .WORD   172666
 1543  002246  155555                    .WORD   155555
 1544  002250  172666                    .WORD   172666
 1545  002252  155555                    .WORD   155555
 1546  002254  172666                    .WORD   172666
 1547  002256  155555                    .WORD   155555
 1548  002260  172666                    .WORD   172666
 1549  002262  155555                    .WORD   155555
 1550  002264  172666                    .WORD   172666
 1551  002266  155555                    .WORD   155555
 1552  002270  172666                    .WORD   172666
 1553  002272  155555                    .WORD   155555
 1554  002274  172666                    .WORD   172666
 1555  002276  155555                    .WORD   155555
 1556
 1557  002300  077777          PAT12:    .WORD   077777          ;PATTERN 12
 1558  002302  137777                    .WORD   137777
 1559  002304  157777                    .WORD   157777
 1560  002306  167777                    .WORD   167777
 1561  002310  173777                    .WORD   173777
 1562  002312  175777                    .WORD   175777
 1563  002314  176777                    .WORD   176777
 1564  002316  177377                    .WORD   177377
 1565  002320  177577                    .WORD   177577
 1566  002322  177677                    .WORD   177677
 1567  002324  177737                    .WORD   177737
 1568  002326  177757                    .WORD   177757
 1569  002330  177767                    .WORD   177767
 1570  002332  177773                    .WORD   177773
 1571  002334  177775                    .WORD   177775
 1572  002336  177776                    .WORD   177776
 1573
```

```
1574   002340  177777                  PAT13:  .WORD   177777          ;PATTERN 13
1575   002342  177777                          .WORD   177777
1576   002344  177777                          .WORD   177777
1577   002346  177777                          .WORD   177777
1578   002350  177777                          .WORD   177777
1579   002352  177777                          .WORD   177777
1580   002354  177777                          .WORD   177777
1581   002356  177777                          .WORD   177777
1582   002360  177777                          .WORD   177777
1583   002362  177777                          .WORD   177777
1584   002364  177777                          .WORD   177777
1585   002366  177777                          .WORD   177777
1586   002370  177777                          .WORD   177777
1587   002372  177777                          .WORD   177777
1588   002374  177777                          .WORD   177777
1589   002376  177777                          .WORD   177777
1590
1591   002400  000000                  PAT14:  .WORD   000000          ;PATTERN 14
1592   002402  000000                          .WORD   000000
1593   002404  177777                          .WORD   177777
1594   002406  177777                          .WORD   177777
1595   002410  177777                          .WORD   177777
1596   002412  177777                          .WORD   177777
1597   002414  177777                          .WORD   177777
1598   002416  177777                          .WORD   177777
1599   002420  177777                          .WORD   177777
1600   002422  177777                          .WORD   177777
1601   002424  177777                          .WORD   177777
1602   002426  177777                          .WORD   177777
1603   002430  177777                          .WORD   177777
1604   002432  177777                          .WORD   177777
1605   002434  177777                          .WORD   177777
1606   002436  177777                          .WORD   177777
1607
1608   002440  177777                  PAT15:  .WORD   177777          ;PATTERN 15
1609   002442  177777                          .WORD   177777
1610   002444  000000                          .WORD   000000
1611   002446  000000                          .WORD   000000
1612   002450  000000                          .WORD   000000
1613   002452  000000                          .WORD   000000
1614   002454  000000                          .WORD   000000
1615   002456  000000                          .WORD   000000
1616   002460  000000                          .WORD   000000
1617   002462  000000                          .WORD   000000
1618   002464  000000                          .WORD   000000
1619   002466  000000                          .WORD   000000
1620   002470  000000                          .WORD   000000
1621   002472  000000                          .WORD   000000
1622   002474  000000                          .WORD   000000
1623   002476  000000                          .WORD   000000
1624
1625                                   ;DRIVE OPERATION CONTROL BLOCK
1626
1627   002500      000                 DPB:    .BYTE   0               ;DRIVE NUMBER
1628   002501      000                         .BYTE   0               ;'MEX' BITS
1629   002502      000                         .BYTE   0               ;OPERATION CODE
```

```
1630  002503  000                .BYTE   0       ;MODE AND HDR BITS
1631  002504  000000   $WC:      .WORD   0       ;WORD COUNT
1632  002506  000000   $BUF:     .WORD   0       ;BUFFER ADDRESS
1633  002510  000000   $CYL:     .WORD   0       ;CYLINDER ADDRESS
1634  002512  000      $SEC:     .BYTE   0       ;SECTOR ADDRESS
1635  002513  000      $TRK:     .BYTE   0       ;TRACK ADDRESS
1636
```

```
                                .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


002514                          $ERRTB:
                                ;ERROR 1

002514  032173                          EM1             ;RP11 DIDN'T RESPOND TO ADDRESSING
002516  040315                          DH1
002520  043036                          DT1
002522  043516                          DF1

                                ;ERROR 2

002524  032235                          EM2             ;CAN'T START READ COMMAND
002526  040323                          DH2
002530  043040                          DT2
002532  043522                          DF2

                                ;ERROR 3

002534  032266                          EM3             ;'CLEAR' COMMAND DIDN'T TERMINATE DATA TRANSFER OPERATION
002536  040323                          DH2
002540  043040                          DT2
002542  043522                          DF2

                                ;ERROR 4

002544  032357                          EM4             ;ERROR WRITING TEST SECTOR(S)
002546  040323                          DH2
002550  043040                          DT2
002552  043522                          DF2

                                ;ERROR 5

002554  032414                          EM5             ;WRITE CHECK ERROR CHECKING TEST SECTOR(S)
002556  040465                          DH5
002560  043072                          DT5
002562  043532                          DF5

                                ;ERROR 6

002564  032466                          EM6             ;EXPECTED 'WCE' DIDN'T OCCUR
002566  040465                          DH5
002570  043072                          DT5
002572  043532                          DF5
```

```
                                ;ERROR 7
                                
002574  032522                          EM7             ;ERROR WRITING LESS THAN A FULL SECTOR
002576  040323                          DH2
002600  043040                          DT2
002602  043522                          DF2
                                
                                ;ERROR 10
                                
002604  000000                          0               ;UNUSED
002606  000000                          0
002610  000000                          0
002612  000000                          0
                                
                                ;ERROR 11
                                
002614  032570                          EM11            ;ONE OF 2 WORDS WRITTEN IN THE PARTIAL WRITE DIDN'T COMPARE
002616  040637                          DH11
002620  043126                          DT11
002622  043542                          DF11
                                
                                ;ERROR 12
                                
002624  032663                          EM12            ;NON-ZERO DATA IN PART OF SECTOR WHICH SHOULD HAVE
                                                        ;BEEN ZERO FILLED DURING PARTIAL WRITE
002626  040720                          DH12
002630  043142                          DT12
002632  043542                          DF11
                                
                                ;ERROR 13
                                
002634  033014                          EM13            ;'EOP' DIDN'T SET DURING A 2 SECTOR WRITE BEGINNING
                                                        ;AT THE LAST SECTOR OF THE PACK
002636  040323                          DH2
002640  043040                          DT2
002642  043522                          DF2
                                
                                ;ERROR 14
                                
002644  033137                          EM14            ;'ERR' DIDN'T SET WITH 'EOP'
002646  040775                          DH14
002650  043156                          DT14
002652  043542                          DF11
                                
                                ;ERROR 15
                                
002654  033173                          EM15            ;'EOP' DIDN'T CLEAR RPCA
002656  040323                          DH2
002660  043040                          DT2
002662  043522                          DF2
                                
                                ;ERROR 16
                                
002664  033223                          EM16            ;SUCA NOT CORRECT AFTER 'EOP'
002666  041052                          DH16
```

```
1749  002670  043172                    DT16
1750  002672  043616                    DF53
1751
1752                                ;ERROR 17
1753
1754  002674  033260                    EM17          ;ERROR READING TEST SECTOR
1755  002676  040323                    DH2
1756  002700  043040                    DT2
1757  002702  043522                    DF2
1758
1759                                ;ERROR 20
1760
1761  002704  033312                    EM20          ;CONTENTS OF FIRST SECTOR OF PACK CHANGED AFTER
1762                                                    ;FORCING 'EOP' DURING WRITE
1763  002706  040637                    DH11
1764  002710  043126                    DT11
1765  002712  043542                    DF11
1766
1767                                ;ERROR 21
1768
1769  002714  033425                    EM21          ;'PROG' ERROR DIDN'T SET WHEN READ COMMAND ISSUED
1770                                                    ;WHILE CONTROLLER BUSY
1771  002716  040775                    DH14
1772  002720  043156                    DT14
1773  002722  043542                    DF11
1774
1775                                ;ERROR 22
1776
1777  002724  033535                    EM22          ;'ERR' DIDN'T SET WITH 'PROG'
1778  002726  040775                    DH14
1779  002730  043156                    DT14
1780  002732  043542                    DF11
1781
1782                                ;ERROR 23
1783
1784  002734  033572                    EM23          ;ERROR ATTEMPTING TO MISFORMAT SECTOR 0
1785  002736  040323                    DH2
1786  002740  043040                    DT2
1787  002742  043522                    DF2
1788
1789                                ;ERROR 24
1790
1791  002744  033641                    EM24          ;ERROR VERIFYING THE MISFORMATTED TEST HEADER
1792  002746  040775                    DH14
1793  002750  043156                    DT14
1794  002752  043542                    DF11
1795
1796                                ;ERROR 25
1797
1798  002754  033716                    EM25          ;MISFORMATTED TEST HEADER IS NOT CORRECT
1799  002756  041167                    DH25
1800  002760  043204                    DT25
1801  002762  043546                    DF25
1802
1803                                ;ERROR 26
1804
```

```
1805  002764  033766                  EM26          ;'HNF' DIDN'T SET WHEN SEARCHING FOR MISFORMATTED SECTOR
1806  002766  040323                  DH2
1807  002770  043040                  DT2
1808  002772  043522                  DF2
1809
1810                                ;ERROR 27
1811
1812  002774  034056                  EM27          ;'DSK' NOT SET WITH 'HNF'
1813  002776  040775                  DH14
1814  003000  043156                  DT14
1815  003002  043542                  DF11
1816
1817                                ;ERROR 30
1818
1819  003004  034107                  EM30          ;'HE' DIDN'T SET WITH 'HNF'
1820  003006  040775                  DH14
1821  003010  043156                  DT14
1822  003012  043542                  DF11
1823
1824                                ;ERROR 31
1825
1826  003014  034142                  EM31          ;'ERR' NOT SET WITH 'HNF'
1827  003016  040775                  DH14
1828  003020  043156                  DT14
1829  003022  043542                  DF11
1830
1831                                ;ERROR 32
1832
1833  003024  034173                  EM32          ;ERROR WHILE RESTORING MISFORMATTED TEST HEADER
1834  003026  040323                  DH2
1835  003030  043040                  DT2
1836  003032  043522                  DF2
1837
1838                                ;ERROR 33
1839
1840  003034  034252                  EM33          ;CONTROLLER DIDN'T BECOME BUSY WHEN READ COMMAND
1841                                                ;ISSUED TO SEEKING DRIVE
1842  003036  040323                  DH2
1843  003040  043040                  DT2
1844  003042  043522                  DF2
1845
1846                                ;ERROR 34
1847
1848  003044  034363                  EM34          ;ERROR DURING READ COMMAND WHICH WAS ISSUED TO SEEKING DRIVE
1849  003046  040323                  DH2
1850  003050  043040                  DT2
1851  003052  043522                  DF2
1852
1853                                ;ERROR 35
1854
1855  003054  034457                  EM35          ;DATA INCORRECT FROM HEADER READ COMMAND ISSUED
1856                                                ;TO SEEKING DRIVE
1857  003056  041167                  DH25
1858  003060  043204                  DT25
1859  003062  043546                  DF25
1860
```

```
1861                                    ;ERROR 36
1862
1863   003064  034561                             EM36            ;'NXME' DIDN'T SET WHEN LOCATION 760000 ADDRESSED
1864   003066  040323                             DH2
1865   003070  043040                             DT2
1866   003072  043522                             DF2
1867
1868                                    ;ERROR 37
1869
1870   003074  034642                             EM37            ;'HE' DIDN'T SET WITH 'NXME'
1871   003076  040775                             DH14
1872   003100  043156                             DT14
1873   003102  043542                             DF11
1874
1875                                    ;ERROR 40
1876
1877   003104  034676                             EM40            ;'ERR' DIDN7T SET WITH 'NXME'
1878   003106  040775                             DH14
1879   003110  043156                             DT14
1880   003112  043542                             DF11
1881
1882                                    ;ERROR 41
1883
1884   003114  034733                             EM41            ;ERROR FORMATTING TRACK 0
1885   003116  040323                             DH2
1886   003120  043040                             DT2
1887   003122  043522                             DF2
1888
1889                                    ;ERROR 42
1890
1891   003124  034764                             EM42            ;ERROR READING THE HEADER FROM THE TEST SECTOR
1892   003126  040323                             DH2
1893   003130  043040                             DT2
1894   003132  043522                             DF2
1895
1896                                    ;ERROR 43
1897
1898   003134  035042                             EM43            ;SECTOR FIELD FROM HEADER IS NOT CORRECT
1899   003136  041256                             DH43
1900   003140  043224                             DT43
1901   003142  043556                             DF43
1902
1903                                    ;ERROR 44
1904
1905   003144  035112                             EM44            ;ERROR WRITING SECTOR ADDRESS IN DATA FIELD
1906   003146  040323                             DH2
1907   003150  043040                             DT2
1908   003152  043522                             DF2
1909
1910                                    ;ERROR 45
1911
1912   003154  035165                             EM45            ;'SOT' OR SECTOR ADDRESS REGISTER IS NOT CORRECT
1913   003156  041502                             DH45
1914   003160  043262                             DT45
1915   003162  043572                             DF45
1916
```

```
1917                                    ;ERROR 46
1918
1919  003164  035245                            EM46            ;DATA NOT CORRECT FOR SECTOR READ
1920  003166  041674                            DH46
1921  003170  043306                            DT46
1922  003172  043542                            DF11
1923
1924                                    ;ERROR 47
1925
1926  003174  035316                            EM47            ;SECTOR CONTENTS WRONG, DATA SHOULD BE THE
1927                                                            ;TRACK NUMBER OF THE CURRENTLY SELECTED HEAD
1928  003176  042031                            DH47
1929  003200  043322                            DT47
1930  003202  043602                            DF47
1931
1932                                    ;ERROR 50
1933
1934  003204  035445                            EM50            ;ERROR AFTER 2 WORD READ STARTING AT LOC 177776
1935  003206  040323                            DH2
1936  003210  043040                            DT2
1937  003212  043522                            DF2
1938
1939                                    ;ERROR 51
1940
1941  003214  035524                            EM51            ;'MEXO' DIDN'T SET AFTER 2 WORD READ STARTING
1942                                                            ;AT LOC 177776
1943  003216  042145                            DH51
1944  003220  043346                            DT51
1945  003222  043612                            DF51
1946
1947                                    ;ERROR 52
1948
1949  003224  035620                            EM52            ;'MEX1' SET AFTER 2 WORD READ STARTING AT LOC 177776
1950  003226  042145                            DH51
1951  003230  043346                            DT51
1952  003232  043612                            DF51
1953
1954                                    ;ERROR 53
1955
1956  003234  035704                            EM53            ;CONTENTS OF LOC 200000 WRONG AFTER 2 WORD READ
1957                                                            ;STARTING AT LOC 177776
1958  003236  041167                            DH25
1959  003240  043204                            DT25
1960  003242  043616                            DF53
1961
1962                                    ;ERROR 54
1963
1964  003244  036013                            EM54            ;ERROR AFTER 2 WORD READ STARTING AT LOC 377776
1965  003246  040323                            DH2
1966  003250  043040                            DT2
1967  003252  043522                            DF2
1968
1969                                    ;ERROR 55
1970
1971
1972  003254  036072                            EM55            ;'MEXO' DIDN'T CLEAR AFTER 2 WORD STARTING AT LOC 377776
```

```
                                        DH51
                                        DT51
                                        DF51

                                ;ERROR 56

                                        EM56            ;'MEX1' DIDN'T SET AFTER 2 WORD READ STARTING AT LOC 377776
                                        DH51
                                        DT51
                                        DF51

                                ;ERROR 57

                                        EM57            ;CONTENTS OF LOC 400000 WRONG AFTER 2 WORD READ
                                                        ;STARTING AT LOCATION 377776
                                        DH25
                                        DT25
                                        DF53

                                ;ERROR 60

                                        EM60            ;ERROR RETURNING SECTOR TO PDP-11 MODE USING A
                                                        ;2 WORD WRITE
                                        DH2
                                        DT2
                                        DF2

                                ;ERROR 61

                                        EM61            ;ERROR WRITING TEST SECTOR(S)
                                        DH5
                                        DT5
                                        DF5

                                ;ERROR 62

                                        EM62            ;ERROR READING TEST SECTOR(S)
                                        DH5
                                        DT5
                                        DF5

                                ;ERROR 63

                                        EM63            ;DRIVE DIDN'T RETURN TO CYL 0 FROM THE MAXIMUM
                                                        ;CYL ON CONTROLLER POWER FAIL
                                        DH63
                                        DT63
                                        DF63

                                ;ERROR 64

                                        EM64            ;CONTENTS OF MEMORY CHANGED DURING POWER
                                                        ;FAIL WHILE READING THE DISK
                                        DH12
                                        DT12
                                        DF11
```

```
                                ;ERROR 65
003354  037000                          EM65            ;'SUOL' SET WITH DRIVE DISABLED
003356  040775                          DH14
003360  043156                          DT14
003362  043542                          DF11

                                ;ERROR 66
003364  037037                          EM66            ;'SUOL' NOT SET WITH DRIVE ENABLED
003366  040775                          DH14
003370  043156                          DT14
003372  043542                          DF11

                                ;ERROR 67
003374  037101                          EM67            ;'SUWP' NOT SET WITH DRIVE SET TO 'READ ONLY'
003376  040775                          DH14
003400  043156                          DT14
003402  043542                          DF11

                                ;ERROR 70
003404  037156                          EM70            ;'SUWP' SET WITH DRIVE SET TO 'READ/WRITE'
003406  040775                          DH14
003410  043156                          DT14
003412  043542                          DF11

                                ;ERROR 71
003414  037230                          EM71            ;'SUWP' SET WITH RP11 'WRITE LOCKOUT' SET, CYLINDER
                                                        ;LOA'S = 0, AND RPCA = 0
003416  042327                          DM71
003420  043404                          DT71
003422  043622                          DF63

                                ;ERROR 72
003424  037344                          EM72            ;'SUWP' SET WITH RPCA = 2 & CYL LOA'S = 0
003426  042327                          DM71
003430  043404                          DT71
003432  043622                          DF63

                                ;ERROR 73
003434  037415                          EM73            ;'SUWP' NOT SET; RPCA = VALUE IN CYL LOA'S
003436  042327                          DM71
003440  043404                          DT71
003442  043622                          DF63

                                ;ERROR 74
003444  037467                          EM74            ;'SUWP' SET WITH RPCA ONE GREATER THAN CYL LOA VALUE
003446  042327                          DM71
003450  043404                          DT71
```

```
2085   003452   043622                          DF63
2086
2087                                    ;ERROR 75
2088
2089   003454   037553                          EM75            ;'SUWP' SET WITH DRIVE LOA SWITCHES EQUAL SELECTED DRIVE
2090   003456   040775                          DH14
2091   003460   043156                          DT14
2092   003462   043542                          DF11
2093
2094                                    ;ERROR 76
2095
2096   003464   037646                          EM76            ;DRIVE HAS GONE OFFLINE
2097   003466   040323                          DH2
2098   003470   043040                          DT2
2099   003472   043522                          DF2
2100
2101                                    ;ERROR 77
2102
2103   003474   037675                          EM77            ;DRIVE IS UNSAFE
2104   003476   040323                          DH2
2105   003500   043040                          DT2
2106   003502   043522                          DF2
2107
2108                                    ;ERROR 100
2109
2110   003504   037715                          EM100           ;DRIVE TIMED OUT
2111   003506   040323                          DH2
2112   003510   043040                          DT2
2113   003512   043522                          DF2
2114
2115                                    ;ERROR 101
2116
2117   003514   037735                          EM101           ;CONTROLLER TIMED OUT
2118   003516   040323                          DH2
2119   003520   043040                          DT2
2120   003522   043522                          DF2
2121
2122                                    ;ERROR 102
2123
2124   003524   037762                          EM102           ;DATA COMPARE ERROR
2125   003526   042414                          DH102
2126   003530   043422                          DT102
2127   003532   043626                          DF102
2128
2129                                    ;ERROR 103
2130
2131   003534   040005                          EM103           ;DATA COMPARE ERROR (MEMORY MANAGEMENT ENABLED)
2132   003536   042414                          DH102
2133   003540   043450                          DT103
2134   003542   043636                          DF103
2135
2136                                    ;ERROR 104
2137
2138   003544   000000                          0               ;DATA COMPARE ERROR DETAIL LINE
2139   003546   000000                          0
2140   003550   043504                          DT104
```

```
2141   003552  043652                  DF104
2142
2143                                   ;ERROR 105
2144
2145   003554  000000                  0                   ;DATA COMPARE ERROR SUMMARY LINE
2146   003556  042717                  DH105
2147   003560  043512                  DT105
2148   003562  043516                  DF1
2149
2150                                   ;ERROR 106
2151
2152   003564  040064                  EM106               ;'ERR' DIDN'T SET WITH 'WCE'
2153   003566  040775                  DH14
2154   003570  043156                  DT14
2155   003572  043542                  DF11
2156
2157                                   ;ERROR 107
2158
```

```
2159   003574   040120                  EM107           ;'HE' DIDN'T SET WITH 'PROG'
2160   003576   040775                  DH14
2161   003600   043156                  DT14
2162   003602   043542                  DF11
2163
2164                                  ;ERROR 110
2165
2166   003604   000000                  0               ;UNSUCESSFUL AFTER 3 RETRIES
2167   003606   042745                  DH110
2168   003610   000000                  0
2169   003612   000000                  0
2170
2171                                  ;ERROR 111
2172
2173   003614   000000                  0               ;SUCESSFUL AFTER 'N' RETRIES
2174   003616   043000                  DH111
2175   003620   043514                  DT111
2176   003622   043516                  DF1
2177
2178                                  ;ERROR 112
2179
2180   003624   040154                  EM112           ;DRIVE UNSAFE AFTER POWER FAIL
2181   003626   040323                  DH2
2182   003630   043040                  DT2
2183   003632   043522                  DF2
2184
2185                                  ;ERROR 113
2186
2187   003634   040212                  EM113           ;'SUWP' NOT SET WITH DRIVE LOA SWITCHES GREATER THAN SELECTED DR
2188   003636   040775                  DH14
2189   003640   043156                  DT14
2190   003642   043542                  DF11
2191
```

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

                                        .SBTTL  START OF PROGRAM

                                        ;;**********************************************************************

003644  012737  177777  001216  START3: MOV     #-1,BUSADR      ;SET BUSADR FLAG
003652  000402                   START1: BR      START1A
003654  005037  001216           START1: CLR     @#BUSADR        ;CLR BUSADR FLAG
003660  005037  001206           START1A:CLR     @#CNTRLC        ;NO CONVERSATION MODE
003664  000411                   START4: BR      START           ;
003666  012737  177777  001216  START4: MOV     #-1,BUSADR      ;SET BUSADR FLAG
003674  000402                   START2: BR      START2A
003676  005037  001216           START2: CLR     @#BUSADR        ;CLR BUSADR FLAG
003702  012737  177777  001206  START2A:MOV     #-1,CNTRLC      ;SET CONVERSATION MODE FLAG
003710  000005                   START:  RESET                   ;CLEAR THE BUS
                                        .SBTTL  INITIALIZE THE COMMON TAGS
                                        ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
003712  012706  001100                   MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
003716  005026                   1$:     CLR     (R6)+           ;;CLEAR MEMORY LOCATION
003720  022706  001140                   CMP     #SWR,R6 ;;DONE?
003724  001374                           BNE     1$-6            ;;LOOP BACK IF NO
003726  012706  001100                   MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                        ;;INITIALIZE A FEW VECTORS
003732  012737  021646  000020           MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
003740  012737  000340  000022           MOV     #340,@#IOTVEC+2 ;;LEVEL 7
003746  012737  017070  000030           MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
003754  012737  000340  000032           MOV     #340,@#EMTVEC+2 ;;LEVEL 7
003762  012737  022502  000034           MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
003770  012737  000340  000036           MOV     #340,@#TRAPVEC+2;LEVEL 7
003776  013737  017004  016776           MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
004004  005037  001162                   CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
004010  005037  001164                   CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
004014  112737  000001  001115           MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
004022  012737  004022  001106           MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
004030  012737  004030  001110           MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                        ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                        ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
004036  013746  000004                   MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
004042  012737  004076  000004           MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
004050  012737  177570  001140           MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
004056  012737  177570  001142           MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
004064  022777  177777  175046           CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
004072  001012                           BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                 ;;AND  THE HARDWARE SWR IS NOT = -1
004074  000403                           BR      65$             ;;BRANCH IF NO TIMEOUT
004076  012716  004104           64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
004102  000002                           RTI
004104  012737  000176  001140  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
004112  012737  000174  001142           MOV     #DISPREG,DISPLAY
004120  012637  000004           66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

004124  023737  000042  000046           CMP     @#42,@#46       ;ACT11 AUTO MODE?
```

```
                                                BEQ     STARTS          ;YES, SKIP TITLE PRINTOUT
                                                INC     #-1             ;FIRST START ?
                                                BNE     STARTS          ;BR IF NOT
                                                TYPE    ,TITLE          ;TYPE THE PROGRAM'S MAINDEC NUMBER AND NAME
                                STARTS: JSR     PC,$TKINT       ;SETUP THE TTY KEYBOARD
                                        .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                                                TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
                                                BNE     64$             ;;BRANCH IF YES
                                                CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
                                                BNE     65$             ;;BRANCH IF NO
                                                GTSWR                   ;;GET SOFT-SWR SETTINGS
                                                BR      65$
                                64$:            MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                                65$:
                                                JSR     PC,RESLDR       ;RESTORE THE LOADER
                                                JSR     PC,SIZMEM       ;SET MEMORY SIZE VARIABLES
                                                CLR     $ERTTL          ;CLEAR TOTAL ERROR COUNT
                                                CLR     OPRSEL          ;CLEAR OPERATOR SELECTED ADDRESS FLAG
                                                CLR     SCYL            ;OPERATOR SPECIFIED CYLINDER
                                                CLR     STRK            ;OPERATOR SPECIFIED TRACK
                                                CLR     SSEC            ;OPERATOR SPECIFIED SECTOR
                                                MOV     MAXWC,WC14      ;ASSUME THAT MAXIMUM WORD COUNT LESS THAN 8K
                                                CMP     MAXWC,LMT8K     ;MAXIMUM WORD COUNT GREATER THAN LIMIT
                                                BLOS    1$              ;BR IF NOT
                                                MOV     LMT8K,WC14      ;USE AN 8K MAXIMUM
                                1$:             MOV     #177777,PATTRN  ;ENABLE ALL PATTERNS
                                                MOV     #17777,TSTNMS   ;SELECT TESTS 0 - 14
                                                CLR     PSW             ;ENSURE THE PRIORITY = 0
                                                CLR     DRVBAD          ;CLEAR OFFLINE/UNSAFE DRIVE BITS
                                                CLR     STIMES          ;INITIALIZE NUMBER OF ITERATIONS
                                                CLR     $ESCAPE         ;CLEAR THE ESCAPE ON ERROR ADDRESS
                                                MOV     #$EOP,BYPASS    ;INITIAL BYPASS ADDRESS
                                                MOV     #1,$ICNT        ;PRESET ITERATION COUNT TO 1
                                                MOVB    #1,$ERMAX       ;ALLOW ONE ERROR PER TEST
                                                MOV     #.,$LPADR       ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
                                                MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
                                                CLR     MMACTV          ;CLEAR MEMORY MANAGEMENT FLAG
                                START6: CLR     PS              ;CLEAR THE PROCESSOR PRIORITY
                                                JSR     PC,GETADR       ;CHECK THE RP11 ADDRESS
                                                JSR     PC,RPINIT       ;FIND OUT WHICH DRIVES ARE ON SYSTEM
                                                MOV     #202.,MAXCYL    ;ASSUME RP02'S
                                                TST     DRVTYP          ;WHICH DRIVES ?
                                                BEQ     2$              ;BR IF THEY REALLY WERE RP02'S
                                                MOV     #405.,MAXCYL    ;SET MAX CYLINDER FOR RP03'S
                                2$:             INC     #-1             ;FIRST START ?
                                                BEQ     9$              ;BR IF IT IS
                                                BIT     #SW07,@SWR      ;BYPASS DRIVE STATUS TYPEOUT ?
                                                BNE     START7          ;BR IF YES
                                9$:             TYPE    ,DRSTAT         ;'DRIVE STATUS'
                                                CLR     R1              ;CLEAR TABLE POINTER
                                                MOV     #8.,R2          ;COUNTER
                                3$:
                                                MOV     R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
                                                                        ;;TYPE DRIVE NUMBER
                                                TYPOS                   ;;GO TYPE--OCTAL ASCII
                                                .BYTE   1               ;;TYPE 1 DIGIT(S)
```

```
2304  004453  000                        .BYTE   0               ;SUPPRESS LEADING ZEROS
2305  004454  105761  001370             TSTB    DRVSTA(R1)      ;CHECK DRIVE'S STATUS
2306  004460  001420                     BEQ     7$              ;DRIVE IS OFFLINE OR NON-EXISTENT
2307  004462  100403                     BMI     4$              ;DRIVE IS UNSAFE
2308  004464  104401  027262             TYPE    ,ONLINE         ;'ONLINE'
2309  004470  000402                     BR      5$              ;
2310  004472  104401  027276     4$:     TYPE    ,UNSAFE         ;'UNSAFE'
2311  004476  136137  001360  001304 5$: BITB    @TABIT(R1),DRVTYP  ;SEE WHICH DRIVE TYPE
2312  004504  001003                     BNE     6$              ;BR IF RP03
2313  004506  104401  027327             TYPE    ,RP02           ;'RP02'
2314  004512  000405                     BR      8$              ;
2315  004514  104401  027336     6$:     TYPE    ,RP03           ;'RP03'
2316  004520  000402                     BR      8$              ;
2317  004522  104401  027312     7$:     TYPE    ,OFFLIN         ;'OFFLINE'
2318  004526  104401  001173     8$:     TYPE    ,$CRLF          ;CR-LF
2319  004532  005201                     INC     R1              ;INCREMENT TABLE POINTER
2320  004534  005302                     DEC     R2              ;DECREMENT THE COUNTER
2321  004536  001343                     BNE     3$              ;CONTINUE
2322  004540  104401  001173             TYPE    ,$CRLF          ;CR-LF
2323  004544  005737  001206     START7: TST     @CNTRLC         ;CONVERSATION MODE ?
2324  004550  001403                     BEQ     1$              ;NO--BRANCH
2325  004552  004737  026314             JSR     PC,ENTPRM       ;YES--GET PARAMETERS
2326  004556  000416                     BR      5$              ;GO TYPE DRIVES TO BE TESTED
2327  004560  005037  001312     1$:     CLR     DRVSEL          ;NO DRIVES SELECTED
2328  004564  005000                     CLR     R0              ;DETERMINE THE DRIVES THAT
2329  004566  012701  000001             MOV     #1,R1           ;ARE AVAILABLE FOR TESTING
2330  004572  105760  001370     2$:     TSTB    DRVSTA(R0)      ;DRIVE ONLINE ?
2331  004576  003403                     BLE     3$              ;BR IF NOT
2332  004600  156037  001360  001312     BISB    @TABIT(R0),DRVSEL  ;SET SELECTION BIT FOR DRIVE
2333  004606  005200     3$:     INC     R0              ;INCREMENT DRIVE ADDRESS
2334  004610  106301                     ASLB    R1              ;COUNT
2335  004612  001367                     BNE     2$              ;BR IF NOT ALL DRIVES CHECKED
2336  004614  104401  027345     5$:     TYPE    ,DRVTST         ;'DRIVES TO BE TESTED'
2337  004620  005037  017004             CLR     @#NDCT          ;DETERMINE PASSES TO MAKE AND
2338  004624  005000                     CLR     R0              ;THE DRIVES TO BE TESTED
2339  004626  013701  001312             MOV     DRVSEL,R1       ;ANY DRIVES SELECTED?
2340  004632  001004                     BNE     6$              ;YES--BRANCH
2341  004634  104401  027372             TYPE    ,NONE           ;'NONE'
2342  004640  000137  016616             JMP     @#$EOP          ;GO TO END OF PROGRAM
2343  004644  005737  001206     6$:     TST     @CNTRLC         ;CONVERSATION MODE START ?
2344  004650  001011                     BNE     7$              ;BR IF NOT
2345  004652  005737  000042             TST     @42             ;UNDER MONITOR CONTROL ?
2346  004656  001406                     BEQ     7$              ;BR IF NOT
2347  004660  122737  000003  000041     CMPB    #3,@41          ;LOADED BY RP02/RP03 ?
2348  004666  001002                     BNE     7$              ;BR IF NOT
2349  004670  006201                     ASR     R1              ;EXCLUDE DRIVE 0 FROM TESTING
2350  004672  000413                     BR      8$              ;CHECK ON THE OTHER DRIVES
2351  004674  006201             7$:     ASR     R1              ;REPORT THE DRIVES TO BE TESTED
2352  004676  103011                     BCC     8$              ;BR IF DRIVE IS NOT SELECTED
2353  004700  005237  017004             INC     @#NDCT          ;GIVE THIS DRIVE A PASS
2354  004704  010046                     MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
2355                                                             ;;TYPE THE DRIVE NUMBER
2356  004706  104403             TYPOS                           ;;GO TYPE--OCTAL ASCII
2357  004710     001                     .BYTE   1               ;;TYPE 1 DIGIT(S)
2358  004711     000                     .BYTE   0               ;;SUPPRESS LEADING ZEROS
2359  004712  005701                     TST     R1              ;MORE DRIVES?
```

```
                                        BEQ     9$                      ;NO--BRANCH
                                        TYPE    ,COMMA                  ;','
                                8$:     INC     R0                      ;INCREMENT DRIVE NUMBER
                                        BR      7$                      ;CONTINUE
                                9$:     MOV     $ENDCT,$EOPCT           ;SETUP TEST COUNT
                                        CLR     CNTRLC                  ;CLEAR CONVERSATION MODE FLAG
                                RSTRT1: CLR     CHKDRV                  ;INITIALIZE THE CHECK DRIVE KEY
                                        MOV     #1,DRVMSK               ;START TO CHECK DESIRED DRIVES
                                RSTRT2: BIT     DRVMSK,DRVSEL           ;IS THIS DRIVE SELECTED?
                                RSTRT3: BNE     DRVOK                   ;YES--GO CHECK IF DRIVE IS READY FOR TESTING
                                RESTART:INC     CHKDRV                  ;MOVE  TO NEXT DRIVE NUMBER
                                        ASLB    DRVMSK                  ;POSITION THE MASK
                                        BCC     RSTRT2                  ;BR IF MORE DRIVES
                                        BIC     DRVBAD,DRVSEL           ;CLEAR SELECTION BITS FOR ANY OFFLINE/UNSAFE
                                                                        ;DRIVES
                                        BR      RSTRT1                  ;CONTINUE WITH CYCLE
                                DRVOK:  MOVB    CHKDRV,DPB              ;PICKUP THE DRIVE NUMBER
                                        TYPE    ,TSTING                 ;'TESTING WITH DRIVE '
                                        MOV     CHKDRV,-(SP)            ;;SAVE CHKDRV FOR TYPEOUT
                                                                        ;;TYPE THE DRIVE NUMBER
                                        TYPOS                           ;;GO TYPE--OCTAL ASCII
                                        .BYTE   1                       ;;TYPE 1 DIGIT(S)
                                        .BYTE   0                       ;;SUPPRESS LEADING ZEROS
                                        TYPE    ,$CRLF                  ;CR-LF
                                        MOV     RPADR,R4                ;RP11 ADDRESS
                                        JSR     PC,CLRP                 ;CLEAR THE RP11
                                        MOVB    #HOMSEK,DPB+2           ;HOME SEEK COMMAND
                                        JSR     PC,RP11                 ;START THE COMMAND
                                        JSR     PC,DRVRDY               ;WAIT FOR THE DRIVE TO FINISH
                                        JMP     TST0                    ;START THE PROGRAM

                                .SBTTL  #### TESTS ####

                                ;;*****************************************************************
                                ;*TEST 0         TEST 'CLEAR' TERMINATION

                                ;*VERIFY THAT 'CLEAR' TERMINATES AN OPERATION AND CAUSES CONTROLLER
                                ;*READY TO SET.

                                ;;*****************************************************************
                                TST0:
                                        BIT     BITS+<$TN*2>,TSTNMS     ;IS THIS TEST SELECTED
                                        BNE     .+6                     ;BR IF YES
                                        JMP     TST1                    ;GO TO THE NEXT TEST
                                        MOV     #TEST0,$LPERR           ;SETUP THE ERROR LOOP ADDRESS
                                        MOV     #TST0,$LPADR            ;SETUP THE SCOPE LOOP ADDRESS
                                        MOV     #0,$TSTNM               ;SETUP TEST NUMBER AND
                                                                        ;CLEAR THE ERROR FLAG ($ERFLG)
                                        MOV     $TSTNM,@DISPLAY         ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
                                        MOV     #EXIT0,BYPASS           ;SETUP BYPASS ADDRESS
                                        MOV     RPADR,R4                ;RP11E BUS ADDRESS
                                        CLRB    DPB+1                   ;CLEAR ANY EXTENDED MEMORY BITS
                                        CLRB    DPB+3                   ;CLEAR 'MODE' & 'HDR' BITS
                                        MOV     #100.,$TIMES            ;;DO 100. ITERATIONS
                                TEST0:  MOV     #STACK,SP               ;SETUP THE STACK POINTER
```

```
2416  005164  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
2417  005170  012737  043656  002506            MOV     #BUFFER,$BUF    ;LOAD BUFFER ADDRESS
2418  005176  012737  000001  002504            MOV     #1,$WC          ;SETUP WORD COUNT FOR 1 WORD
2419  005204  005037  002510                    CLR     $CYL            ;CYLINDER 0
2420  005210  005037  002512                    CLR     $SEC            ;TRACK & SECTOR 0
2421  005214  112737  000005  002502            MOVB    #RDSEK,DPB+2    ;READ COMMAND
2422  005222  004737  022770                    JSR     PC,RP11         ;ISSUE THE COMMAND AND RETURN
2423  005226  032764  000200  000004            BIT     #RDY,RPCS(R4)   ;IS THE CONTROLLER READY ?
2424  005234  001404                            BEQ     1$              ;BR IF NOT
2425  005236  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
2426  005242  104002                            ERROR   2               ;CAN'T START READ COMMAND
2427  005244  000420                            BR      EXITO           ;EXIT FROM TEST
2428  005246  012764  000001  000004  1$:       MOV     #CLEAR,RPCS(R4) ;ISSUE THE CONTROLLER CLEAR
2429  005254  012764  000001  000004            MOV     #CLEAR,RPCS(R4) ;CLEAR THE CONTROLLER AGAIN
2430  005262  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
2431  005266  032737  000200  001340            BIT     #RDY,$RPCS      ;IS THE CONTROLLER READY ?
2432  005274  001004                            BNE     EXITO           ;BR IF IT IS
2433  005276  104003                            ERROR   3               ;'READY' DIDN'T SET AFTER ISSUING 'CLEAR'
2434                                                                    ;DURING READ COMMAND
2435  005300  000005                            RESET                   ;FORCE THE CONTROLLER READY
2436  005302  004737  020416                    JSR     PC,$TKINT       ;SETUP THE TTY KEYBOARD
2437  005306  000004                EXITO:      SCOPE                   ;LOOP ?
2438
2439
2440                                            ;;*****************************************************************
2441                                            ;*TEST 1         WRITE CHECK TEST
2442
2443                                            ;*THIS TEST VERIFIES THE WRITE CHECK LOGIC AND TO VERIFY THAT A WRITE
2444                                            ;*CHECK ERROR CAN BE DETECTED.  'FLOATING ONES' AND 'FLOATING ZEROS'
2445                                            ;*PATTERNS ARE USED TO TEST THE WRITE CHECK COMPARE LOGIC.
2446
2447                                            ;;*****************************************************************
2448  005310                        TST1:
2449  005310  033737  001402  001276                    BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
2450  005316  001002                                    BNE     .+6                  ;BR IF YES
2451  005320  000137  005724                            JMP     TST2                 ;GO TO THE NEXT TEST
2452  005324  012737  005404  001110                    MOV     #TEST1,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
2453  005332  012737  005310  001106                    MOV     #TST1,$LPADR         ;SETUP THE SCOPE LOOP ADDRESS
2454  005340  012737  000001  001102                    MOV     #1,$TSTNM            ;SETUP TEST NUMBER AND
2455                                                                                 ;CLEAR THE ERROR FLAG ($ERFLG)
2456  005346  013777  001102  173566                    MOV     $TSTNM,@DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2457  005354  012737  005722  001200                    MOV     #EXIT1,BYPASS        ;SETUP BYPASS ADDRESS
2458  005362  013704  001324                            MOV     RPADR,R4             ;RP11E BUS ADDRESS
2459  005366  105037  002501                            CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
2460  005372  105037  002503                            CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
2461  005376  012737  000144  001162                    MOV     #100.,$TIMES         ;DO 100. ITERATIONS
2462  005404  012706  001100                TEST1:      MOV     #STACK,SP            ;SETUP THE STACK POINTER
2463  005410  005037  002512                            CLR     $SEC                 ;CLEAR TRACK & SECTOR ADDRESS
2464  005414  005037  002510                            CLR     $CYL                 ;CLEAR CYLINDER ADDRESS
2465  005420  012737  000400  002504                    MOV     #256.,$WC            ;SET WORD COUNT TO 1 SECTOR
2466  005426  012737  000011  001204                    MOV     #9.,PATNUM           ;STARTING PATTERN NUMBER
2467  005434  004737  024432                            JSR     PC,SETBUF            ;FILL THE BUFFER WITH THE PATTERN
2468  005440  012737  043656  002506                    MOV     #BUFFER,$BUF         ;SETUP OUTPUT BUFFER
2469  005446  012737  005446  001110  1$:               MOV     #1$,$LPERR           ;SETUP LOOP ON ERROR ADDRESS
2470  005454  004737  025654                            JSR     PC,CLRP              ;CLEAR THE RP11
2471  005460  012737  000003  002502                    MOV     #WRTSEK,DPB+2        ;LOAD COMMAND
```

```
2472  005466  004737  022770                    JSR     PC,RP11         ;WRITE THE SECTOR WITH THE TEST PATTERN
2473  005472  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE OPERATION TO COMPLETE
2474  005476  004737  023240                    JSR     PC,SAVRP        ;SAVE THE REGISTERS
2475  005502  032737  100000  001340            BIT     #ERR,$RPCS      ;DID AN ERROR OCCUR ?
2476  005510  001401                            BEQ     3$              ;BR IF NOT
2477  005512  104004                            ERROR   4               ;ERROR OCCURED WRITING WRITE CHECK TEST PATTERN
2478  005514  012737  005514  001110  3$:       MOV     #3$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
2479  005522  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
2480  005526  012737  000007  002502            MOV     #WRTCHK,DPB+2   ;LOAD WRITE CHECK COMMAND
2481  005534  004737  022770                    JSR     PC,RP11         ;CHECK THE SECTOR JUST WRITTEN
2482  005540  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR WRITE CHECK TO COMPLETE
2483  005544  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
2484  005550  032737  100000  001340            BIT     #ERR,$RPCS      ;CHECK OK ?
2485  005556  001403                            BEQ     4$              ;BR IF OK
2486  005560  104005                            ERROR   5               ;ERROR ATTEMPTING TO CHECK TEST PATTERN
2487  005562  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
2488  005566  005001                  4$:       CLR     R1              ;SETUP TO CLEAR BUFFER
2489  005570  005002                            CLR     R2              ;BUFFER INDEX
2490  005572  023727  001204  000011            CMP     PATNUM,#9.      ;'FLOATING 1' PATTERN ?
2491  005600  001002                            BNE     5$              ;BRANCH IF NOT
2492  005602  012701  177777                    MOV     #-1,R1          ;FILL BUFFER WITH ONES
2493  005606  010162  043656          5$:       MOV     R1,BUFFER(R2)   ;FILL BUFFER WITH PATTERN IN R1
2494  005612  022702  000400                    CMP     #256.,R2        ;SEE IF FINISHED
2495  005616  001403                            BEQ     6$              ;BR IF FINISHED
2496  005620  062702  000002                    ADD     #2,R2           ;INCREMENT THE INDEX
2497  005624  000770                            BR      5$              ;CONTINUE
2498  005626  012737  005626  001110  6$:       MOV     #6$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
2499  005634  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
2500  005640  004737  022770                    JSR     PC,RP11         ;CHECK THE SECTOR AGAIN, ERROR SHOULD OCCUR
2501  005644  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE WRITE CHECK TO COMPLETE
2502  005650  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
2503  005654  032737  000010  001336            BIT     #BIT03,$RPER    ;DID WRITE CHECK ERROR SET?
2504  005662  001002                            BNE     7$              ;BRANCH IF YES
2505  005664  104006                            ERROR   6               ;WRITE CHECK ERROR DID NOT SET
2506  005666  000405                            BR      8$              ;BYPASS REST OF TEST
2507  005670  032737  100000  001340  7$:       BIT     #ERR,$RPCS      ;DID 'ERR' SET ?
2508  005676  001001                            BNE     8$              ;BR IF IT SET
2509  005700  104106                            ERROR   106             ;'ERR' DIDN'T SET WITH 'WCE'
2510  005702  023727  001204  000012  8$:       CMP     PATNUM,#10.     ;FLOATING A ZERO ?
2511  005710  001404                            BEQ     EXIT1           ;BR IF ZERO
2512  005712  012737  000012  001204            MOV     #10.,PATNUM     ;'FLOATING ZERO' PATTERN
2513  005720  000652                            BR      1$              ;DO THE TEST AGAIN
2514  005722  000004                  EXIT1:    SCOPE                   ;LOOP ?
2515
2516
2517                                            ;;*****************************************************************
2518                                            ;*TEST 2         PARTIAL SECTOR WRITE TEST
2519
2520                                            ;*CHECK THE ABILITY OF THE RP11E TO CLEAR THE REMAINDER OF A SECTOR
2521                                            ;*ON A PARTIAL WRITE OPERATION. A SECTOR OF ALL ONES IS WRITTEN AND
2522                                            ;*THEN A TWO WORD WRITE OPERATION IS PERFORMED. THE SECTOR IS THEN
2523                                            ;*READ BACK AND VERIFIED. THE FIRST TWO WORDS SHOULD BE ONES AND
2524                                            ;*THE REST SHOULD BE ZEROS.
2525
2526                                            ;;*****************************************************************
2527  005724                                    ;ST2:
```

```
2528  005724  033737  001404  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
2529  005732  001002                          BNE     .+6                  ;BR IF YES
2530  005734  000137  006402                  JMP     TST3                 ;GO TO THE NEXT TEST
2531  005740  012737  006020  001110          MOV     #TEST2,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
2532  005746  012737  005724  001106          MOV     #TST2,$LPADR         ;SETUP THE SCOPE LOOP ADDRESS
2533  005754  012737  000002  001102          MOV     #2,$TSTNM            ;SETUP TEST NUMBER AND
2534                                                                       ;CLEAR THE ERROR FLAG ($ERFLG)
2535  005762  013777  001102  173152          MOV     $TSTNM,@DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2536  005770  012737  006400  001200          MOV     #EXIT2,BYPASS        ;SETUP BYPASS ADDRESS
2537  005776  013704  001324                  MOV     RPADR,R4             ;RP11E BUS ADDRESS
2538  006002  105037  002501                  CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
2539  006006  105037  002503                  CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
2540  006012  012737  000144  001162          MOV     #100.,$TIMES         ;DO 100. ITERATIONS
2541  006020  012706  001100          TEST2:  MOV     #STACK,SP            ;SETUP THE STACK POINTER
2542  006024  012737  006020  001110          MOV     #TEST2,$LPERR        ;SETUP INITIAL LOOP ON ERROR ADDRESS
2543  006032  004737  025654                  JSR     PC,CLRP              ;CLEAR THE RP11
2544  006036  012737  043656  002506          MOV     #BUFFER,$BUF         ;LOAD BUFFER ADDRESS
2545  006044  012737  000400  002504          MOV     #256.,$WC            ;WORD COUNT = 1 SECTOR
2546  006052  005037  002512                  CLR     $SEC                 ;CLEAR SECTOR/TRACK ADDRESS
2547  006056  005037  002510                  CLR     $CYL                 ;CLEAR CYLINDER ADDRESS
2548  006062  012737  000015  001204          MOV     #13.,PATNUM          ;ONE'S PATTERN NUMBER
2549  006070  004737  024432                  JSR     PC,SETBUF            ;LOAD THE PATTERN INTO THE BUFFER
2550  006074  012737  000003  002502          MOV     #WRTSEK,DPB+2        ;WRITE COMMAND
2551  006102  004737  022770                  JSR     PC,RP11              ;WRITE THE SECTOR
2552  006106  004737  023664                  JSR     PC,CONRDY            ;WAIT FOR THE WRITE TO COMPLETE
2553  006112  004737  023240                  JSR     PC,SAVRP             ;STORE THE REGISTERS
2554  006116  032737  100000  001340          BIT     #ERR,$RPCS           ;DID AN ERROR OCCUR ?
2555  006124  001402                          BEQ     1$                   ;BR IF NOT
2556  006126  104004                          ERROR   4                    ;ERROR OCCURED DURING WRITE
2557  006130  000523                          BR      EXIT2                ;BYPASS REST OF TEST
2558  006132  012737  006132  001110  1$:     MOV     #1$,$LPERR           ;CHANGE LOOP ON ERROR ADDRESS
2559  006140  004737  025654                  JSR     PC,CLRP              ;CLEAR THE RP11
2560  006144  004737  024432                  JSR     PC,SETBUF            ;LOAD ONE'S PATTERN INTO THE BUFFER
2561  006150  012737  000002  002504          MOV     #2,$WC               ;2 WORD TRANSFER
2562  006156  112737  000003  002502          MOVB    #WRTSEK,DPB+2        ;CHANGE COMMAND TO WRITE
2563  006164  004737  022770                  JSR     PC,RP11              ;WRITE
2564  006170  004737  023664                  JSR     PC,CONRDY            ;WAIT FOR CONTROLLER READY
2565  006174  004737  023240                  JSR     PC,SAVRP             ;STORE THE REGISTERS
2566  006200  032737  100000  001340          BIT     #ERR,$RPCS           ;ERROR ?
2567  006206  001402                          BEQ     2$                   ;BR IF NOT
2568  006210  104007                          ERROR   7                    ;ERROR WRITING THE PARTIAL SECTOR
2569  006212  000472                          BR      EXIT2                ;BYPASS THE REST OF THE TEST
2570  006214  012737  006214  001110  2$:     MOV     #2$,$LPERR           ;CHANGE THE ERROR LOOP ADDRESS
2571  006222  004737  025654                  JSR     PC,CLRP              ;CLEAR THE RP11
2572  006226  012737  000400  002504          MOV     #256.,$WC            ;CHANGE THE WORD COUNT
2573  006234  012737  000005  002502          MOV     #RDSEK,DPB+2         ;READ COMMAND
2574  006242  004737  022770                  JSR     PC,RP11              ;READ THE SECTOR
2575  006246  004737  023664                  JSR     PC,CONRDY            ;WAIT FOR CONTROLLER READY
2576  006252  004737  023240                  JSR     PC,SAVRP             ;STORE THE REGISTERS
2577  006256  032737  100000  001340          BIT     #ERR,$RPCS           ;DID AN ERROR OCCUR ?
2578  006264  001402                          BEQ     3$                   ;BR IF NOT
2579  006266  104017                          ERROR   17                   ;ERROR OCCURED READING SECTOR
2580  006270  000443                          BR      EXIT2                ;BYPASS REST OF TEST
2581  006272  022737  177777  043656  3$:     CMP     #177777,BUFFER       ;COMPARE FIRST WORD SHOULD BE ONES
2582  006300  001004                          BNE     4$                   ;BR IF NOT
2583  006302  022737  177777  043660          CMP     #177777,BUFFER+2     ;CHECK THE SECOND WORD
```

```
 2584  006310  001407                        BEQ     5$                      ;BR IF OK
 2585  006312  012737  177777  001124  4$:   MOV     #177777,$GDDAT          ;EXPECTED DATA
 2586  006320  013737  043656  001126        MOV     BUFFER,$BDDAT           ;ACTUAL DATA
 2587  006326  104011                        ERROR   11                      ;DATA COMPARE ERROR IN FIRST 2 WORDS
 2588  006330  012700  043662          5$:   MOV     #BUFFER+4,R0            ;STARTING ADDRESS OF LAST PART OF THE BUFFER
 2589  006334  012701  000376                MOV     #254.,R1                ;WORD COUNT
 2590  006340  005720                  6$:   TST     (R0)+                   ;REMAINDER OF SECTOR SHOULD BE ZEROS
 2591  006342  001003                        BNE     7$                      ;BRANCH IF NOT
 2592  006344  005301                        DEC     R1                      ;DECREMENT THE CUNT
 2593  006346  001374                        BNE     6$                      ;BR IF NOT FINISHED
 2594  006350  000413                        BR      EXIT2                   ;EXIT
 2595  006352  016037  177776  001126  7$:   MOV     -2(R0),$BDDAT           ;INCORRECT DATA
 2596  006360  005037  001124                CLR     $GDDAT                  ;EXPECTED DATA (ZEROS)
 2597  006364  010037  001122                MOV     R0,$BDADR               ;ADDRESS OF INCORRECT CHARACTER
 2598  006370  162737  000002  001122        SUB     #2,$BDADR               ;DECREMENT THE ADDRESS
 2599  006376  104012                        ERROR   12                      ;DATA FOUND IN AREA OF SECTOR
 2600                                                                        ;WHICH SHOULD HAVE BEEN CLEARED
 2601                                                                        ;BY A ONE WORD WRITE
 2602  006400  000004                  EXIT2:  SCOPE
 2603
 2604
 2605                                  ;;*****************************************************************
 2606                                  ;*TEST 3          'EOP' TEST
 2607
 2608                                  ;*TEST THAT 'EOP' SETS WHEN THE CONTROLLER TRIES TO WRITE BEYOND THE
 2609                                  ;*LIMITS OF THE DRIVE.  THE FIRST SECTOR OF THE PACK IS WRITTEN WITH
 2610                                  ;*ZEROS. THEN A TWO SECTOR WRITE OF ALL ONE'S IS ISSUED FOR THE MAXIMUM
 2611                                  ;*CYLINDER, HEAD 19, SECTOR 9.  'EOP' AND THE ERROR BITS IN 'RPCS' SHOULD
 2612                                  ;*BE SET.  THE FIRST SECTOR OF THE PACK IS CHECKED TO MAKE SURE THAT IS
 2613                                  ;*STILL CONTAINS ZEROS.
 2614
 2615                                  ;;*****************************************************************
 2616  006402                          TST3:
 2617  006402  033737  001406  001276        BIT     BITS+<$TN*2>,TSTNMS     ;IS THIS TEST SELECTED
 2618  006410  001002                        BNE     .+6                     ;BR IF YES
 2619  006412  000137  007060                JMP     TST4                    ;GO TO THE NEXT TEST
 2620  006416  012737  006476  001110        MOV     #TEST3,$LPERR           ;SETUP THE ERROR LOOP ADDRESS
 2621  006424  012737  006402  001106        MOV     #TST3,$LPADR            ;SETUP THE SCOPE LOOP ADDRESS
 2622  006432  012737  000003  001102        MOV     #3,$TSTNM               ;SETUP TEST NUMBER AND
 2623                                                                        ;CLEAR THE ERROR FLAG ($ERFLG)
 2624  006440  013777  001102  172474        MOV     $TSTNM,@DISPLAY         ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
 2625  006446  012737  007056  001200        MOV     #EXIT3,BYPASS           ;SETUP BYPASS ADDRESS
 2626  006454  013704  001324                MOV     RPADR,R4                ;RP11E BUS ADDRESS
 2627  006460  105037  002501                CLRB    DPB+1                   ;CLEAR ANY EXTENDED MEMORY BITS
 2628  006464  105037  002503                CLRB    DPB+3                   ;CLEAR 'MODE' & 'HDR' BITS
 2629  006470  012737  000031  001162        MOV     #25.,$TIMES             ;;DO 25. ITERATIONS
 2630  006476  012706  001100          TEST3:MOV     #STACK,SP               ;SETUP THE STACK POINTER
 2631  006502  012737  006476  001110        MOV     #TEST3,$LPERR           ;INITIAL LOOP ON ERRORR ADDRESS
 2632  006510  004737  025654                JSR     PC,CLRP                 ;CLEAR THE RP11
 2633  006514  012737  000400  002504        MOV     #256.,$WC               ;SET WORD COUNT TO 1 SECTOR
 2634  006522  012737  043656  002506        MOV     #BUFFER,$BUF            ;SETUP OUTPUT BUFFER
 2635  006530  005037  001204                CLR     PATNUM                  ;SETUP FOR PATTERN ZERO
 2636  006534  004737  024432                JSR     PC,SETBUF               ;LOAD ZEROS INTO THE BUFFER
 2637  006540  005037  002512                CLR     $SEC                    ;SET SECTOR & TRACK TO ZERO
 2638  006544  005037  002510                CLR     $CYL                    ;SET CYLINDER TO ZERO
 2639  006550  112737  000003  002502        MOVB    #WRTSEK,DPB+2           ;WRITE COMMAND
```

```
2640  006556  004737  022770                  JSR     PC,RP11         ;WRITE ZEROS IN CO, TO, SO
2641  006562  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
2642  006566  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
2643  006572  032737  100000  001340          BIT     #ERR,$RPCS      ;DID AN ERROR OCCUR ?
2644  006600  001402                          BEQ     1$              ;BR IF NOT
2645  006602  104004                          ERROR   4               ;ERROR OCCURED DURING SETUP
2646  006604  000524                          BR      EXIT3           ;BYPASS REST OF TEST
2647  006606  012737  006606  001110  1$:     MOV     #1$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
2648  006614  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
2649  006620  012737  000015  001204          MOV     #13.,PATNUM     ;ONE'S PATTERN INDEX
2650  006626  004737  024432                  JSR     PC,SETBUF       ;LOAD THE PATTERN INTO THE BUFFER
2651  006632  012737  001000  002504          MOV     #512.,$WC       ;CHANGE WORD COUNT TO 2 SECTORS
2652  006640  013737  001210  002510          MOV     MAXCYL,$CYL     ;SELECT THE DRIVE'S MAXIMUM CYLINDER
2653  006646  012737  000011  002512          MOV     #9.,$SEC        ;SELECT SECTOR 9
2654  006654  112737  000023  002513          MOVB    #19.,$TRK       ;SELECT HEAD 19
2655  006662  004737  022770                  JSR     PC,RP11         ;START THE READ
2656  006666  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
2657  006672  004737  023240                  JSR     PC,SAVRP        ;STORE THE RP11 REGISTERS
2658  006676  032737  000002  001336          BIT     #BIT01,$RPER    ;DID EOP ERROR SET?
2659  006704  001002                          BNE     2$              ;BRANCH IF SET
2660  006706  104013                          ERROR   13              ;'EOP' DIDN'T SET
2661  006710  000405                          BR      3$              ;CHECK RPCA
2662  006712  032737  100000  001340  2$:     BIT     #ERR,$RPCS      ;DID THE ERROR FLAG SET?
2663  006720  001001                          BNE     3$              ;BRANCH IF SET
2664  006722  104014                          ERROR   14              ;ERROR DID NOT SET AFTER GENERATING 'EOP'
2665  006724  005737  001346          3$:     TST     $RPCA           ;SEE IF THE CYLINDER ADDRESS REGISTER WAS CLEARED
2666  006730  001401                          BEQ     4$              ;BR IF CLEARED
2667  006732  104015                          ERROR   15              ;'RPCA' NOT CLEARED BY 'EOP'
2668  006734  023737  001354  001210  4$:     CMP     $SUCA,MAXCYL    ;SUCA STILL EQUAL TO THE MAXIMUM CLINDER
2669  006742  001401                          BEQ     5$              ;BR IF IT IS
2670  006744  104016                          ERROR   16              ;SUCA NOT CORRECT AFTER EOP ERROR
2671  006746  012737  006746  001110  5$:     MOV     #5$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
2672  006754  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
2673  006760  012737  000002  002504          MOV     #2,$WC          ;WORD COUNT
2674  006766  005037  002512                  CLR     $SEC            ;CLEAR SECTOR/TRACK ADDRESS
2675  006772  005037  002510                  CLR     $CYL            ;CLEAR THE CYLINDER ADDRESS
2676  006776  012737  000005  002502          MOV     #RDSEK,DPB+2    ;CHANGE COMMAND TO READ
2677  007004  004737  022770                  JSR     PC,RP11         ;EXECUTE THE COMMAND
2678  007010  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
2679  007014  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
2680  007020  032737  100000  001340          BIT     #ERR,$RPCS      ;WERE THERE ANY ERRORS?
2681  007026  001402                          BEQ     6$              ;BRANCH IF NOT
2682  007030  104017                          ERROR   17              ;ERROR ENCOUNTERED ON 2 WORD READ
2683  007032  000411                          BR      EXIT3           ;OF FIRST SECTOR ON THE PACK
2684  007034  013737  043656  001126  6$:     MOV     BUFFER,$BDDAT   ;GET FIRST WORD OF BUFFER
2685  007042  005737  001126                  TST     $BDDAT          ;DOES 1ST SECTOR STILL CONTAIN ZEROS?
2686  007046  001403                          BEQ     EXIT3           ;BRANCH IF YES
2687  007050  005037  001124                  CLR     $GDDAT          ;ZEROS WERE EXPECTED
2688  007054  104020                          ERROR   20              ;CONTENTS OF THE FIRST SECTOR OF THE
2689                                                                  ;PACK CHANGED AFTER FORCING EOP
2690  007056  000004                  EXIT3:  SCOPE
2691
2692
2693                                  ;;*****************************************************************
2694                                  ;*TEST 4          TEST 'PROG' ERROR
2695
```

```
                                ;*VERIFY THAT THE RP11E GENERATES A 'PROG' ERROR IF A COMMAND IS ISSUED
                                ;*WHILE THE CONTROLLER IS BUSY.

                                ;;*****************************************************************
007060                          ↑ST4:
007060  033737  001410  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
007066  001002                          BNE     .+6                  ;BR IF YES
007070  000137  007310                  JMP     ↑ST5                 ;GO TO THE NEXT TEST
007074  012737  007154  001110          MOV     #TEST4,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
007102  012737  007060  001106          MOV     #ST4,$LPADR          ;SETUP THE SCOPE LOOP ADDRESS
007110  012737  000004  001102          MOV     #4,$TSTNM            ;SETUP TEST NUMBER AND
                                                                     ;CLEAR THE ERROR FLAG ($ERFLG)
007116  013777  001102  172016          MOV     $TSTNM,@DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
007124  012737  007306  001200          MOV     #EXIT4,BYPASS        ;SETUP BYPASS ADDRESS
007132  013704  001324                  MOV     RPADR,R4             ;RP11E BUS ADDRESS
007136  105037  002501                  CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
007142  105037  002503                  CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
007146  012737  000144  001162          MOV     #100.,$TIMES         ;DO 100. ITERATIONS
007154  012706  001100          TEST4:  MOV     #$STACK,SP           ;SETUP THE STACK POINTER
007160  004737  025654                  JSR     PC,CLRP              ;CLEAR THE RP11
007164  005037  002512                  CLR     $SEC                 ;CLEAR THE SECTOR - TRACK ADDRESSES
007170  005037  002510                  CLR     $CYL                 ;CLEAR THE CYLINDER ADDRESS
007174  012737  043656  002506          MOV     #BUFFER,$BUF         ;SETUP BUFFER ADDRESS
007202  012737  002000  002504          MOV     #1024.,$WC           ;SETUP WORD COUNT
007210  112737  000005  002502          MOVB    #RDSEK,DPB+2         ;READ COMMAND
007216  004737  022770                  JSR     PC,RP11              ;ISSUE THE COMMAND
007222  012737  000001  001220          MOV     #1,STALLT            ;LOAD STALL VALUE
007230  004737  023202                  JSR     PC,STALL             ;STALL FOR 1 MILLISECONDS
007234  112764  000005  000004          MOVB    #RDSEK,RPCS(R4)      ;ISSUE READ COMMAND WHILE BUSY
007242  004737  023240                  JSR     PC,SAVRP             ;STORE THE REGISTERS
007246  032737  002000  001336          BIT     #BIT10,$RPER         ;DID PROGRAM ERROR SET?
007254  001002                          BNE     1$                   ;BRANCH IF SET
007256  104021                          ERROR   21                   ;PROGRAM ERROR DID NOT SET WHEN A
                                                                     ;READ COMMAND WAS ISSUED WHILE
                                                                     ;THE DEVICE WAS BUSY
007260  000412                          BR      EXIT4                ;BYPASS REST OF TEST
007262  032737  040000  001340  1$:     BIT     #HE,$RPCS            ;'HE' SET ?
007270  001001                          BNE     2$                   ;BR IF NOT
007272  104107                          ERROR   107                  ;'HE' DIDN'T SET WITH 'PROG'
007274  032737  100000  001340  2$:     BIT     #ERR,$RPCS           ;DID 'ERR' SET ?
007302  001001                          BNE     EXIT4                ;BRANCH IF SET
007304  104022                          ERROR   22                   ;'ERR' DID NOT SET WITH PROGRAM ERROR
007306  000004                  EXIT4:  SCOPE

                                ;;*****************************************************************
                                ;*TEST 5          'HEADER NOT FOUND' TEST

                                ;*UNFORMAT THE FIRST SECTOR ON THE PACK AND THEN READ IT BACK.  VERIFY
                                ;*THAT READ AND WRITE HEADER OPERATIONS WILL TRANSFER DATA CORRECTLY.
                                ;*ISSUE A WRITE COMMAND TO SECTOR ZERO; THIS SHOULD CAUSE 'HEADER NOT
                                ;*FOUND' TO SET.  REFORMAT THE SECTOR.

                                ;;*****************************************************************
007310                          ↑ST5:
007310  033737  001412  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
```

```
2752  007316  001002                          BNE     .+6                     ;BR IF YES
2753  007320  000137  010034                  JMP     @T5T6                   ;GO TO THE NEXT TEST
2754  007324  012737  007404  001110          MOV     #TEST5,$LPERR           ;SETUP THE ERROR LOOP ADDRESS
2755  007332  012737  007310  001106          MOV     #T5T5,$LPADR            ;SETUP THE SCOPE LOOP ADDRESS
2756  007340  012737  000005  001102          MOV     #5,$TSTNM               ;SETUP TEST NUMBER AND
2757                                                                          ;CLEAR THE ERROR FLAG ($ERFLG)
2758  007346  013777  001102  171566          MOV     $TSTNM,@DISPLAY         ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2759  007354  012737  010032  001200          MOV     #EXIT5,BYPASS   ;SETUP BYPASS ADDRESS
2760  007362  013704  001324                  MOV     RPADR,R4                ;RP11E BUS ADDRESS
2761  007366  105037  002501                  CLRB    DPB+1                   ;CLEAR ANY EXTENDED MEMORY BITS
2762  007372  105037  002503                  CLRB    DPB+3                   ;CLEAR 'MODE' & 'HDR' BITS
2763  007376  012737  000144  001162          MOV     #100.,$TIMES            ;DO 100. ITERATIONS
2764  007404  012706  001100          TEST5:  MOV     #STACK,SP               ;SETUP THE STACK POINTER
2765  007410  012737  007404  001110          MOV     #TEST5,$LPERR           ;INITIAL LOOP ON ERROR ADDRESS
2766  007416  004737  025654                  JSR     PC,CLRP                 ;CLEAR THE RP11
2767  007422  012737  000001  043656          MOV     #1.BUFFER               ;SETUP INVALID HEADER IMAGE
2768  007430  012737  000001  043660          MOV     #1.BUFFER+2             ;CYLINDER/TRACK
2769  007436  012737  000001  043662          MOV     #1.BUFFER+4             ;SECTOR
2770  007444  012737  000003  002504          MOV     #3,$WC                  ;LOAD WORD COUNT
2771  007452  005037  002512                  CLR     $SEC                    ;CLEAR SECTOR/TRACK ADDRESS
2772  007456  005037  002510                  CLR     $CYL                    ;CLEAR THE CYLINDER ADDRESS
2773  007462  012737  014000  002502          MOV     #MODE!HDR,DPB+2         ;'MODE' AND 'HEADER' BITS
2774  007470  112737  000003  002502          MOVB    #WRTSEK,DPB+2           ;WRITE ORDER
2775  007476  004737  022770                  JSR     PC,RP11                 ;MISFORMAT SECTOR 0
2776  007502  004737  023664                  JSR     PC,CONRDY               ;WAIT FOR THE OPERATION TO COMPLETE
2777  007506  004737  023240                  JSR     PC,SAVRP                ;STORE THE REGISTERS
2778  007512  032737  100000  001340          BIT     #ERR,$RPCS              ;DID AN ERROR OCCUR ?
2779  007520  001402                          BEQ     1$                      ;BR IF NOT
2780  007522  104023                          ERROR   23                      ;ERROR OCCURED ATTEMPTING TO MISFORMAT SECTOR 0
2781  007524  000506                          BR      T5T5B                   ;BYPASS TEST & REFORMAT SECTOR 0
2782  007526  012737  007526  001110  1$:     MOV     #1$,$LPERR              ;CHANGE LOOP ON ERROR ADDRESS
2783  007534  004737  025654                  JSR     PC,CLRP                 ;CLEAR THE RP11
2784  007540  112737  000005  002502          MOVB    #RDSEK,DPB+2            ;CHANGE COMMAND
2785  007546  004737  022770                  JSR     PC,RP11                 ;EXECUTE THE COMMAND
2786  007552  004737  023664                  JSR     PC,CONRDY               ;WAIT FOR CONTROLLER READY
2787  007556  032737  100000  001340          BIT     #ERR,$RPCS              ;ANY ERRORS?
2788  007564  001401                          BEQ     2$                      ;BRANCH IF NOT
2789  007566  104024                          ERROR   24                      ;ERROR WHILE READING THE HEADER
2790                                                                          ;ON SECTOR ZERO
2791  007570  012737  000001  001124  2$:     MOV     #1,$GDDAT               ;EXPECTED RESULT
2792  007576  005000                          CLR     R0                      ;CLEAR THE INDEX
2793  007600  023760  001124  043656  3$:     CMP     $GDDAT,BUFFER(R0)       ;CHECK DATA READ BACK
2794  007606  001006                          BNE     4$                      ;BRANCH ON NON COMPARE
2795  007610  062700  000002                  ADD     #2,R0                   ;UPDATE MODIFIER
2796  007614  022700  000006                  CMP     #6,R0                   ;END OF BUFFER?
2797  007620  001367                          BNE     3$                      ;BR IF NOT
2798  007622  000404                          BR      T5T5A                   ;CONTINUE WITH TEST
2799  007624  016037  043656  001126  4$:     MOV     BUFFER(R0),$BDDAT       ;GET BAD DATA
2800  007632  104025                          ERROR   25                      ;DATA DID NOT VERIFY AFTER READING
2801                                                                          ;THE HEADER OF SECTOR ZERO
2802  007634  012737  007634  001110  T5T5A:  MOV     #T5T5A,$LPERR           ;CHANGE LOOP ON ERROR ADDRESS
2803  007642  004737  025654                  JSR     PC,CLRP                 ;CLEAR THE RP11
2804  007646  012737  000003  002502          MOV     #WRTSEK,DPB+2           ;CHANGE COMMAND (AND CLEAR 'MODE' & 'HDR')
2805  007654  004737  022770                  JSR     PC,RP11                 ;EXECUTE THE COMMAND
2806  007660  004737  023664                  JSR     PC,CONRDY               ;WAIT FOR CONTROLLER READY
2807  007664  004737  023240                  JSR     PC,SAVRP                ;STORE THE REGISTERS
```

```
2808  007670  032737  010000  001334  1$:     BIT     #BIT12,$RPDS    ;DID HEADER NOT FOUND SET?
2809  007676  001002                          BNE     2$              ;BRANCH IF YES
2810  007700  104026                          ERROR   26              ;'HNF' DID NOT SET
2811  007702  000417                          BR      TST5B           ;BYPASS REST OF TEST
2812  007704  032737  000001  001336  2$:     BIT     #BIT00,$RPER    ;DID DISK ERROR SET?
2813  007712  001001                          BNE     3$              ;BRANCH IF YES
2814  007714  104027                          ERROR   27              ;'DSK' DID NOT SET WITH 'HNF'
2815  007716  032737  040000  001340  3$:     BIT     #HE,$RPCS       ;DID HARD ERROR SET?
2816  007724  001001                          BNE     4$              ;BRANCH IF YES
2817  007726  104030                          ERROR   30              ;HARD ERROR NOT SET AFTER 'HNF'
2818  007730  032737  100000  001340  4$:     BIT     #ERR,$RPCS      ;DID 'ERR' SET ?
2819  007736  001001                          BNE     TST5B           ;BRANCH IF YES
2820  007740  104031                          ERROR   31              ;'ERR' DID NOT SET WITH 'HNF'
2821  007742  012737  007742  001110  TST5B:  MOV     #TST5B,$LPERR   ;CHANGE LOOP ON ERROR ADDRESS
2822  007750  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
2823  007754  005037  043656                  CLR     BUFFER          ;SETUP HEADER IMAGE
2824  007760  005037  043660                  CLR     BUFFER+2        ;CYLINDER/TRACK = 0
2825  007764  005037  043662                  CLR     BUFFER+4        ;SECTOR = 0
2826  007770  012737  014000  002502          MOV     #MODE!HDR,DPB+2 ;'MODE' AND 'HDR' BITS
2827  007776  112737  000003  002502          MOVB    #WRTSEK,DPB+2   ;WRITE COMMAND
2828  010004  004737  022770                  JSR     PC,RP11         ;RESTORE THE HEADER
2829  010010  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR THE OPERATION TO COMPLETE
2830  010014  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
2831  010020  032737  100000  001340          BIT     #ERR,$RPCS      ;DID AN ERROR OCCUR ?
2832  010026  001557                          BEQ     EXIT6           ;BR IF NOT
2833  010030  104032                          ERROR   32              ;ERROR ATTEMPTING TO RESTORE HEADER
2834  010032  000004                  EXIT5:  SCOPE
2835
2836
2837                                  ;;*****************************************************************
2838                                  ;*TEST 6          TEST COMMAND ISSUED TO A SEEKING DRIVE
2839
2840                                  ;*ISSUE A SEEK COMMAND AND WAIT FOR DONE TO SET. THEN ISSUE A READ
2841                                  ;*COMMAND WHILE THE HEADS ARE STILL MOVING. THE RP11E SHOULD
2842                                  ;*HOLD THE WRITE COMMAND TILL THE SEEK IS COMPLETE.
2843
2844                                  ;;*****************************************************************
2845  010034
2846  010034  033737  001414  001276  TST6:   BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
2847  010042  001002                          BNE     .+6             ;BR IF YES
2848  010044  000137  010370                  JMP     TST7            ;GO TO THE NEXT TEST
2849  010050  012737  010130  001110          MOV     #TEST6,$LPERR   ;SETUP THE ERROR LOOP ADDRESS
2850  010056  012737  010034  001106          MOV     #TST6,$LPADR    ;SETUP THE SCOPE LOOP ADDRESS
2851  010064  012737  000006  001102          MOV     #6,$TSTNM       ;SETUP TEST NUMBER AND
2852                                          ;CLEAR THE ERROR FLAG ($ERFLG)
2853  010072  013777  001102  171042          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2854  010100  012737  010366  001200          MOV     #EXIT6,BYPASS   ;SETUP BYPASS ADDRESS
2855  010106  013704  001324                  MOV     RPADR,R4        ;RP11E BUS ADDRESS
2856  010112  105037  002501                  CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
2857  010116  105037  002503                  CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
2858  010122  012737  000031  001162          MOV     #25.,$TIMES     ;DO 25. ITERATIONS
2859  010130  012706  001100          TEST6:  MOV     #STACK,SP       ;SETUP THE STACK POINTER
2860  010134  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
2861  010140  005037  002512                  CLR     $SEC            ;SECTOR/TRACK 0
2862  010144  005037  002510                  CLR     $CYL            ;CYLINDER 0
2863  010150  112737  000015  002502          MOVB    #HOMSEK,DPB+2   ;HOME SEEK COMMAND
```

```
2864  010156  004737  022770                        JSR     PC,RP11         ;DO THE COMMAND
2865  010162  004737  023546                        JSR     PC,DRVRDY       ;WAIT FOR DRIVE READY
2866  010166  012737  000300  002510                MOV     #192.,$CYL      ;CHANGE CYLINDER TO CYLINDER 192
2867  010174  112737  000011  002502                MOVB    #SEEK,DPB+2     ;SEEK COMMAND
2868  010202  004737  022770                        JSR     PC,RP11         ;INITIATE THE SEEK
2869  010206  012737  000001  001220                MOV     #1,STALLT       ;LOAD STALL VALUE
2870  010214  004737  023202                        JSR     PC,STALL        ;STALL FOR 1 MILLISECONDS
2871  010220  012764  177775  000006                MOV     #-3,RPWC(R4)    ;SET WORD COUNT TO THREE WORDS
2872  010226  012764  043656  000010                MOV     #BUFFER,RPBA(R4)  ;BUFFER ADDRESS
2873  010234  052737  014000  002502                BIS     #MODE!HDR,DPB+2 ;SETUP TO READ THE HEADER
2874  010242  112764  000017  000004                MOVB    #READ,RPCS(R4)  ;TRY TO DO A READ
2875  010250  004737  023546                        JSR     PC,DRVRDY       ;WAIT FOR DRIVE READY
2876  010254  004737  023240                        JSR     PC,SAVRP        ;STORE THE REGISTERS
2877  010260  032737  000200  001340                BIT     #RDY,$RPCS      ;DID CONTROLLER BECOME READY ALSO ?
2878  010266  001402                                BEQ     1$              ;BR IF NOT
2879  010270  104033                                ERROR   33              ;READ DIDN'T START DURING SEEK
2880  010272  000435                                BR      EXIT6           ;EXIT
2881  010274  004737  023664          1$:           JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
2882  010300  004737  023240                        JSR     PC,SAVRP        ;STORE THE REGISTERS
2883  010304  032737  100000  001340                BIT     #ERR,$RPCS      ;ANY ERRORS?
2884  010312  001401                                BEQ     2$              ;BRANCH IF NO
2885  010314  104034                                ERROR   34              ;ERROR OCCURED DURING A READ ISSUED
2886                                                                        ;WHILE THE DRIVE IS SEEKING
2887  010316  012737  014000  001124  2$:           MOV     #<192.*32.>,$GDDAT  ;EXPECTED CYLINDER VALUE
2888  010324  013737  043660  001126                MOV     BUFFER+2,$BDDAT ;RECEIVED DATA
2889  010332  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;CORRECT ?
2890  010340  001366                                BNE     2$              ;BR IF NOT
2891  010342  005037  001124                        CLR     $GDDAT          ;EXPECTED VALUE FOR SECTOR
2892  010346  013737  043662  001126                MOV     BUFFER+4,$BDDAT ;RECEIVED SECTOR
2893  010354  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;CORRECT ?
2894  010362  001401                                BEQ     EXIT6           ;BR IF OK
2895  010364  104035                                ERROR   35              ;HEADER DATA INCORRECT
2896  010366  000004                  EXIT6:        SCOPE
2897
2898
2899                                                ;;*********************************************************************
2900                                                ;*TEST 7         TEST 'NXME' BIT
2901
2902                                                ;*THE NON-EXISTENT MEMORY ERROR BIT IS TESTED BY ATTEMPTING TO READ 1
2903                                                ;*WORD INTO LOCATION 760000.   'NXME', 'HE', & 'ERR' SHOULD ALL BE SET.
2904
2905                                                ;;*********************************************************************
2906  010370                                        TST7:
2907  010370  033737  001416  001276                        BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
2908  010376  001002                                        BNE     .+6             ;BR IF YES
2909  010400  000137  010612                                JMP     TST10           ;GO TO THE NEXT TEST
2910  010404  012737  010464  001110                        MOV     #TEST7,$LPERR   ;SETUP THE ERROR LOOP ADDRESS
2911  010412  012737  010370  001106                        MOV     #TST7,$LPADR    ;SETUP THE SCOPE LOOP ADDRESS
2912  010420  012737  000007  001102                        MOV     #7,$TSTNM       ;SETUP TEST NUMBER AND
2913                                                                                ;CLEAR THE ERROR FLAG (SERFLG)
2914  010426  013777  001102  170506                        MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2915  010434  012737  010610  001200                        MOV     #EXIT7,BYPASS   ;SETUP BYPASS ADDRESS
2916  010442  013704  001324                                MOV     RPADR,R4        ;RP11E BUS ADDRESS
2917  010446  105037  002501                                CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
2918  010452  105037  002503                                CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
2919  010456  012737  000144  001162                        MOV     #100.,$TIMES    ;;DO 100. ITERATIONS
```

```
                                TEST7:  MOV     #STACK,SP       ;SETUP THE STACK POINTER
                                        JSR     PC,CLRP         ;CLEAR THE RP11
                                        MOV     #1,$WC          ;WORD COUNT
                                        CLR     $SEC            ;SECTOR/TRACK ZERO
                                        CLR     $CYL            ;CYLINDER ZERO
                                        MOVB    #MEX1!MEX0,DPB+1  ;LOAD BOTH EXTENDED MEMORY BITS
                                        MOV     #160000,$BUF    ;LOAD ADDR TO FORCE NON EX MEMORY
                                        MOVB    #RDSEK,DPB+2    ;TRY TO READ INTO THE LOCATION
                                        JSR     PC,RP11         ;INITIATE THE OPERATION
                                        JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
                                        JSR     PC,SAVRP        ;STORE THE REGISTERS
                                        BIT     #BIT02,$RPER    ;DID NON EX MEMORY SET ?
                                        BNE     2$              ;BRANCH IF SET
                                        ERROR   36              ;'NXME' ERROR DIDN'T SET
                                        BR      EXIT7           ;DON'T CHECK THE RPCS BITS
                                2$:     BIT     #HE,$RPCS       ;DID 'HE' SET ON NON EX MEMORY
                                        BNE     3$              ;BRANCH IF SET
                                        ERROR   37              ;'HE' DIDN'T SET
                                3$:     BIT     #ERR,$RPCS      ;DID 'ERR' SET WITH NON EX MEMORY
                                        BNE     EXIT7           ;BRANCH IF SET
                                        ERROR   40              ;'ERR' DIDN'T SET
                                EXIT7:  SCOPE

                                ;*****************************************************************
                                ;*TEST 10       SECTOR ADDRESSING TEST

                                ;*THE SECTOR ADDRESSING TEST TESTS THE CAPABILTIY OF THE RP11E TO LOCATE
                                ;*A SECTOR BY USING THE 'SOT' COUNTER (FOR 'HDR' OPERATIONS) AND TO LOCATE
                                ;*A SECTOR BY COMPARING THE HEADER CONTENTS AGAINST THE CYLINDER AND TRACK/
                                ;*SECTOR ADDRESS REGISTERS (NORMAL MODE).  THE TEST IS PERFORMED IN 3 PARTS.
                                ;*
                                ;*      1.  WRITE ALL HEADERS ON TRACK 0, CYLINDER 0 IN ASCENDING SEQUENCE
                                ;*      FROM INDEX (0, 1, 2 ... 9).
                                ;*
                                ;*      2.  READ EACH HEADER, BEGINNING WITH SECTOR 0, AND VERIFY THAT THE
                                ;*      SECTOR FIELD IN THE HEADER IS CORRECT FOR THE HEADER EXPECTED.
                                ;*
                                ;*      VERIFY THAT THE 'SOT' COUNTER AND THE SECTOR ADDRESS IN 'RPDA' ARE
                                ;*      EQUAL AND ARE THE EXPECTED VALUE.  THE EXPECTED VALUE IS 1 GREATER
                                ;*      THAN THE SECTOR READ.  THE SECTOR ADDRESS REGISTER AND THE 'SOT'
                                ;*      WILL BE INCREMENTED BY THE TIME THE PROGRAM IS ABLE TO READ THE
                                ;*      REGISTER.
                                ;*
                                ;*      WRITE THE SECTOR'S ADDRESS IN THE DATA FIELD.  (FOR SECTOR 0, AN
                                ;*      OCTAL 12 IS WRITTEN.)
                                ;*
                                ;*      CONTINUE THIS SEQUENCE FOR THE REMAINING SECTORS ON THE TRACK.
                                ;*
                                ;*      3.  READ THE DATA FROM EACH SECTOR USING A NORMAL (I.E., NON-HEADER)
                                ;*      READ AND VERIFY THAT THE DATA IS CORRECT FOR THE EXPECTED SECTOR.

                                ;*****************************************************************
                                TST10:
                                        BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
                                        BNE     .+6             ;BR IF YES
```

```
2976  010622  000137  011532            JMP     TST11           ;GO TO THE NEXT TEST
2977  010626  012737  010706  001110    MOV     #TEST10,$LPERR  ;SETUP THE ERROR LOOP ADDRESS
2978  010634  012737  010612  001106    MOV     #TST10,$LPADR   ;SETUP THE SCOPE LOOP ADDRESS
2979  010642  012737  000010  001102    MOV     #10,$TSTNM      ;SETUP TEST NUMBER AND
2980                                                            ;CLEAR THE ERROR FLAG ($ERFLG)
2981  010650  013777  001102  170264    MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2982  010656  012737  011530  001200    MOV     #EXIT10,BYPASS  ;SETUP BYPASS ADDRESS
2983  010664  013704  001324            MOV     RPADR,R4        ;RP11E BUS ADDRESS
2984  010670  105037  002501            CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
2985  010674  105037  002503            CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
2986  010700  012737  000031  001162    MOV     #25.,$TIMES     ;DO 25. ITERATIONS
2987  010706  012706  001100  TEST10:   MOV     #STACK,SP       ;RESET THE STACK POINTER
2988  010712  012737  010706  001110    MOV     #TEST10,$LPERR  ;LOOP ON ERROR ADDRESS
2989  010720  004737  025654            JSR     PC,CLRP         ;CLEAR THE RP11
2990  010724  012737  043656  002506    MOV     #BUFFER,$BUF    ;SETUP BUFFER ADDRESS
2991  010732  012737  000036  002504    MOV     #30.,$WC        ;SETUP WORD COUNT FOR 10 SECTORS
2992  010740  005037  002510            CLR     $CYL            ;SETUP FOR CYLINDER 0
2993  010744  005037  002512            CLR     $SEC            ;TRACK/SECTOR 0
2994  010750  004737  024304            JSR     PC,LODSEC       ;SETUP FORMAT BUFFER
2995  010754  012737  014000  002502    MOV     #MODE!HDR,DPB+2 ;SETUP THE HEADER CONTROL BITS
2996  010762  112737  000003  002502    MOVB    #WRTSEK,DPB+2   ;THE COMMAND CODE
2997  010770  004737  022770            JSR     PC,RP11         ;FORMAT THE TRACK
2998  010774  004737  023664            JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
2999  011000  004737  023240            JSR     PC,SAVRP        ;STORE THE REGISTERS
3000  011004  032737  100000  001340    BIT     #ERR,$RPCS      ;ERROR ?
3001  011012  001403                    BEQ     1$              ;BR IF NOT
3002  011014  104041                    ERROR   41              ;ERROR TRYING TO FORMAT THE TRACK
3003  011016  000137  011530            JMP     EXIT10          ;BYPASS REST OF THE TEST
3004  011022  105037  002512  1$:       CLRB    $SEC            ;START CHECK WITH SECTOR 0
3005  011026  012737  011026  001110  2$:       MOV     #2$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
3006  011034  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
3007  011040  012737  014000  002502            MOV     #MODE!HDR,DPB+2 ;SET 'MODE' AND 'HDR' BITS
3008  011046  112737  000005  002502            MOVB    #RDSEK,DPB+2    ;CHANGE THE OPERATION TO A READ
3009  011054  012737  000003  002504            MOV     #3,$WC          ;CHANGE THE WORD COUNT TO 3
3010  011062  004737  022770                    JSR     PC,RP11         ;READ THE SECTOR'S HEADER
3011  011066  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
3012  011072  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
3013  011076  012737  000000  001220            MOV     #0,STALLT       ;LOAD STALL VALUE
3014  011104  004737  023202                    JSR     PC,STALL        ;STALL FOR 0 MILLISECONDS
3015  011110  016437  000014  001350            MOV     RPDA(R4),$RPDA  ;GET THE CONTENTS OF THE ADDRESS REGISTER
3016  011116  032737  100000  001340            BIT     #ERR,$RPCS      ;ERROR ?
3017  011124  001402                            BEQ     3$              ;BR IF NOT
3018  011126  104042                            ERROR   42              ;ERROR READING THE HEADER
3019  011130  000457                            BR      6$              ;BYPASS DATA CHECK
3020  011132  005037  001124            3$:     CLR     $GDDAT          ;CLEAR FOR EXPECTED DATA
3021  011136  113737  002512  001124            MOVB    $SEC,$GDDAT     ;MOVE SECTOR ADDRESS
3022  011144  023737  001124  043662            CMP     $GDDAT,BUFFER+4 ;CHECK THE SECTOR FIELD
3023  011152  001401                            BEQ     4$              ;BR IF CORRECT
3024  011154  104043                            ERROR   43              ;SECTOR FIELD FROM HEADER NOT CORRECT
3025  011156  005237  001124            4$:     INC     $GDDAT          ;INCREMENT SECTOR ADDRESS FOR 'SOT' CHECK
3026  011162  023727  001124  000012            CMP     $GDDAT,#10.     ;EXCEED THE MAXIMUM ?
3027  011170  103402                            BLO     .+6             ;BR IF NOT
3028  011172  005037  001124                    CLR     $GDDAT          ;RESET COMPARISON VALUE
3029  011176  013737  001350  001126            MOV     $RPDA,$BDDAT    ;CHECK THE 'SOT'
3030  011204  042737  177417  001126            BIC     #^C360,$BDDAT   ;CLEAR THE OTHER BITS
3031  011212  006237  001126                    ASR     $BDDAT          ;SHIFT THE 'SOT'
```

```
3032  011216  006237  001126                        ASR     $BDDAT          ;SHIFT THE 'SOT'
3033  011222  006237  001126                        ASR     $BDDAT          ;SHIFT THE 'SOT'
3034  011226  006237  001126                        ASR     $BDDAT          ;SHIFT THE 'SOT'
3035  011232  013737  001350  001122                MOV     $RPDA,$BDADR    ;CHECK THE SECTOR ADDRESS
3036  011240  042737  177760  001122                BIC     #^C17,$BDADR    ;LEAVE THE SECTOR ADDRESS
3037  011246  023737  001124  001126                CMP     $GDDAT,$BDDAT   ;IS THE 'SOT' CORRECT ?
3038  011254  001004                                BNE     5$              ;BR IF NOT
3039  011256  023737  001126  001122                CMP     $BDDAT,$BDADR   ;DOES THE SECTOR ADDRESS EQUAT THE 'SOT' ?
3040  011264  001401                                BEQ     6$              ;BR IF IT DOES
3041  011266  104045                          5$:   ERROR   45              ;'SOT' OR SECTOR ADDRESS IS NOT CORRECT
3042  011270  012737  011270  001110        6$:   MOV     #6$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
3043  011276  004737  025654                        JSR     PC,CLRP         ;CLEAR THE RP11
3044  011302  105037  002503                        CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
3045  011306  004737  024336                        JSR     PC,FILSEC       ;SETUP TO WRITE SECTOR ADDRESS AS DATA
3046  011312  112737  000003  002502                MOVB    #WRTSEK,DPB+2   ;CHANGE OPERATION
3047  011320  004737  022770                        JSR     PC,RP11         ;WRITE THE SECTOR
3048  011324  004737  023664                        JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
3049  011330  004737  023240                        JSR     PC,SAVRP        ;STORE THE REGISTERS
3050  011334  032737  100000  001340                BIT     #ERR,$RPCS      ;ERROR ?
3051  011342  001401                                BEQ     7$              ;BR IF NOT
3052  011344  104044                                ERROR   44              ;ERROR TRYING TO WRITE SECTOR ADDRESS AS DATA
3053  011346  112737  000005  002502  7$:   MOVB    #RDSEK,DPB+2    ;CHANGE THE OPERATION
3054  011354  105237  002512                        INCB    $SEC            ;INCREMENT THE SECTOR
3055  011360  123727  002512  000012                CMPB    $SEC,#10.       ;AT MAXIMUM SECTOR ADDRESS ?
3056  011366  001217                                BNE     2$              ;BR IF NOT
3057  011370  012737  011424  001110                MOV     #8$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
3058  011376  012737  000400  002504                MOV     #256.,$WC       ;CHANGE WORD COUNT
3059  011404  105037  002512                        CLRB    $SEC            ;START WITH SECTOR ZERO
3060  011410  012737  000012  001124                MOV     #10.,$GDDAT     ;SECTOR ZERO PATTERN
3061  011416  112737  000005  002502                MOVB    #RDSEK,DPB+2    ;CHANGE OPERATION CODE
3062  011424  004737  025654                  8$:   JSR     PC,CLRP         ;CLEAR THE RP11
3063  011430  004737  022770                        JSR     PC,RP11         ;READ THE SECTOR
3064  011434  004737  023664                        JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
3065  011440  004737  023240                        JSR     PC,SAVRP        ;STORE THE REGISTERS
3066  011444  032737  100000  001340                BIT     #ERR,$RPCS      ;ERROR ?
3067  011452  001402                                BEQ     9$              ;BR IF NOT
3068  011454  104017                                ERROR   17              ;ERROR TRYING TO READ THE SECTOR
3069  011456  000412                                BR      10$             ;BYPASS DATA CHECK
3070  011460  005037  001120                  9$:   CLR     $GDADR          ;CLEAR FOR EXPECTED DATA
3071  011464  113737  002512  001120                MOVB    $SEC,$GDADR     ;EXPECTED DATA
3072  011472  023737  001124  043656                CMP     $GDDAT,BUFFER   ;CORRECT CONTENTS ?
3073  011500  001401                                BEQ     10$             ;BR IF CORRECT
3074  011502  104046                                ERROR   46              ;SECTOR PATTERN NOT CORRECT FOR SECTOR ADDRESS
3075  011504  105237  002512                 10$:   INCB    $SEC            ;INCREMENT SECTOR ADDRESS
3076  011510  123727  002512  000012                CMPB    $SEC,#10.       ;FINISHED SECTOR NINE ?
3077  011516  001404                                BEQ     EXIT10          ;BR IF COMPLETED
3078  011520  113737  002512  001124                MOVB    $SEC,$GDDAT     ;NEXT SECTOR
3079  011526  000736                                BR      8$              ;CONTINUE CHECKING
3080  011530  000004                          EXIT10: SCOPE                 ;LOOP ?
3081
3082
3083                                          ;;*****************************************************************
3084                                          ;*TEST 11        TRACK ADDRESSING TEST
3085
3086                                          ;*TRACK ADDRESSING IS TESTED BY WRITING THE TRACK'S ADDRESS AS DATA IN
3087                                          ;*SECTOR 0 OF EACH TRACK IN CYLINDER 0.  SECTOR 0 IS THEN READ BACK AND
```

```
                                        ;*THE DATA IS CHECKED TO VERIFY THAT THE PROPER HEAD WAS SELECTED.
                                        ;
                                        ;**********************************************************************
011532                                  TST11:
011532  033737  001422  001276                BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
011540  001002                            BNE     .+6                 ;BR IF YES
011542  000137  012062                    JMP     TST12               ;GO TO THE NEXT TEST
011546  012737  011652  001110            MOV     #TEST11,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
011554  012737  011532  001106            MOV     #TST11,$LPADR       ;SETUP THE SCOPE LOOP ADDRESS
011562  012737  000011  001102            MOV     #11,$TSTNM          ;SETUP TEST NUMBER AND
                                                                      ;CLEAR THE ERROR FLAG ($ERFLG)
011570  013777  001102  167344            MOV     $TSTNM,@DISPLAY     ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
011576  012737  012060  001200            MOV     #EXIT11,BYPASS      ;SETUP BYPASS ADDRESS
011604  013704  001324                    MOV     RPADR,R4            ;RP11E BUS ADDRESS
011610  105037  002501                    CLRB    DPB+1               ;CLEAR ANY EXTENDED MEMORY BITS
011614  105037  002503                    CLRB    DPB+3               ;CLEAR 'MODE' & 'HDR' BITS
011620  012737  000031  001162            MOV     #25.,$TIMES         ;DO 25. ITERATIONS
011626  005037  002512                    CLR     $SEC                ;CLEAR SECTOR/TRACK ADDRESS
011632  005037  002510                    CLR     $CYL                ;CLEAR CYLINDER ADDRESS
011636  012737  000400  002504            MOV     #256.,$WC           ;SET WORD COUNT TO ONE SECTOR
011644  012737  043656  002506            MOV     #BUFFER,$BUF        ;SETUP THE BUFFER ADDRESS
011652  012706  001100                TEST11: MOV     #STACK,SP           ;SETUP THE STACK POINTER
011656  012737  011656  001110        1$:     MOV     #1$,$LPERR          ;CHANGE LOOP ON ERROR ADDRESS
011664  004737  025654                    JSR     PC,CLRP             ;CLEAR THE RP11
011670  112737  000003  002502            MOVB    #WRTSEK,DPB+2       ;CHANGE COMMAND TO WRITE
011676  004737  024376                    JSR     PC,FILTRK           ;FILL BUFFER WITH THE TRACK ADDRESS
011702  004737  022770                    JSR     PC,RP11             ;WRITE THE TRACK'S ADDRESS
011706  004737  023664                    JSR     PC,CONRDY           ;WAIT FOR CONTROLLER TO FINISH
011712  004737  023240                    JSR     PC,SAVRP            ;STORE THE REGISTERS
011716  032737  100000  001340            BIT     #ERR,$RPCS          ;ERROR ?
011724  001401                            BEQ     2$                  ;BR IF NOT
011726  104004                            ERROR   4                   ;ERROR WRITTING TEST PATTERN
011730  105237  002513                2$:     INCB    $TRK                ;INCREMENT THE TRACK ADDRESS
011734  123727  002513  000024            CMPB    $TRK,#20.           ;FINISHED ?
011742  103745                            BLO     1$                  ;BR IF NOT FINISHED
011744  005037  001124                3$:     CLR     $GDDAT              ;SETUP EXTPECTED VALUE
011750  005037  002512                    CLR     $SEC                ;START WITH ZERO
011754  012737  011754  001110        4$:     MOV     #4$,$LPERR          ;CHANGE THE LOOP ON ERROR ADDRESS
011762  004737  025654                    JSR     PC,CLRP             ;CLEAR THE RP11
011766  112737  000005  002502            MOVB    #RDSEK,DPB+2        ;CHANGE COMMAND
011774  004737  022770                    JSR     PC,RP11             ;READ THE SECTOR
012000  004737  023664                    JSR     PC,CONRDY           ;WAIT FOR THE CONTROLLER TO BE READY
012004  004737  023240                    JSR     PC,SAVRP            ;STORE THE REGISTERS
012010  032737  100000  001340            BIT     #ERR,$RPCS          ;ERROR ?
012016  001402                            BEQ     5$                  ;BR IF NOT
012020  104017                            ERROR   17                  ;ERROR READING THE SECTOR
012022  000405                            BR      6$                  ;BYPASS THE DATA CHECK
012024  123737  001125  043657    5$:     CMPB    $GDDAT+1,BUFFER+1   ;CORRECT VALUE ?
012032  001401                            BEQ     6$                  ;BR IF OK
012034  104047                            ERROR   47                  ;CONTENTS WRONG
012036  123727  002513  000023    6$:     CMPB    $TRK,#19.           ;FINISHED ?
012044  001405                            BEQ     EXIT11              ;BR IF FINISHED
012046  105237  002513                    INCB    $TRK                ;INCREMENT THE TRACK ADDRESS
012052  105237  001125                    INCB    $GDDAT+1            ;COMPARSION WORD
012056  000736                            BR      4$                  ;CONTINUE
012060  000004                    EXIT11: SCOPE                       ;LOOP ?
```

```
                                ;;*****************************************************************
                                ;*TEST 12       EXTENDED MEMORY ADDRESS TEST
                                ;
                                ;*THIS TEST TESTS THE OPERATION OF THE EXTENDED MEMORY ADDRESS BITS.  IF THE
                                ;*SYSTEM DOES NOT HAVE MEMORY MANAGEMENT OR HAS MEMORY MANAGEMENT AND ONLY
                                ;*32K, THE TEST WILL NOT BE PERFORMED.
                                ;*
                                ;*      1.  THE PROGRAM WRITES A 2 WORD TEST SECTOR OF ALL ONES.
                                ;*
                                ;*      2.  EXTENDED ADDRESS BIT 'MEX00' IS TESTED BY CLEARING LOCATION
                                ;*      200000 AND READING THE TEST SECTOR INTO LOCATION 177776.  LOCATION
                                ;*      200000 IS CHECKED TO VERIFY THAT THE DATA IS CORRECT (ONES).  THE
                                ;*      PROGRAM VERIFIES THAT 'MEX00' HAS SET AND THAT 'MEX01' IS NOT SET.
                                ;*
                                ;*      3.  IF THE SYSTEM HAS AT LEAST 64K, 'MEX01' IS TESTED.  LOCATION
                                ;*      400000 IS CLEARED AND THE TEST SECTOR IS READ INTO LOCATION 377776
                                ;*      ('MEX00' IS SET FOR THE READ).  LOCATION 400000 IS CHECKED FOR THE
                                ;*      PROPER CONTENTS (ONES).  'MEX00' SHOULD HAVE RESET AND 'MEX01' SHOULD
                                ;*      BE SET.
                                ;
                                ;;*****************************************************************
012062                          TST12:
012062  033737  001424  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
012070  001002                        BNE     .+6                  ;BR IF YES
012072  000137  012720                JMP     TST13                ;GO TO THE NEXT TEST
012076  012737  012202  001110        MOV     #TEST12,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
012104  012737  012062  001106        MOV     #TST12,$LPADR        ;SETUP THE SCOPE LOOP ADDRESS
012112  012737  000012  001102        MOV     #12,$TSTNM           ;SETUP TEST NUMBER AND
                                                                   ;CLEAR THE ERROR FLAG ($ERFLG)
012120  013777  001102  167014        MOV     $TSTNM,@DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
012126  012737  012712  001200        MOV     #EXIT12,BYPASS       ;SETUP BYPASS ADDRESS
012134  013704  001324                MOV     RPADR,R4             ;RP11E BUS ADDRESS
012140  105037  002501                CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
012144  105037  002503                CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
012150  012737  000144  001162        MOV     #100.,$TIMES         ;DO 100. ITERATIONS
012156  005737  022232                TST     $KT11                ;MEMORY MANAGEMENT ON THE SYSTEM ?
012162  100402                        BMI     1$                   ;BR IF ON SYSTEM
012164  000137  012720                JMP     TST13                ;GO TO THE NEXT TEST
012170  005737  001232          1$:   TST     MAXMEM+2             ;ENOUGH MEMORY FOR THIS TEST ?
012174  001002                        BNE     TEST12               ;BR IF YES
012176  000137  012720                JMP     TST13                ;GO TO THE NEXT TEST
012202  012706  001100          TEST12: MOV     #STACK,SP            ;SETUP THE STACK POINTER
012206  012737  012202  001110        MOV     #TEST12,$LPERR       ;INITIAL ERROR LOOP ADDRESS
012214  004737  025654                JSR     PC,CLRP              ;CLEAR THE RP11
012220  005037  043656                CLR     BUFFER               ;TEST PATTERN
012224  012737  177777  043660        MOV     #-1,BUFFER+2         ;SECOND WORD OF PATTERN
012232  012737  043656  002506        MOV     #BUFFER,$BUF         ;SETUP THE BUFFER ADDRESS
012240  012737  000002  002504        MOV     #2,$WC               ;WORD COUNT OF 2
012246  005037  002512                CLR     $SEC                 ;SECTOR/TRACK ZERO
012252  005037  002510                CLR     $CYL                 ;CYLINDER ZERO
012256  012737  000003  002502        MOV     #WRTSEK,DPB+2        ;WRITE COMMAND
012264  004737  022770                JSR     PC,RP11              ;INITIATE THE COMMAND
012270  004737  023664                JSR     PC,CONRDY            ;WAIT FOR CONTROLLER READY
012274  004737  023240                JSR     PC,SAVRP             ;STORE THE REGISTERS
```

```
3200  012300  032737  100000  001340          BIT     #ERR,$RPCS      ;DID OPERATION TERMINATE WITH AN ERROR ?
3201  012306  001403                          BEQ     1$              ;BR IF NOT
3202  012310  104004                          ERROR   4               ;ERROR AFTER 2 WORD WRITE
3203  012312  000137  012712                  JMP     EXIT12          ;BYPASS REST OF TEST
3204  012316  012737  012316  001110  1$:     MOV     #1$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
3205  012324  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
3206  012330  012737  177600  172356          MOV     #177600,KIPAR7  ;OPEN I/O REGISTERS
3207  012336  005037  172340                  CLR     KIPAR0          ;ENABLE FIRST 4K
3208  012342  012737  000200  172342          MOV     #200,KIPAR1     ;ENABLE SECOND 4K
3209  012350  012737  002000  172344          MOV     #2000,KIPAR2    ;ENABLE FIRST TEST LOCATION
3210  012356  012737  077406  172300          MOV     #77406,KIPDR0   ;LOAD DESCRIPTOR REGISTER 0
3211  012364  012737  077406  172302          MOV     #77406,KIPDR1   ;LOAD DESCRIPTOR REGISTER 1
3212  012372  012737  077406  172304          MOV     #77406,KIPDR2   ;LOAD DESCRIPTOR REGISTER 2
3213  012400  012737  077406  172316          MOV     #77406,KIPDR7   ;LOAD DESCRIPTOR REGISTER 7
3214  012406  012737  000001  177572          MOV     #1,@#SR0        ;TURN ON MEMORY MANAGEMENT
3215  012414  005037  040000                  CLR     @#40000         ;CLEAR LOCATION 200000
3216  012420  012737  177776  002506          MOV     #177776,$BUF    ;SETUP BUS ADDR
3217  012426  112737  000005  002502          MOVB    #RDSEK,DPB+2    ;CHANGE COMMAND TO READ
3218  012434  004737  022770                  JSR     PC,RP11         ;START THE COMMAND
3219  012440  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
3220  012444  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
3221  012450  032737  100000  001340          BIT     #ERR,$RPCS      ;ANY ERRORS?
3222  012456  001401                          BEQ     2$              ;BRANCH IF NOT
3223  012460  104050                          ERROR   50              ;ERROR AFTER READING 2 WORDS
3224  012462  032737  000020  001340  2$:     BIT     #MEX0,$RPCS     ;DID 'MEX0' SET ?
3225  012470  001001                          BNE     3$              ;BRANCH IF IT DID
3226  012472  104051                          ERROR   51              ;'MEX0' DID NOT SET AFTER 2 WORD READ
3227                                                                  ;STARTING AT LOCATION 177776
3228  012474  032737  000040  001340  3$:     BIT     #MEX1,$RPCS     ;'MEX1' SHOULD NOT BE SET
3229  012502  001401                          BEQ     4$              ;BRANCH IF NOT SET
3230  012504  104052                          ERROR   52              ;'MEX1' IS SET - SHOULD NOT BE
3231  012506  022737  177777  040000  4$:     CMP     #177777,@#40000 ;WAS DATA READ INTO LOCATION
3232  012514  001407                          BEQ     TST12B          ;200000 CORRECTLY? - BRANCH IF YES
3233  012516  012737  177777  001124          MOV     #177777,$GDDAT  ;EXPECTED CONTENTS
3234  012524  013737  040000  001126          MOV     @#40000,$BDDAT  ;CONTENTS OF LOCATION 200000
3235  012532  104053                          ERROR   53              ;DATA COMPARE ERROR AT LOC 200000
3236  012534  023727  001232  000002  TST12B: CMP     MAXMEM+2,#2     ;ENOUGH MEMORY FOR THE NEXT PART OF THE TEST ?
3237  012542  103463                          BLO     EXIT12          ;BR IF NOT
3238  012544  012737  004000  172344          MOV     #4000,KIPAR2    ;CHANGE TEST LOCATION
3239  012552  012737  012552  001110  1$:     MOV     #1$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
3240  012560  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
3241  012564  005037  040000                  CLR     @#40000         ;CLEAR LOCATION 400000
3242  012570  012737  177776  002506          MOV     #177776,$BUF    ;BUFFER ADDRESS
3243  012576  112737  000020  002501          MOVB    #MEX0,DPB+1     ;SETUP EXTENDED MEMORY ADDRESS BIT
3244  012604  112737  000005  002502          MOVB    #RDSEK,DPB+2    ;READ COMMAND
3245  012612  004737  022770                  JSR     PC,RP11         ;START THE COMMAND
3246  012616  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
3247  012622  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
3248  012626  032737  100000  001340          BIT     #ERR,$RPCS      ;ANY ERRORS?
3249  012634  001401                          BEQ     2$              ;BRANCH IF NOT
3250  012636  104054                          ERROR   54              ;ERROR WHILE READING TWO WORDS
3251  012640  032737  000020  001340  2$:     BIT     #MEX0,$RPCS     ;IS 'MEX0' SET ?
3252  012646  001401                          BEQ     3$              ;BRANCH IF NOT
3253  012650  104055                          ERROR   55              ;'MEX0' DID NOT CLEAR AFTER 2 WORD
3254                                                                  ;READ STARTING AT LOCATION 377776
3255  012652  032737  000040  001340  3$:     BIT     #MEX1,$RPCS     ;DID 'MEX1' SET ?
```

```
                                        BNE     4$              ;BR IF SET
                                        ERROR   56              ;'MEX1' DID NOT SET AFTER 2 WORD TRANSFER
                                                                ;STARTING AT LOCATION 377776
012664  022737  177777  040000  4$:   CMP     #177777,@#40000 ;WAS DATA READ INTO LOCATION 400000
012672  001407                        BEQ     EXIT12          ;CORRECTLY ? - BRANCH IF YES
012674  012737  177777  001124        MOV     #177777,$GDDAT  ;EXPECTED DATA
012702  013737  040000  001126        MOV     @#40000,$BDDAT  ;DATA FROM LOCATION 400000
012710  104057                        ERROR   57              ;DATA COMPARE ERROR AT LOC 400000
012712  005037  177572        EXIT12: CLR     @#SR0           ;TURN OFF MEMORY MANAGEMENT
012716  000004                        SCOPE                   ;LOOP ?


;;*********************************************************************
;*TEST 13        DATA BUFFER REGISTER BIT TEST

;*THE DATA BUFFER REGISTER TEST VERIFIES THAT ALL 36 BITS OF THE DATA
;*BUFFER REGISTER CAN BE SET AND CLEARED.
;*
;*      1.  THE TEST FIRST WRITES THE FOLLOWING TEST PATTERN IN CYLINDER 0,
;*      TRACK 0, SECTOR 0 USING 10/15 MODE ('MODE' BIT SET).
;*
;*                      052525
;*                      052525
;*                      000005
;*                      125252
;*                      125252
;*                      000012
;*                      052525
;*                      052525
;*                      000005
;*                      125252
;*                        .
;*                        .
;*                      ETC
;*
;*      2.  THE TEST SECTOR IS THEN READ AND THE DATA IS COMPARED.  AT THE
;*      COMPLETION OF THE TEST, THE PROGRAM RESTORES THE TEST SECTOR TO
;*      PDP-11 MODE.
;*
;;*********************************************************************
012720
TST13:
012720  033737  001426  001276        BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
012726  001002                        BNE     .+6             ;BR IF YES
012730  000137  013574                JMP     TST14           ;GO TO THE NEXT TEST
012734  012737  013014  001110        MOV     #TEST13,$LPERR  ;SETUP THE ERROR LOOP ADDRESS
012742  012737  012720  001106        MOV     #TST13,$LPADR   ;SETUP THE SCOPE LOOP ADDRESS
012750  012737  000013  001102        MOV     #13,$TSTNM      ;SETUP TEST NUMBER AND
                                                                ;CLEAR THE ERROR FLAG ($ERFLG)
012756  013777  001102  166156        MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
012764  012737  013572  001200        MOV     #EXIT13,BYPASS  ;SETUP BYPASS ADDRESS
012772  013704  001324                MOV     RPADR,R4        ;RP11E BUS ADDRESS
012776  105037  002501                CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
013002  105037  002503                CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
013006  012737  000144  001162        MOV     #100.,$TIMES    ;;DO 100. ITERATIONS
013014  012706  001100        TEST13: MOV     #STACK,SP       ;SETUP THE STACK POINTER
```

```
 3312  013020  012701  043656            MOV     #BUFFER,R1      ;BUFFER ADDRESS
 3313  013024  012702  000200            MOV     #128.,R2        ;SIZE
 3314  013030  012721  052525      1$:   MOV     #52525,(R1)+    ;PATTERN FOR BITS 20 - 35
 3315  013034  012721  052525            MOV     #52525,(R1)+    ;PATTERN FOR BITS 4 - 19
 3316  013040  012721  000005            MOV     #5,(R1)+        ;PATTERN FOR BITS 0 - 3
 3317  013044  005302                    DEC     R2              ;AT THE END ?
 3318  013046  001410                    BEQ     2$              ;BR IF END
 3319  013050  012721  125252            MOV     #125252,(R1)+   ;PATTERN FOR BITS 20 - 35
 3320  013054  012721  125252            MOV     #125252,(R1)+   ;PATTERN FOR BITS 4 - 19
 3321  013060  012721  000012            MOV     #12,(R1)+       ;PATTERN FOR BITS 0 - 3
 3322  013064  005302                    DEC     R2              ;AT THE END ?
 3323  013066  001360                    BNE     1$              ;BR IF NOT
 3324  013070  012737  013070  001110  2$:   MOV     #2$,$LPERR      ;LOOP ON ERROR ADDRESS
 3325  013076  004737  025654            JSR     PC,CLRP         ;CLEAR THE RP11
```

```
                                                ;SECTOR/TRACK ADDRESS OF ZERO
                                                ;CYLINDER ZERO
                                                ;1 SECTOR IN 10/15 MODE
                                                ;BUFFER ADDRESS
                                                ;SET THE 'MODE' BIT
                                                ;WRITE COMMAND
                                                ;DO THE COMMAND
                                                ;WAIT FOR THE COMMAND TO COMPLETE
                                                ;SAVE THE REGISTERS
                                                ;OPERATION GO OK ?
                                                ;BR IF IT DID
                                                ;SETUP OPERATION TERMINATED WITH AN ERROR
                                                ;CORRECT THE SECTOR AND BYPASS REST OF TEST
                                                ;CHANGE THE LOOP ON ERROR ADDRESS
                                                ;CLEAR THE RP11
                                                ;CLEAR THE INPUT BUFFER
                                                ;BEGINNING ADDRESS
                                                ;SIZE
                                                ;CLEAR THE BUFFER
                                                ;FINISHED ?
                                                ;BR IF NOT
                                                ;CHANGE BUFFER ADDRESS
                                                ;DOUBLE THE WORD COUNT
                                                ;ADD BUFFER BASE ADDRESS
                                                ;CHANGE COMMAND
                                                ;READ THE SECTOR
                                                ;WAIT FOR THE OPERATION TO COMPLETE
                                                ;STORE THE REGISTERS
                                                ;DID AN ERROR OCCUR ?
                                                ;BR IF NOT
                                                ;ERROR OCCURED READING THE TEST SECTOR
                                                ;BYPASS THE COMPARSION
                                                ;GOOD DATA
                                                ;SIZE
                                                ;FORM ADDRESS OF TEST DATA
                                                ;DOUBLE THE WORD COUNT
                                                ;ADD THE BUFFER START ADDRESS
                                                ;CLEAR THE COMPARSION ERROR COUNTER
                                                ;COMPARE THE DATA ?
                                                ;BR IF NOT
                                                ;COMPARE THE DATA
                                                ;BR IF OK
                                                ;GOOD DATA
                                                ;BAD DATA
                                                ;FORM BAD ADDRESS
                                                ;DECREMENT IT
                                                ;ANY COMPARE ERROR ALREADY ?
                                                ;BR IF SOME
                                                ;PC FOR ERROR MESSAGE
                                                ;ERROR TABLE INDEX
                                                ;REPORT COMPARSION ERROR
                                                ;INDEX OF DATA DISPLAY MESSAGE
                                                ;TYPE THE MESSAGE
                                                ;INCREMENT COMPARSION ERROR COUNT
                                                ;COMPARE AGAINST THE LIMIT
                                                ;BR IF NOT GREATER
```

```
                                        BIT     #SW04,@SWR      ;CONTINUE COMPARING ANYWAY ?
                                        BEQ     TST13A          ;BR IF NOT
                                9$:     DEC     R2              ;FINISHED COMPARING ?
                                        BNE     6$              ;BR IF NOT
                                        BR      TST13A          ;CHANGE MODE OF THE SECTOR
                                10$:    .WORD   0               ;COMPARSION ERROR COUNTER
013474  012737  013474  001110  TST13A: MOV     #TST13A,$LPERR  ;CHANGE LOOP ON ERROR ADDRESS
013502  004737  025654          JSR     PC,CLRP         ;CLEAR THE RP11
013506  105037  002503          CLRB    DPB+3           ;CLEAR THE 'MODE' & 'HDR' BITS
013512  012737  000002  002504  MOV     #2,$WC          ;CHANGE WORD COUNT
013520  112737  000003  002502  MOVB    #WRTSEK,DPB+2   ;CHANGE COMMAND
013526  005037  043656          CLR     BUFFER          ;SET DATA TO ZERO
013532  005037  043660          CLR     BUFFER+2        ;OTHER DATA WORD
013536  012737  043656  002506  MOV     #BUFFER,$BUF    ;BUFFER ADDRESS
013544  004737  022770          JSR     PC,RP11         ;RE-WRITE THE SECTOR
013550  004737  023664          JSR     PC,CONRDY       ;WAIT FOR THE OPERATION TO COMPLETE
013554  004737  023240          JSR     PC,SAVRP        ;STORE THE REGISTERS
013560  032737  100000  001340  BIT     #ERR,$RPCS      ;DID THE WRITE FINISH WITHOUT ERROR ?
013566  001401                  BEQ     EXIT13          ;BR IF IT DID
013570  104060                  ERROR   60              ;ERROR RE-WRITING THE SECTOR
013572  000004          EXIT13: SCOPE                   ;LOOP ?

;;*****************************************************************
;*TEST 14        DATA STORAGE/RETRIEVAL TEST

;*THE DATA TEST PERFORMS DATA STORAGE AND RETRIEVAL TESTING USING THE CONTROLLER
;*AND THE CURRENTLY SELECTED DRIVE.  UNLESS ALTERED BY THE OPERATOR, THE
;*TEST WILL WRITE AND WRITE CHECK THE ENTIRE DISK PACK USING ALL 16 DATA
;*PATTERNS IN A ROTATING MANNER.  THE BUFFER SIZE USED FOR THE WRITE AND
;*WRITE CHECK ORDERS IS DETERMINED BY THE AVAILABLE MEMORY ON THE SYSTEM
;*(MINUS THE SPACE REQUIRED BY THE PROGRAM LOADER).  THE PROGRAM WILL USE
;*THE AVAILABLE MEMORY SIZE OR 8K, WHICHEVER IS LESS, AS THE BUFFER
;*SIZE.
;*
;*AFTER THE PACK HAS BEEN WRITTEN, THE TEST WILL READ AND COMPARE THE
;*DATA.  THE BUFFER SIZE USED FOR THE READ IS THE SAME SIZE AS THAT USED
;*FOR THE WRITE/WRITE CHECK PART OF THE TEST.  FOR THE READ, THE TEST WILL
;*ROTATE THE READ BUFFER THROUGH MEMORY SO THAT ALL OF THE AVAILABLE MEMORY
;*IS USED BY THE READ AND COMPARE OPERATIONS.  THE TEST USES MEMORY MANAGEMENT
;*ON SYSTEMS WITH MORE THAN 28K.  (MEMORY MANAGEMENT WILL NOT BE USED IF
;*ON MEMORY MANAGEMENT SYSTEMS WITH 28K OR LESS.)
;*
;*THE DATA PATTERNS USED BY THE DATA TEST ARE GIVEN BELOW:
;*
;*      PAT 0   PAT 1   PAT 2   PAT 3   PAT 4   PAT 5   PAT 6   PAT 7
;*      ------  ------  ------  ------  ------  ------  ------  ------
;*      000000  000001  177776  000000  000000  052525  007417  026455
;*      000000  000003  177774  000000  010421  052525  007417  026455
;*      000000  000007  177770  000000  021042  052525  007417  026455
;*      000000  000017  177760  177777  031463  125252  170360  151322
;*      000000  000037  177740  177777  042104  125252  170360  151322
;*      000000  000077  177700  177777  052525  125252  170360  151322
;*      000000  000177  177600  000000  063146  052525  007417  026455
;*      000000  000377  177400  000000  073567  052525  007417  026455
;*      000000  000777  177000  177777  104210  125252  170360  151322
;*      000000  001777  176000  177777  114631  125252  170360  151322
```

```
                                ;*      000000  003777  174000  000000  125252  052525  007417  026455
                                ;*      000000  007777  170000  177777  135673  125252  170360  151322
                                ;*      000000  017777  160000  000000  146314  052525  007417  026455
                                ;*      000000  037777  140000  177777  156735  125252  170360  151322
                                ;*      000000  077777  100000  177777  167356  125252  170360  151322
                                ;*      000000  177777  000000  000000  177777  052525  007417  026455
                                ;*
                                ;*      PAT 8   PAT 9   PAT 10  PAT 11  PAT 12  PAT 13  PAT 14  PAT 15
                                ;*      ------  ------  ------  ------  ------  ------  ------  ------
                                ;*      077577  000001  177776  172666  077777  177777  000000  177777
                                ;*      077577  000002  177775  155555  137777  177777  000000  177777
                                ;*      077577  000004  177773  172666  157777  177777  177777  000000
                                ;*      077577  000010  177767  155555  167777  177777  177777  000000
                                ;*      077577  000020  177757  172666  173777  177777  177777  000000
                                ;*      077577  000040  177737  155555  175777  177777  177777  000000
                                ;*      077577  000100  177677  172666  176777  177777  177777  000000
                                ;*      077577  000200  177577  155555  177377  177777  177777  000000
                                ;*      077577  000400  177377  172666  177577  177777  177777  000000
                                ;*      077577  001000  176777  155555  177677  177777  177777  000000
                                ;*      077577  002000  175777  172666  177737  177777  177777  000000
                                ;*      077577  004000  173777  155555  177757  177777  177777  000000
                                ;*      077577  010000  167777  172666  177767  177777  177777  000000
                                ;*      077577  020000  157777  155555  177773  177777  177777  000000
                                ;*      077577  040000  137777  172666  177775  177777  177777  000000
                                ;*      077577  100000  077777  155555  177776  177777  177777  000000
                                ;*
                                ;;*****************************************************************
013574                          ;TST14:
013574  033737  001430  001276          BIT     BITS+<$TN#2>,TSTNMS  ;IS THIS TEST SELECTED
013602  001002                          BNE     .+6                  ;BR IF YES
013604  000137  014630                  JMP     @#TST15              ;GO TO THE NEXT TEST
013610  012737  014066  001110          MOV     #TEST14,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
013616  012737  013574  001106          MOV     #TST14,$LPADR        ;SETUP THE SCOPE LOOP ADDRESS
013624  012737  000014  001102          MOV     #14,$TSTNM           ;SETUP TEST NUMBER AND
                                                                     ;CLEAR THE ERROR FLAG ($ERFLG)
013632  013777  001102  165302          MOV     $TSTNM,@#DISPLAY     ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
013640  012737  014616  001200          MOV     #EXIT14,BYPASS       ;SETUP BYPASS ADDRESS
013646  013704  001324                  MOV     RPADR,R4             ;RP11E BUS ADDRESS
013652  105037  002501                  CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
013656  105037  002503                  CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
013662  012706  001100                  MOV     #STACK,SP            ;SETUP THE STACK POINTER
013666  032777  000100  165244          BIT     #SW06,@#SWR          ;USE MEMORY MANAGEMENT ?
013674  001021                          BNE     2$                   ;BR IF NOT
013676  005737  022232          1$:     TST     $KT11                ;MEMORY MANAGEMENT AVAILABLE ?
013702  100016                          BPL     2$                   ;BR IF NOT
013704  012737  177777  001222          MOV     #-1,MMACTV           ;SET MEMORY MANAGEMENT ACTIVE INDICATOR
013712  004737  023346                  JSR     PC,MOVLDR            ;MOVE THE LOADER
013716  012637  001240                  MOV     (SP)+,LOMEM          ;BEGINNING ADDRESS OF BUFFER SPACE
013722  013737  001230  001234          MOV     MAXMEM,HIMEM         ;HIGHEST MEMORY ADDRESS USED BY THE TEST
013730  013737  001232  001236          MOV     MAXMEM+2,HIMEM+2     ;UPPER ADDRESS BITS
013736  000412                          BR      3$                   ;CONTINUE
013740  005037  001222          2$:     CLR     MMACTV               ;CLEAR MEMORY MANAGEMENT ACTIVE INDICATOR
013744  005037  001236                  CLR     HIMEM+2              ;CLEAR UPPER 'HIMEM' ADDRESS BITS
013750  013737  001302  001234          MOV     MEMSIZ,HIMEM         ;HIGHEST NON MEMORY MANAGENEMT LOCATION
013756  012737  043656  001240          MOV     #BUFFER,LOMEM        ;DEFAULT BUFFER STARTING ADDRESS
```

```
3494  013764  005037  001242            3$:     CLR     LOMEM+2         ;CLEAR UPPER STORAGE WORD
3495  013770  013737  001240  001224            MOV     LOMEM,ACTMEM    ;STARTING ADDRESS OF FIRST BUFFER
3496  013776  013737  001242  001226            MOV     LOMEM+2,ACTMEM+2 ;UPPER ADDRESS BITS
3497  014004  013737  001256  001254            MOV     WC14,BLKS14     ;CONVERT THE WORD COUNT TO A SECTOR COUNT
3498  014012  000337  001254                    SWAB    BLKS14          ;RIGHT JUSTIFY THE SECTOR COUNT
3499  014016  105037  001255                    CLRB    BLKS14+1        ;CLEAR THE RESIDUE
3500  014022  005237  001254                    INC     BLKS14          ;CHANGE TO ABSOLUTE COUNT
3501  014026  013746  001254                    MOV     BLKS14,-(SP)    ;CONVERT BLOCK COUNT TO SECTOR/TRACK INCREMENTS
3502  014032  005037  001252                    CLR     INCTRK          ;CLEAR TRACK STORAGE
3503  014036  005037  001250                    CLR     INCSEC          ;CLEAR SECTOR STORAGE
3504  014042  162716  000012            4$:     SUB     #10.,(SP)       ;SUBTRACT A FULL SECTOR
3505  014046  100403                    BMI     5$              ;BR IF MINUS
3506  014050  005237  001252                    INC     INCTRK          ;COUNT THE TRACK
3507  014054  000772                    BR      4$              ;CONTINUE
3508  014056  062716  000012            5$:     ADD     #10.,(SP)       ;CORRECT THE SECTOR RESIDUE
3509  014062  012637  001250                    MOV     (SP)+,INCSEC    ;SECTOR INCREMENT VALUE
3510  014066  012706  001100            TEST14: MOV     #STACK,SP       ;LOAD THE STACK POINTER
3511  014072  004737  024002                    JSR     PC,SECNBR       ;GET THE NUMBER OF SECTORS ON THE PACK
3512                                                                    ;BEING USED
3513  014076  113737  001272  002512    1$:     MOVB    $SEC,$SEC       ;STARTING SECTOR ADDRESS
3514  014104  113737  001270  002513            MOVB    $TRK,$TRK       ;STARTING TRACK ADDRESS
3515  014112  013737  001266  002510            MOV     $CYL,$CYL       ;STARTING CYLINDER ADDRESS
3516  014120  013737  001240  002506            MOV     LOMEM,$BUF      ;BUFFER ADDRESS
3517  014126  013737  001256  002504            MOV     WC14,$WC        ;STARTING WORD COUNT
3518  014134  012737  000017  001204            MOV     #15.,PATNUM     ;PRESET THE PATTERN INDEX
3519  014142  005737  001260                    TST     OPRSEL          ;USING AN OPERATOR SPECIFIED ADDRESS ?
3520  014146  001005                    BNE     2$              ;BR IF YES
3521  014150  163737  001254  001244            SUB     BLKS14,PAKSIZ   ;DECREMENT THE SECTOR COUNT FOR THE
3522  014156  005637  001246                    SBC     PAKSIZ+2        ;FIRST OPERATION
3523  014162  004737  024050            2$:     JSR     PC,PATSEL       ;SELECT THE PATTERN
3524  014166  012737  014166  001110    3$:     MOV     #3$,$LPERR      ;LOOP ON ERROR ADDRESS
3525  014174  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
3526  014200  017746  164734                    MOV     @SWR,-(SP)      ;INHIBIT WRITE AND
3527  014204  005116                    COM     (SP)            ;WRITE CHECK ?
3528  014206  032726  000003                    BIT     #SW00!SW01,(SP)+
3529  014212  001453                    BEQ     TST14A          ;YES--BRANCH
3530  014214  004737  024432                    JSR     PC,SETBUF       ;MOVE DATA PATTERN INTO THE BUFFER
3531  014220  032777  000001  164712            BIT     #SW00,@SWR      ;INHIBIT WRITE ?
3532  014226  001016                    BNE     4$              ;YES--BRANCH
3533  014230  112737  000003  002502            MOVB    #WRTSEK,DPB+2   ;COMMAND=WRITE
3534  014236  004737  022770                    JSR     PC,RP11         ;DO THE COMMAND
3535  014242  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE COMMAND TO COMPLETE
3536  014246  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
3537  014252  032737  100000  001340            BIT     #ERR,$RPCS      ;ERROR OCCURED ?
3538  014260  001401                    BEQ     4$              ;BR IF NOT
3539  014262  104061                    ERROR   61              ;REPORT THE ERROR
3540  014264  032777  000002  164646    4$:     BIT     #SW01,@SWR      ;INHIBIT WRITE CHECK ?
3541  014272  001020                    BNE     5$              ;YES--BRANCH
3542  014274  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
3543  014300  112737  000007  002502            MOVB    #WRTCHK,DPB+2   ;COMMAND=WRITE CHECK
3544  014306  004737  022770                    JSR     PC,RP11         ;DO THE WRITE CHECK
3545  014312  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE COMMAND TO FINISH
3546  014316  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
3547  014322  032737  100000  001340            BIT     #ERR,$RPCS      ;ERROR ?
3548  014330  001401                    BEQ     5$              ;BR IF NOT
3549  014332  104005                    ERROR   5               ;REPORT THE WRITE CHECK ERROR
```

```
3550  014334  004737  024114          5$:     JSR     PC,INCADR       ;INCREMENT THE ADDRESS
3551  014340  000710                  BR      2$                      ;CONTINUE
3552  014342  004737  024002  TST14A: JSR     PC,SECNBR       ;GET NUMBER OF SECTORS ON THE DRIVE
3553                                                                  ;BEING TESTED
3554  014346  113737  001272  002512  MOVB    SSEC,$SEC       ;STARTING SECTOR
3555  014354  113737  001270  002513  MOVB    STRK,$TRK       ;STARTING TRACK
3556  014362  013737  001266  002510  MOV     SCYL,$CYL       ;STARTING CYLINDER
3557  014370  013737  001256  002504  MOV     WC14,$WC        ;INITIAL WORD COUNT
3558  014376  012737  000017  001204  MOV     #15.,PATNUM     ;PRESET PATTERN NUMBER
3559  014404  005737  001260          TST     OPRSEL          ;USING AN OPERATOR SPECIFIED PATTERN
3560  014410  001005                  BNE     1$              ;BR IF YES
3561  014412  163737  001254  001244  SUB     BLKS14,PAKSIZ   ;DECREMENT THE SECTOR COUNT FOR
3562  014420  005637  001246          SBC     PAKSIZ+2        ;THE FIRST OPERATION
3563  014424  005037  001314          1$:     CLR     RETRY           ;CLEAR THE RETRY COUNTER
3564  014430  004737  024050          JSR     PC,PATSEL       ;SELECT A PATTERN
3565  014434  004737  024502          JSR     PC,GETBUF       ;GENERATE A BUFFER ADDRESS
3566  014440  012737  014440  001110  2$:     MOV     #2$,$LPERR      ;LOOP ON ERROR ADDRESS
3567  014446  004737  025654          JSR     PC,CLRP         ;CLEAR THE RP11
3568  014452  032777  000004  164460  BIT     #SW02,@SWR      ;INHIBIT READ DATA AND SOFTWARE COMPARE ?
3569  014460  001056                  BNE     EXIT14          ;YES--BRANCH
3570  014462  112737  000005  002502  MOVB    #RDSEK,DPB+2    ;LOAD READ COMMAND
3571  014470  004737  022770          JSR     PC,RP11         ;START THE READ
3572  014474  004737  023664          JSR     PC,CONRDY       ;WAIT FOR THE READ TO COMPLETE
3573  014500  004737  023240          JSR     PC,SAVRP        ;STORE THE REGISTERS
3574  014504  032737  100000  001340  BIT     #ERR,$RPCS      ;ERROR ?
3575  014512  001420                  BEQ     5$              ;BR IF NOT
3576  014514  005737  001314          TST     RETRY           ;RETRY ATTEMPT ?
3577  014520  001404                  BEQ     3$              ;BR IF NOT
3578  014522  032777  000040  164410  BIT     #SW05,@SWR      ;DISPLAY ALL RETRY ATTEMPTS
3579  014530  001401                  BEQ     4$              ;BR IF NOT
3580  014532  104062                  3$:     ERROR   62              ;REPORT THE ERROR
3581  014534  005237  001314          4$:     INC     RETRY           ;INCREMENT THE RETRY COUNT
3582  014540  023737  001314  001316  CMP     RETRY,LRETRY    ;RETRY LIMIT ?
3583  014546  101734                  BLOS    2$              ;BR IF NOT
3584  014550  104110                  ERROR   110             ;REPORT UNSUCESSFUL RETRY
3585  014552  000410                  BR      6$              ;CONTINUE
3586  014554  005737  001314          5$:     TST     RETRY           ;COMING FROM A SUCESSFUL RETRY ?
3587  014560  001405                  BEQ     6$              ;BR IF NOT
3588  014562  112737  000111  001114  MOVB    #111,$ITEMB     ;RETRY COUNT MESSAGE NUMBER
3589  014570  004737  017254          JSR     PC,TYPERR       ;REPORT THE RETRY COUNT
3590  014574  032777  000010  164336  6$:     BIT     #SW03,@SWR      ;COMPARE THE DATA ?
3591  014602  001002                  BNE     7$              ;NO--BRANCH
3592  014604  004737  024614          JSR     PC,DATCMP       ;YES--DO IT
3593  014610  004737  024114          7$:     JSR     PC,INCADR       ;INCREMENT THE DISK ADDRESS
3594  014614  000703                  BR      1$              ;CONTINUE
3595  014616  000004                  EXIT14: SCOPE                   ;LOOP ?
3596  014620  004737  023452          JSR     PC,RESLDR       ;RESTORE THE LOADER IF IT WAS MOVED
3597  014624  000137  016616          JMP     $EOP            ;GO TO END OF TEST
3598
3599                                  ;;*****************************************************************
3600                                  ;*TEST 15        POWER FAIL TEST
3601
3602                                  ;*TEST THE ABILITY OF THE RP11E TO SENSE POWER FAILURE AND FOR THE DRIVES
3603                                  ;*TO RETRACT THEIR HEADS.  WHEN POWER IS TURNED ON AGAIN
3604                                  ;*THE CYLINDER ADDRESS IS TESTED FOR ZERO.  AFTER TYPING THE MESSAGE
3605                                  ;*REQUESTING POWER TO BE TURNED OFF, THE PROGRAM GOES INTO A LOOP
```

```
                                        ;*READING FROM THE SELECTED DISK DRIVE.  AFTER POWER IS RESTORED,
                                        ;*MEMORY IS CHECKED TO SEE THAT THE DISK DID NOT TRANSFER ANY DATA
                                        ;*TO MEMORY WHILE POWER WAS GOING DOWN.

                                        ;;*****************************************************************
014630                                  TST15:
014630 033737 001432 001276                     BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
014636 001002                                   BNE     .+6                  ;BR IF YES
014640 000137 015372                            JMP     TST16                ;GO TO THE NEXT TEST
014644 012737 014736 001110                     MOV     #TEST15,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
014652 012737 014630 001106                     MOV     #TST15,$LPADR        ;SETUP THE SCOPE LOOP ADDRESS
014660 012737 000015 001102                     MOV     #15,$TSTNM           ;SETUP TEST NUMBER AND
                                                                             ;CLEAR THE ERROR FLAG ($ERFLG)
014666 013777 001102 164246                     MOV     $TSTNM,@DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
014674 012737 015302 001200                     MOV     #EXIT15,BYPASS       ;SETUP BYPASS ADDRESS
014702 013704 001324                            MOV     RPADR,R4             ;RP11E BUS ADDRESS
014706 105037 002501                            CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
014712 105037 002503                            CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
014716 104401 027433                            TYPE    ,MSG15A              ;'POWER FAIL TEST'
014722 004737 023346                            JSR     PC,MOVLDR            ;MOVE THE LOADER
014726 012637 001240                            MOV     (SP)+,LOMEM          ;BUFFER STARTING ADDRESS
014732 005037 001242                            CLR     LOMEM+2              ;CLEAR THE UPPER BITS JUST TO BE NICE
014736 012706 001100                    TEST15: MOV     #STACK,SP            ;SETUP THE STACK POINTER
014742 004737 025654                            JSR     PC,CLRP              ;CLEAR THE RP11
014746 012737 015314 000024                     MOV     #TST15B,@#24         ;SET UP POWER FAIL VECTOR
014754 012737 000340 000026                     MOV     #PR7,@#26            ;LOCKOUT INTERRUPTS
014762 013700 001240                            MOV     LOMEM,R0             ;START OF FREE MEMORY SPACE
014766 012720 125252                    1$:     MOV     #125252,(R0)+        ;FILL MEMORY WITH CHECKERBOARD
014772 020037 001302                            CMP     R0,MEMSIZ            ;PATTERN
014776 101773                                   BLOS    1$                   ;BR IF NOT FINISHED
015000 013737 001210 002510                     MOV     MAXCYL,$CYL          ;LOAD MAXIMUM CYLINDER
015006 005037 002512                            CLR     $SEC                 ;CLEAR SECTOR/TRACK
015012 012737 000256 002504                     MOV     #256,$WC             ;WORD COUNT = 1 SECTOR
015020 013737 001240 002506                     MOV     LOMEM,$BUF           ;BUFFER ADDRESS
015026 112737 000003 002502                     MOVB    #WRTSEK,DPB+2        ;WRITE OPERATION
015034 004737 022770                            JSR     PC,RP11              ;START THE COMMAND
015040 004737 023664                            JSR     PC,CONRDY            ;WAIT FOR THE CONTROLLER TO BE READY
015044 004737 023240                            JSR     PC,SAVRP             ;STORE THE REGISTERS
015050 032737 100000 001340                     BIT     #ERR,$RPCS           ;ERROR DURING OPERATION ?
015056 001401                                   BEQ     2$                   ;BR IF NOT
015060 104004                                   ERROR   4                    ;REPORT THE ERROR
015062 104401 027456                    2$:     TYPE    ,MSG15B              ;'TURN CPU POWER OFF AFTER MESSAGE'
015066 004737 025654                    3$:     JSR     PC,CLRP              ;CLEAR THE RP11
015072 112737 000005 002502                     MOVB    #RDSEK,DPB+2         ;READ COMMAND
015100 004737 022770                            JSR     PC,RP11              ;INITIATE THE READ
015104 004737 023664                            JSR     PC,CONRDY            ;WAIT FOR CONTROLLER READY
015110 000766                                   BR      3$                   ;LOOP, READING THE DISK

                                        ;AFTER MACHINE IS POWERED DOWN AND UP, CONTROL IS TRANSFERRED HERE.

015112 004737 022566                    TST15A: JSR     PC,RPINIT            ;INITIALZIE THE RP11E
015116 104401 027652                            TYPE    ,MSG15D              ;'PROGRAM WILL LOOP, WAITING FOR DRIVE TO COME ONLINE'
015122 012737 015122 001110             1$:     MOV     #1$,$LPERR           ;SETUP LOOP ON ERROR ADDRESS
015130 012737 002500 001220                     MOV     #2500,STALLT         ;LOAD STALL VALUE
015136 004737 023202                            JSR     PC,STALL             ;STALL FOR 2500 MILLISECONDS
015142 013701 001176                            MOV     CHKDRV,R1            ;DRIVE ADDRESS
```

```
3662  015146  004737  022666                  JSR     PC,DRVINT       ;CHECK THE DRIVE'S STATUS
3663  015152  105761  001370                  ISIB    DRVSTA(R1)      ;IS THE DRIVE ONLINE ?
3664  015156  001761                          BEQ     1$              ;CONTINUE WAITING FOR DRIVE TO COME ONLINE
3665  015160  003002                          BGT     2$              ;BR IF IT IS ONLINE
3666  015162  104112                          ERROR   112             ;DRIVE UNSAFE AFTER POWER FAIL
3667  015164  000446                          BR      EXIT15          ;DRIVE UNSAFE, EXIT
3668  015166  113764  001176  000005  2$:     MOVB    CHKDRV,RPCS+1(R4)   ;SELECT THE DRIVE
3669  015174  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
3670  015200  005737  001354                  TST     $SUCA           ;SELECTED CYLINDER SHOULD BE ZERO
3671  015204  001404                          BEQ     3$              ;BR IF IT DID
3672  015206  012737  015216  001164          MOV     #3$,$ESCAPE     ;ESCAPE TO 3$ ON ERROR
3673  015214  104063                          ERROR   63              ;DRIVE DID NOT RETURN TO CYLINDER 0
3674                                                                  ;ON POWER FAILURE
3675  015216                          3$:
3676  015216  012737  015276  001164          MOV     #6$,$ESCAPE     ;ESCAPE TO 6$ ON ERROR
3677  015224  013700  001240                  MOV     LOMEM,R0        ;BUFFER ADDRESS
3678  015230  022720  125252          4$:     CMP     #125252,(R0)+   ;DID MEMORY RETAIN PATTERN ON POWER FAIL ?
3679  015234  001004                          BNE     5$              ;BRANCH IF NOT
3680  015236  020037  001302                  CMP     R0,MEMSIZ       ;END OF MEMORY ?
3681  015242  001372                          BNE     4$              ;BR IF NOT
3682  015244  000414                          BR      6$              ;EXIT
3683  015246  012737  125252  001124  5$:     MOV     #125252,$GDDAT  ;EXPECTED MEMORY CONTENTS
3684  015254  016037  177776  001126          MOV     -2(R0),$BDDAT   ;ACTUAL MEMORY CONTENTS
3685  015262  010037  001122                  MOV     R0,$BDADR       ;ADDRESS
3686  015266  162737  000002  001122          SUB     #2,$BDADR       ;CORRECT IT
3687  015274  104064                          ERROR   64              ;CONTENTS OF MEMORY CHANGED AFTER POWER FAIL
3688  015276  104401  027617          6$:     TYPE    ,MSG15C         ;'END OF POWER FAIL TEST'
3689  015302  000004                  EXIT15: SCOPE                   ;LOOP ?
3690  015304  004737  023452                  JSR     PC,RESLDR       ;RESTORE THE LOADER
3691  015310  000137  003676                  JMP     START2          ;RETURN TO 'CONVERSATION MODE' ENTRANCE
3692
3693                                  ;POWER FAIL TRAP HANDLER
3694
3695  015314  012737  015324  000024  TST15B: MOV     #TST15C,@#24    ;SET UP POWER UP VECTOR
3696  015322  000774                          BR      TST15B          ;LOOP
3697
3698                                  ;POWER UP TRAP HANDLER
3699
3700  015324  012737  000026  000024  TST15C: MOV     #26,@#24        ;RESTORE TRAP CATCHER
3701  015332  005037  000026                  CLR     @#26            ;CATCHER HALT
3702  015336  012706  001100                  MOV     #STACK,SP       ;SETUP STACK POINTER
3703  015342  005037  177776                  CLR     PS              ;RETURN TO PRIORITY ZERO
3704  015346  012737  011610  001220          MOV     #5000.,STALLT   ;LOAD STALL VALUE
3705  015354  004737  023202                  JSR     PC,STALL        ;STALL FOR 5000. MILLISECONDS
3706  015360  004737  020416                  JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
3707  015364  013704  001324                  MOV     RPADR,R4        ;RP11E ADDRESS
3708  015370  000650                          BR      TST15A          ;VERIFY POWER DOWN SEQUENCE
3709
3710
3711                                  ;;*****************************************************************
3712                                  ;*TEST 16        HEAD ALIGNMENT ROUTINE
3713
3714                                  ;*THIS ROUTINE PERFORMS HEAD SELECTION AS DIRECTED FROM THE KEYBOARD.
3715                                  ;*THE ALIGNMENT OF THE SELECTED HEAD MAY BE CHECKED OR ADJUSTED.
3716
3717                                  ;;*****************************************************************
```

```
                                TST16:
015372 033737 001434 001276             BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
015400 001002                           BNE     .+6                  ;BR IF YES
015402 000137 015624                    JMP     @#ST17               ;GO TO THE NEXT TEST
015406 012737 015460 001110             MOV     #TEST16,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
015414 012737 015372 001106             MOV     #TST16,$LPADR        ;SETUP THE SCOPE LOOP ADDRESS
015422 012737 000016 001102             MOV     #16,$TSTNM           ;SETUP TEST NUMBER AND
                                                                     ;CLEAR THE ERROR FLAG ($ERFLG)
015430 013777 001102 163504             MOV     $TSTNM,@DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
015436 012737 015620 001200             MOV     #EXIT16,BYPASS       ;SETUP BYPASS ADDRESS
015444 013704 001324                    MOV     RPADR,R4             ;RP11E BUS ADDRESS
015450 105037 002501                    CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
015454 105037 002503                    CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
015460 012706 001100            TEST16: MOV     #STACK,SP            ;SETUP THE STACK POINTER
015464 104401 027743                    TYPE    ,MSG16A              ;'HEAD ALIGNMENT ROUTINE'
015470 000410                           BR      3$                   ;GET THE HEAD ADDRESS WITHOUT THE SW07 NONESENSE
015472 032777 000200 163440     1$:     BIT     #SW07,@SWR           ;SWITCH 7 SET ?
015500 001774                           BEQ     1$                   ;BR IF NOT
015502 032777 000200 163430     2$:     BIT     #SW07,@SWR           ;SWITCH 7 RESET ?
015510 001374                           BNE     2$                   ;BR IF NOT
015512 104401 030255            3$:     TYPE    ,MSG16B              ;'ENTER HEAD ADDRESS (IN OCTAL)'
015516 104411                           RDLIN                        ;GET THE ENTRY
015520 012601                           MOV     (SP)+,R1             ;STARTING ADDRESS OF INPUT ASCII STRING
015522 004037 027136                    JSR     R0,CK.NUM            ;CHECK THE NUMBER
015526 015512                           3$                           ;ILLEGAL INPUT
015530 015534                           4$                           ;TERMINATED WITH A 'CR'
015532 015512                           3$                           ;NO ENTRY - 'CR' ONLY
015534 122702 000024            4$:     CMPB    #20.,R2              ;VALID ?
015540 101764                           BLOS    3$                   ;BR IF NOT
015542 110237 002513                    MOVB    R2,$TRK              ;LOAD NEW HEAD ADDRESS
015546 136237 001360 001304             BITB    @TABIT(R2),DRVTYP    ;RP02 OR RP03 ?
015554 001004                           BNE     5$                   ;BR IF RP03
015556 012737 000111 002510             MOV     #73.,$CYL            ;RP02 ALIGNMENT CYLINDER
015564 000403                           BR      6$                   ;CONTINUE
015566 012737 000222 002510     5$:     MOV     #146.,$CYL           ;RP03 ALIGNMENT CYLINDER
015574 112737 000011 002502     6$:     MOVB    #SEEK,DPB+2          ;SEEK COMMAND
015602 004737 025654                    JSR     PC,CLRP              ;CLEAR THE RP11
015606 004737 022770                    JSR     PC,RP11              ;SEEK AND LOAD THE HEAD
015612 004737 023546                    JSR     PC,DRVRDY            ;WAIT FOR THE DRIVE TO BECOME READY
015616 000725                           BR      1$                   ;WAIT FOR SWITCH 7 TO BE TOGGLED
015620 000000                   EXIT16: HALT                         ;'EXIT' FOR COMPATABILITY WITH OTHER TESTS
015622 000776                           BR      .-2                  ;LOCK THE HALT

                                ;;*******************************************************************
                                ;*TEST 17        RP11E/DRIVE CONTROL PANEL SWITCH TEST

                                ;*THIS TEST TEST THE 'ENABLE/DISABLE' AND 'READ ONLY' SWITCHES ON
                                ;*THE DRIVE CONTROL PANEL AND TESTS THE 'WRITE LOCKOUT' AND 'LOA'
                                ;*SWITCHES ON THE RP11E/DRIVE CONTROL PANEL.

                                ;;*******************************************************************
015624                          TST17:
015624 033737 001436 001276             BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
015632 001002                           BNE     .+6                  ;BR IF YES
015634 000137 016616                    JMP     @#$EOP               ;GO TO END OF TEST
```

```
                                        MOV     #TEST17,$LPERR  ;SETUP THE ERROR LOOP ADDRESS
                                        MOV     #TST17,$LPADR   ;SETUP THE SCOPE LOOP ADDRESS
                                        MOV     #17,$TSTNM      ;SETUP TEST NUMBER AND
                                                                ;CLEAR THE ERROR FLAG ($ERFLG)
                                        MOV     $TSTNM,@DISPLAY ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
                                        MOV     #EXIT17,BYPASS  ;SETUP BYPASS ADDRESS
                                        MOV     RPADR,R4        ;RP11E BUS ADDRESS
                                        CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
                                        CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
                                        TYPE    ,MSG17A         ;'RP11E CONTROL PANEL SWITCH TEST'
                                TEST17: MOV     #STACK,SP       ;SETUP THE STACK POINTER

                                ;CHECK THE ENABLE/DISABLE SWITCH ON THE DRIVE

                                        TYPE    ,MSG17B ;'TYPE A CR TO START TEST AFTER EACH SETUP'
                                2$:     TYPE    ,MSG17C         ;'DISABLE DRIVE # '
                                        MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   1               ;;TYPE 1 DIGIT(S)
                                        .BYTE   1               ;;TYPE LEADING ZEROS
                                        RDLIN                   ;WAIT FOR THE 'CR'
                                3$:     MOV     #3$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
                                        JSR     PC,CLRP         ;CLEAR THE RP11
                                        MOVB    CHKDRV,RPCS+1(R4)  ;SELECT DRIVE
                                        JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
                                        BIT     #SUOL,$RPDS     ;IS THE UNIT ON LINE?
                                        BEQ     4$              ;BRANCH IF NOT
                                        ERROR   65              ;SELECTED UNIT ON LINE WITH THE
                                                                ;DRIVE DISABLED
                                4$:     TYPE    ,MSG17D         ;'ENABLE DRIVE # '
                                        MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   1               ;;TYPE 1 DIGIT(S)
                                        .BYTE   1               ;;TYPE LEADING ZEROS
                                        RDLIN                   ;WAIT FOR THE 'CR'
                                5$:     MOV     #5$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
                                        JSR     PC,CLRP         ;CLEAR THE RP11
                                        MOVB    CHKDRV,RPCS+1(R4)  ;RESELECT THE UNIT
                                        JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
                                        BIT     #SUOL,$RPDS     ;IS THE UNIT ON LINE?
                                        BNE     RDSW            ;BR IF ONLINE
                                        ERROR   66              ;SELECTED UNIT NOT ON LINE WITH
                                                                ;THE DRIVE ENABLED


                                ;CHECK THE DRIVE'S WRITE LOCKOUT SWITCH

                                RDSW:   TYPE    ,MSG17E         ;'SET READ ONLY ON DRIVE '
                                        MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
                                        TYPOS                   ;;GO TYPE--OCTAL ASCII
                                        .BYTE   1               ;;TYPE 1 DIGIT(S)
                                        .BYTE   1               ;;TYPE LEADING ZEROS
                                        RDLIN                   ;WAIT FOR 'CR'
                                2$:     MOV     #2$,$LPERR      ;LOAD LOOP ON ERROR ADDRESS
```

```
3830  016102  004737  025654              JSR     PC,CLRP         ;CLEAR THE RP11
3831  016106  113764  001176  000005      MOVB    CHKDRV,RPCS+1(R4)  ;SELECT THE DRIVE
3832  016114  004737  023240              JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
3833  016120  032737  000400  001334      BIT     #SUWP,$RPDS     ;IS SELECTED UNIT WRITE PROTECTED ?
3834  016126  001001                      BNE     3$              ;BR IF IT IS
3835  016130  104067                      ERROR   67              ;SELECTED UNIT WRITE PROTECT DID
3836                                                              ;NOT SET WITH READ ONLY SET
3837  016132  104401  030542      3$:     TYPE    ,MSG17F         ;'SET READ/WRITE ON DRIVE '
3838  016136  013746  001176              MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
3839                                                              ;;TYPE DRIVE NUMBER
3840  016142  104403                      TYPOS                   ;;GO TYPE--OCTAL ASCII
3841  016144     001                      .BYTE   1               ;;TYPE 1 DIGIT(S)
3842  016145     001                      .BYTE   1               ;;TYPE LEADING ZEROS
3843  016146  104411                      RDLIN                   ;WAIT FOR 'CR'
3844  016150  012737  016150  001110  4$: MOV     #4$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
3845  016156  004737  025654              JSR     PC,CLRP         ;CLEAR THE RP11
3846  016162  113764  001176  000005      MOVB    CHKDRV,RPCS+1(R4)  ;RESELECT THE DRIVE
3847  016170  004737  023240              JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
3848  016174  032737  000400  001334      BIT     #SUWP,$RPDS     ;IS SELECTED UNIT WRITE PROTECTED ?
3849  016202  001401                      BEQ     WRSW            ;BR IF NOT
3850  016204  104070                      ERROR   70              ;SELECTED UNIT WRITE PROTECT SET
3851                                                              ;WITH DRIVE WRITE ENABLED
3852
3853                          ;TEST THE SETTING OF WRITE LOCKOUT AND THE 'LOA' SWITCHES ON THE
3854                          ;RP11E CONTROL PANEL.
3855
3856  016206  104401  030577  WRSW:   TYPE    ,MSG17G         ;'SET WRITE LOCKOUT ' ,ETC
3857  016212  104411                      RDLIN                   ;WAIT FOR 'CR' FROM THE OPERATOR
3858  016214  012737  016214  001110  1$: MOV     #1$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
3859  016222  005064  000012              CLR     RPCA(R4)        ;CLEAR THE CYLINDER REGISTER
3860  016226  004737  023240              JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
3861  016232  032737  000400  001334      BIT     #SUWP,$RPDS     ;IS SELECTED UNIT WRITE PROTECT SET ?
3862  016240  001001                      BNE     2$              ;BR IF SET
3863  016242  104071                      ERROR   71              ;SELECTED UNIT WRITE PROTECT SHOULD
3864                                                              ;BE SET - RPCA EQUAL TO ZERO
3865  016244  012764  000002  000012  2$: MOV     #2,RPCA(R4)     ;LOAD A 2 INTO RPCA
3866  016252  004737  023240              JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
3867  016256  032737  000400  001334      BIT     #SUWP,$RPDS     ;IS SELECTED UNIT WRITE PROTECTED ?
3868  016264  001401                      BEQ     WRSW1           ;BR IF NOT PROTECTED
3869  016266  104072                      ERROR   72              ;SELECTED UNIT WRITE PROTECT IS SET
3870                                                              ;WITH EQUAL TO 2.
3871
3872                          ;CHECK THE CYLINDER 'LOA' SWITCHES
3873
3874  016270  012700  000002  WRSW1:  MOV     #2,R0           ;INITIALIZE SWITCH PATTERN
3875  016274  104401  030714              TYPE    ,MSG17H         ;'SET FOLLOWING VALUES IN  THE
3876                                                              ;CYLINDER LOA SWITCHES'
3877  016300  104401  031014      1$:     TYPE    ,MSG17I         ;'SET '
3878  016304  006200                      ASR     R0              ;SETUP PATTERN FOR TYPOUT
3879  016306  010046                      MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
3880                                                              ;;TYPE THE SWITCH VALUE
3881  016310  104403                      TYPOS                   ;;GO TYPE--OCTAL ASCII
3882  016312     003                      .BYTE   3               ;;TYPE 3 DIGIT(S)
3883  016313     001                      .BYTE   1               ;;TYPE LEADING ZEROS
3884  016314  006300                      ASL     R0              ;RESTORE THE PATTERN
3885  016316  104411                      RDLIN                   ;WAIT FOR 'CR' FROM THE OPERATOR
```

```
3886  016320  012737  016320  001110  2$:     MOV     #2$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
3887  016326  010064  000012                  MOV     R0,RPCA(R4)     ;LOAD PATTERN INTO RPCA
3888  016332  004737  023240                  JSR     PC,$AVRP        ;SAVE THE RP11 REGISTERS
3889  016336  032737  000400  001334          BIT     #SUWP,$RPDS     ;IS UNIT WRITE PROTECT SET ?
3890  016344  001001                          BNE     3$              ;BR IF SET
3891  016346  104073                          ERROR   73              ;UNIT WRITE PROTECT IS NOT SET
3892                                                                  ;RPCA EQUALS CONTENTS OF BOUNDARY
3893                                                                  ;SWITCHES
3894  016350  062764  000002  000012  3$:     ADD     #2,RPCA(R4)     ;INCREMENT CYLINDER ADDRESS
3895  016356  012737  016356  001110  4$:     MOV     #4$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
3896  016364  004737  023240                  JSR     PC,$AVRP        ;SAVE THE RP11 REGISTERS
3897  016370  032737  000400  001334          BIT     #SUWP,$RPDS     ;IS UNIT WRITE PROTECT SET ?
3898  016376  001401                          BEQ     5$              ;BR IF NOT
3899  016400  104074                          ERROR   74              ;UNIT WRITE PROTECT IS SET WITH
3900                                                                  ;RPCA ONE GREATER THAN BOUNDARY
3901                                                                  ;SWITCH SETTINGS
3902  016402  006300                  5$:     ASL     R0              ;UPDATE TEST PATTERN
3903  016404  032700  001000                  BIT     #BIT09,R0       ;IS PATTERN EXCEEDED?
3904  016410  001733                          BEQ     1$              ;BRANCH IF NOT
3905
3906                                          ;CHECK DRIVE 'LOA' SWITCHES
3907
3908  016412  005037  001176          WRSW2:  CLR     CHKDRV          ;START WITH DRIVE 0
3909  016416  104401  031037                  TYPE    ,MSG17J         ;'RESET THE CYLINDER LOA SWITCHES
3910                                                                  ;'SET THE FOLLOWING VALUES IN THE DRIVE
3911                                                                  ;'LOA SWITCHES '
3912  016422  012764  000002  000012          MOV     #2,RPCA(R4)     ;LOAD CYLINDER ADDRESS
3913  016430  113764  001176  000005  1$:     MOVB    CHKDRV,RPCS+1(R4)  ;SELECT THE FIRST DRIVE NUMBER
3914  016436  104401  031014                  TYPE    ,MSG17I         ;'SET SWITCHES TO '
3915  016442  013746  001176                  MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
3916                                                                  ;;TYPE SWITCH SETTING
3917  016446  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3918  016450     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
3919  016451     001                          .BYTE   1               ;;TYPE LEADING ZEROS
3920  016452  104411                          RDLIN                   ;WAIT FOR THE OPERATOR TO ENTER A 'CR'
3921  016454  012737  016454  001110  2$:     MOV     #2$,$LPERR      ;LOOP ON ERROR ADDRESS
3922  016462  004737  023240                  JSR     PC,$AVRP        ;STORE THE RP11 REGISTERS
3923  016466  032737  000400  001334          BIT     #SUWP,$RPDS     ;IS THE SELECTED DRIVE WRITE PROTECTED ?
3924  016474  001401                          BEQ     3$              ;BR IF NOT
3925  016476  104075                          ERROR   75              ;'SUWP' SET WHEN DRIVE LOA SWITCHES
3926                                                                  ;'EQUAL THE SELECTED DRIVE'
3927  016500  005737  001176          3$:     TST     CHKDRV          ;THROUGH HERE THE FIRST TIME ?
3928  016504  001004                          BNE     4$              ;BR IF NOT
3929  016506  012737  000001  001176          MOV     #1,CHKDRV       ;SET FOR DRIVE 1
3930  016514  000406                          BR      5$              ;CONTINUE
3931  016516  006337  001176          4$:     ASL     CHKDRV          ;SHIFT TEST ADDRESS
3932  016522  032737  000010  001176          BIT     #BIT03,CHKDRV   ;FINISHED WITH ADDRESSES ?
3933  016530  001025                          BNE     EXIT17          ;BR IF FINISHED
3934  016532  104401  031014          5$:     TYPE    ,MSG17I         ;'SET SWITCHES TO '
3935  016536  013746  001176                  MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
3936                                                                  ;;TYPE SWITCH SETTING
3937  016542  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3938  016544     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
3939  016545     001                          .BYTE   1               ;;TYPE LEADING ZEROS
3940  016546  104411                          RDLIN                   ;WAIT FOR THE OPERATOR TO ENTER A 'CR'
3941  016550  012737  016550  001110  6$:     MOV     #6$,$LPERR      ;CHANGE ERROR LOOP ADDRESS
```

```
                                                JSR     PC,SAVRP        ;STORE THE REGISTERS
                                                BIT     #SUWP,$RPDS     ;IS THE DRIVE WRITE PROTECTED ?
                                                BNE     7$              ;BR IF IT IS
                                                ERROR   113             ;'SUWP' NOT SET WHEN DRIVE LOA SWITCHES
                                                                        ;'GREATER THAN SELECTED DRIVE NUMBER
                                        7$:     MOVB    CHKDRV,RPCS+1(R4)  ;SELECT THE NEXT DRIVE
                                                BR      2$              ;CONTINUE WITH THE TEST
                                        EXIT17: SCOPE                   ;LOOP ?
                                                TYPE    ,MSG17K         ;'END OF CONTROL PANEL TEST - RETURN SWITCHES TO NORMAL'
                                                JMP     @START2         ;RETURN TO THE 'CONVERSATION MODE' ENTRANCE

                                .SBTTL  END OF PASS ROUTINE

                                ;;**************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO RESTART

                                $EOP:
                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12><12>/END OF PASS/
                                64$:
                                                TST     @#DRVSEL        ;ANY DRIVES SELECTED
                                                BEQ     1$              ;NO, BRANCH
                                                TYPE    ,67$            ;;TYPE ASCIZ STRING
                                                BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  / USING DRIVE/
                                66$:
                                                MOV     @#CHKDRV,-(SP)  ;;SAVE @#CHKDRV FOR TYPEOUT
                                                TYPOS                   ;;GO TYPE--OCTAL ASCII .
                                                .BYTE   2               ;;TYPE 2 DIGIT(S)
                                                .BYTE   0               ;;SUPPRESS LEADING ZEROS
                                                TYPE    ,69$            ;;TYPE ASCIZ STRING
                                                BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  /   ERRORS DETECTED=/
                                68$:
                                                MOV     @#$ERTTL,-(SP)  ;;SAVE @#$ERTTL FOR TYPEOUT
                                                TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                                1$:             CLR     @#$ERTTL        ;ZERO ERROR TOTAL
                                                CLR     $TSTNM          ;;ZERO THE TEST NUMBER
                                                CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
                                                INC     $PASS           ;;INCREMENT THE PASS NUMBER
                                                BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
                                                DEC     (PC)+           ;;LOOP?
                                $EOPCT: .WORD   1
                                                BGT     $DOAGN          ;;YES
                                                MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
                                $ENDCT: .WORD   1
                                                $EOPCT
                                                TYPE    ,65$            ;;TYPE ASCIZ STRING
                                                BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <15><12>/END OF TEST/
                                64$:
                                                TYPE    ,$ENULL         ;TYPE NULL CHARACTER
```

```
 3998  017040  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 3999  017044  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 4000  017046  000005                          RESET                   ;;CLEAR THE WORLD
 4001  017050  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 4002  017052  000240                          NOP                     ;;SAVE ROOM
 4003  017054  000240                          NOP                     ;;FOR
 4004  017056  000240                          NOP                     ;;ACT11
 4005  017060                          $DOAGN:
 4006  017060  000137                          JMP     @(PC)+          ;;RETURN
 4007  017062  004762                  $RTNAD: .WORD   RESTART
 4008  017064     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 4009  017070                          .EVEN
 4010
 4011
 4012
 4013                                  .SBTTL  *** SUBROUTINES ***
 4014
 4015                                  .SBTTL  ERROR HANDLER ROUTINE
 4016
 4017                                  ;;**********************************************************************
 4018                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 4019                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 4020                                  ;*AND GO TO TYPERR ON ERROR
 4021                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 4022                                  ;*SW15=1        HALT ON ERROR
 4023                                  ;*SW13=1        INHIBIT ERROR TYPEOUTS
 4024                                  ;*SW10=1        BELL ON ERROR
 4025                                  ;*SW09=1        LOOP ON ERROR
 4026                                  ;*CALL
 4027                                  ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
 4028
 4029  017070                  $ERROR:
 4030  017070  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 4031  017072  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
 4032  017076  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
 4033  017100  013777  001102  162034          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
 4034  017106  032777  002000  162024          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
 4035  017114  001402                          BEQ     1$              ;;NO - SKIP
 4036  017116  104401  001166                  TYPE    ,$BELL          ;;RING BELL
 4037  017122  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
 4038  017126  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
 4039  017132  162737  000002  001116          SUB     #2,$ERRPC
 4040  017140  117737  161752  001114          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
 4041  017146  032777  020000  161764          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
 4042  017154  001004                          BNE     20$             ;;SKIP TYPEOUTS
 4043  017156  004737  017254                  JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
 4044  017162  104401  001173                  TYPE    ,$CRLF
 4045  017166                          20$:
 4046  017166  005777  161746          2$:     TST     @SWR            ;;HALT ON ERROR
 4047  017172  100002                          BPL     3$              ;;SKIP IF CONTINUE
 4048  017174  000000                          HALT                    ;;HALT ON ERROR!
 4049  017176  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 4050  017200  032777  001000  161732  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
 4051  017206  001402                          BEQ     4$              ;;BR IF NO
 4052  017210  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
 4053  017214  005737  001164          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
```

```
                        BEQ     5$                      ;;BR IF NONE
                        MOV     $ESCAPE,(SP)            ;;FUDGE RETURN ADDRESS FOR ESCAPE
                5$:
                        CMP     @#ENDAD,@#42            ;;ACT-11 AUTO-ACCEPT?
                        BNE     6$                      ;;BRANCH IF NO
                        HALT                            ;;YES
                6$:
                        CMP     @#42,@#46                       ;ACT11 AUTO MODE?
                        BNE     12$                             ;NO, RETURN
                        HALT                                    ;HALT ON ERROR
                12$:    RTI                                     ;RETURN

                ;;**************************************************************************

                ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
                ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
                ;TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
                ;CONCERNING THE ERROR.
                ;CALL
                ;       JSR     PC,TYPERR
                ;       RETURN

                TYPERR: SAVREG                          ;SAVE R0-R5
                        BIT     #SW13,@SWR              ;INHIBIT TYPEOUTS ?
                        BNE     20$                     ;BR IF YES
                        CLR     $TMP0                   ;CLEAR LOCATION FOR TEST NUMBER
                        MOVB    $TSTNM,$TMP0            ;LOAD TEST NUMBER FOR TYPEOUT
                        CLR     R0                      ;CLEAR R0 FOR ERROR NUMBER
                        MOVB    $ITEMB,R0               ;ERROR NUMBER
                        DEC     R0                      ;FORM INDEX FOR ERROR TABLE
                        ASL     R0
                        ASL     R0
                        ASL     R0
                1$:     ADD     #$ERRTB,R0              ;FORM ADDRESS
                        MOV     (R0)+,2$                ;GET ERROR MESSAGE (EM) POINTER
                        BEQ     3$                      ;BRANCH IF THERE ISN'T ONE
                        TYPE    ,$CRLF                  ;"CARRIAGE RETURN - LINE FEED
                        TYPE
                2$:     .WORD   0                       ;"EM" POINTER GOES HERE
                3$:     MOV     (R0)+,4$                ;PICK UP DATA HEADER (DH) POINTER
                        BEQ     5$                      ;BRANCH IF NONE
                        TYPE    ,$CRLF                  ;CARRIAGE RETURN-LINE FEED
                        TYPE
                4$:     .WORD   0                       ;"DH" POINTER GOES HERE
                5$:     MOV     (R0)+,R1                ;PICKUP DATA TABLE (DT) POINTER
                        BEQ     20$                     ;BRANCH IF NONE
                        CLR     R5                      ;SET INDENT SWITCH
                        MOV     (R0)+,R0                ;DATA FORMAT (DF) POINTER
                        MOV     (R0)+,R2                ;NUMBER OF DH'S TO TYPE
                        BEQ     20$                     ;BRANCH IF DH NUMBER IS 0
                        COM     R5                      ;NO INDENT
                10$:    MOVB    (R0)+,R3                ;NUMBER OF DATA WORDS TO TYPE
                        BEQ     20$                     ;BR IF NO WORDS TO TYPE
                        MOVB    (R0)+,R4                ;AND HOW TO TYPE THEM
                        TYPE    ,$CRLF                  ;CR-LF
                        TST     R5                      ;INDENT ?
```

```
                        001002                  BNE     11$             ;BR IF NOT
                        104401  027430          TYPE    $BLNKS2         ;BLANKS
                        006004          11$:    ROR     R4              ;OCTAL OR DECIMAL?
                        103403                  BCS     12$             ;DECIMAL--BRANCH
                        013146                  MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
                        104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
                        000402                  BR      13$
                                        12$:
                        013146                  MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
                        104405                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
                        005303          13$:    DEC     R3              ;MORE NUMBERS TO TYPE?
                        001403                  BEQ     14$             ;NO--BRANCH
                        104401  027430          TYPE    $BLNKS2         ;YES--TYPE SEPARATORS
                        000764                  BR      11$             ;LOOP
                        005302          14$:    DEC     R2              ;MORE DH'S?
                        003413                  BLE     20$             ;NO--BRANCH
                        104401  001173          TYPE    $CRLF           ;YES--START A NEW LINE
                        005105                  COM     R5              ;INDENT?
                        001002                  BNE     15$             ;NO--BRANCH
                        104401  027430          TYPE    $BLNKS2         ;YES--TYPE SPACES
                        012037  017474  15$:    MOV     @(R0)+,16$      ;GET NEXT DH
                        104401                  TYPE                    ;AND TYPE IT
                        000000          16$:    .WORD   0               ;DH POINTER GOES HERE
                        000736                  BR      10$             ;LOOP
                        104413          20$:    RESREG                  ;RESTORE R0-R5
                        000207                  RTS     PC              ;RETURN

                                        .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
                                        ;*

                        105737  001157  $TYPE:   TSTB    $TPFLG          ;;IS THERE A TERMINAL?
                        100002                  BPL     1$              ;;BR IF YES
                        000000                  HALT                    ;;HALT HERE IF NO TERMINAL
                        000407                  BR      3$              ;;LEAVE
                        010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
                        017600  000002          MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
                        112046          2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
                        001005                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
                        005726                  TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
                        012600          60$:    MOV     (SP)+,R0        ;;RESTORE R0
                        062716  000002  3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
                        000002                  RTI                     ;;RETURN
```

```
                                4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
                                        BEQ     8$
                                        CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
                                        BNE     5$
                                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
                                        TYPE                    ;;TYPE A CR AND LF
                                        $CRLF
                                        CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
                                        BR      2$              ;;GET NEXT CHARACTER
                                5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
                                6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
                                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
                                        MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
                                7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
                                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
                                        JSR     PC,$TYPEC       ;;GO TYPE A NULL
                                        DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
                                        BR      7$              ;;LOOP

                                ;HORIZONTAL TAB PROCESSOR

                                8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
                                9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
                                        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
                                        BNE     9$              ;;TAB STOP
                                        TST     (SP)+           ;;POP SPACE OFF STACK
                                        BR      2$              ;;GET NEXT CHARACTER
                                $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
                                        BPL     $TYPEC
                                        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
                                        CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
                                        BNE     1$              ;;BRANCH IF NO
                                        CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
                                        BR      $TYPEX          ;;EXIT
                                1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
                                        BEQ     $TYPEX          ;;BRANCH IF YES
                                        INCB    (PC)+           ;;COUNT THE CHARACTER
                                $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
                                $TYPEX: RTS     PC

                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                ;;**********************************************************************
                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                ;*      TYPOS                   ;;CALL FOR TYPEOUT
                                ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                ;*      .BYTE   M               ;;M=1 OR 0
                                ;*                                      ;;1=TYPE LEADING ZEROS
                                ;*                                      ;;0=SUPPRESS LEADING ZEROS
                                ;*
```

```
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPON                   ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                   ;;CALL FOR TYPEOUT
                                        
017724  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
017730  116637  000001  020147          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
017736  112637  020151                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
017742  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
017746  000406                          BR      $TYPON
017750  112737  000001  020147  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
017756  112737  000006  020151          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
017764  112737  000005  020146  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
017772  010346                          MOV     R3,-(SP)        ;;SAVE R3
017774  010446                          MOV     R4,-(SP)        ;;SAVE R4
017776  010546                          MOV     R5,-(SP)        ;;SAVE R5
020000  113704  020151                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
020004  005404                          NEG     R4
020006  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
020012  110437  020150                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
020016  113704  020147                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
020022  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
020026  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
020030  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
020032  000404                          BR      3$              ;;GO DO MSB
020034  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
020036  006105                          ROL     R5
020040  006105                          ROL     R5
020042  010503                          MOV     R5,R3
020044  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
020046  105337  020150                  DECB    $OMODE          ;;TYPE THIS DIGIT?
020052  100016                          BPL     7$              ;;BR IF NO
020054  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
020060  001002                          BNE     4$              ;;TEST FOR 0
020062  005704                          TST     R4              ;;SUPPRESS THIS 0?
020064  001403                          BEQ     5$              ;;BR IF YES
020066  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
020070  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
020074  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
020100  110337  020144                  MOVB    R3,@$           ;;SAVE FOR TYPING
020104  104401  020144                  TYPE    ,@$             ;;GO TYPE THIS DIGIT
020110  105337  020146          7$:     DECB    $OCNT           ;;COUNT BY 1
020114  003347                          BGT     2$              ;;BR IF MORE TO DO
020116  002402                          BLT     6$              ;;BR IF DONE
020120  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
020122  000744                          BR      2$              ;;GO DO THE LAST DIGIT
020124  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
020126  012604                          MOV     (SP)+,R4        ;;RESTORE R4
020130  012603                          MOV     (SP)+,R3        ;;RESTORE R3
020132  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
```

```
                        MOV     (SP)+,(SP)
                        RTI                             ;;RETURN
                B$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
                        .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
                $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
                $OFILL: .BYTE   0                       ;;ZERO FILL SWITCH
                $OMODE: .WORD   0                       ;;NUMBER OF DIGITS TO TYPE

                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                ;;*******************************************************************
                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                ;*REPLACED WITH SPACES.
                ;*CALL:
                ;*      MOV     NUM,-(SP)               ;;PUT THE BINARY NUMBER ON THE STACK
                ;*      TYPDS                           ;;GO TO THE ROUTINE

                $TYPDS:
                        MOV     R0,-(SP)                ;;PUSH R0 ON STACK
                        MOV     R1,-(SP)                ;;PUSH R1 ON STACK
                        MOV     R2,-(SP)                ;;PUSH R2 ON STACK
                        MOV     R3,-(SP)                ;;PUSH R3 ON STACK
                        MOV     R5,-(SP)                ;;PUSH R5 ON STACK
                        MOV     #20200,-(SP)            ;;SET BLANK SWITCH AND SIGN
                        MOV     20(SP),R5               ;;GET THE INPUT NUMBER
                        BPL     1$                      ;;BR IF INPUT IS POS.
                        NEG     R5                      ;;MAKE THE BINARY NUMBER POS.
                        MOVB    #'-,1(SP)               ;;MAKE THE ASCII NUMBER NEG.
                1$:     CLR     R0                      ;;ZERO THE CONSTANTS INDEX
                        MOV     #$DBLK,R3               ;;SETUP THE OUTPUT POINTER
                        MOVB    #' ,(R3)+               ;;SET THE FIRST CHARACTER TO A BLANK
                2$:     CLR     R2                      ;;CLEAR THE BCD NUMBER
                        MOV     $DTBL(R0),R1            ;;GET THE CONSTANT
                3$:     SUB     R1,R5                   ;;FORM THIS BCD DIGIT
                        BLT     4$                      ;;BR IF DONE
                        INC     R2                      ;;INCREASE THE BCD DIGIT BY 1
                        BR      3$
                4$:     ADD     R1,R5                   ;;ADD BACK THE CONSTANT
                        TST     R2                      ;;CHECK IF BCD DIGIT=0
                        BNE     5$                      ;;FALL THROUGH IF 0
                        TSTB    (SP)                    ;;STILL DOING LEADING 0'S?
                        BMI     7$                      ;;BR IF YES
                5$:     ASLB    (SP)                    ;;MSD?
                        BCC     6$                      ;;BR IF NO
                        MOVB    1(SP),-1(R3)            ;;YES--SET THE SIGN
                6$:     BIS     #'0,R2                  ;;MAKE THE BCD DIGIT ASCII
                7$:     BIS     #' ,R2                  ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
                        MOVB    R2,(R3)+                ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
                        TST     (R0)+                   ;;JUST INCREMENTING
                        CMP     R0,#10                  ;;CHECK THE TABLE INDEX
                        BLT     2$                      ;;GO DO THE NEXT DIGIT
                        BGT     8$                      ;;GO TO EXIT
                        MOV     R5,R2                   ;;GET THE LSD
```

```
4334   020310   000764                          BR      6$                      ;;GO CHANGE TO ASCII
4335   020312   105726                  8$:     TSTB    (SP)+                   ;;WAS THE LSD THE FIRST NON-ZERO?
4336   020314   100003                          BPL     9$                      ;;BR IF NO
4337   020316   116663   177777  177776         MOVB    -1(SP),-2(R3)           ;;YES--SET THE SIGN FOR TYPING
4338   020324   105013                  9$:     CLRB    (R3)                    ;;SET THE TERMINATOR
4339   020326   012605                          MOV     (SP)+,R5                ;;POP STACK INTO R5
4340   020330   012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
4341   020332   012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
4342   020334   012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
4343   020336   012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
4344   020340   104401   020366                 TYPE    $DBLK                   ;;NOW TYPE THE NUMBER
4345   020344   016666   000002  000004         MOV     2(SP),4(SP)             ;;ADJUST THE STACK
4346   020352   012616                          MOV     (SP)+,(SP)
4347   020354   000002                          RTI                             ;;RETURN TO USER
4348   020356   023420                  $DTBL:  10000.
4349   020360   001750                          1000.
4350   020362   000144                          100.
4351   020364   000012                          10.
4352   020366   000004                  $DBLK:  .BLKW   4
4353
4354                                    .SBTTL  TTY INPUT ROUTINE
4355
4356                                    ;;*****************************************************************
4357                                    .ENABL  LSB
4358   020376   000000                  $TKCNT: .WORD   0                       ;;NUMBER OF ITEMS IN QUEUE
4359   020400   000000                  $TKQIN: .WORD   0                       ;;INPUT POINTER
4360   020402   000000                  $TKQOUT: .WORD  0                       ;;OUTPUT POINTER
4361   020404   000012                  $TKQSRT: .BLKB  10.                     ;;TTY KEYBOARD QUEUE
4362            020416                  $TKQEND=.
4363
4364                                    ;*TK INITIALIZE ROUTINE
4365                                    ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
4366                                    ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
4367                                    ;
4368                                    ;*CALL:
4369                                    ;*      JSR     PC,$TKINT
4370                                    ;*      RETURN
4371                                    ;
4372   020416   005037   020376          $TKINT: CLR     $TKCNT                  ;;CLEAR COUNT OF ITEMS IN QUEUE
4373   020422   012737   020404  020400         MOV     #$TKQSRT,$TKQIN         ;;MOVE THE STARTING ADDRESS OF THE
4374   020430   013737   020400  020402         MOV     $TKQIN,$TKQOUT          ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
4375   020436   012737   020466  000060         MOV     #$TKSRV,@#TKVEC         ;;INITIALIZE THE KEYBOARD VECTOR
4376   020444   012737   000200  000062         MOV     #200,@#TKVEC+2          ;;"BR" LEVEL 4
4377   020452   005777   160470                 TST     @$TKB                   ;;CLEAR DONE FLAG
4378   020456   012777   000100  160460         MOV     #100,@$TKS              ;;ENABLE TTY KEYBOARD INTERRUPT
4379   020464   000207                          RTS     PC                      ;;RETURN TO CALLER
4380
4381                                    ;*TK SERVICE ROUTINE
4382                                    ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
4383                                    ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
4384                                    ;*IT IN THE QUEUE.
4385                                    ;*IF THE CHARACTER IS A "CONTROL-C" (↑C) $TKINT IS CALLED AND
4386                                    ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START2)
4387                                    ;
4388   020466   117746   160454          $TKSRV: MOVB    @$TKB,-(SP)             ;;PICKUP THE CHARACTER
4389   020472   042716   177600                 BIC     #↑C177,(SP)             ;;STRIP THE JUNK
```

```
 4390  020476  021627  000003            CMP     (SP),#3         ;;IS IT A CONTROL C?
 4391  020502  001007                    BNE     1$              ;;BRANCH IF NO
 4392  020504  104401  021604            TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
 4393  020510  004737  020416            JSR     PC,$TKINT       ;;INIT THE KEYBOARD
 4394  020514  005726                    TST     (SP)+           ;;CLEAN UP STACK
 4395  020516  000137  003676            JMP     START2          ;;CONTROL C RESTART
 4396  020522  021627  000007    1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
 4397  020526  001004                    BNE     2$              ;;BRANCH IF NO
 4398  020530  022737  000176  001140    CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
 4399  020536  001500                    BEQ     6$              ;;GO TO SWR CHANGE
 4400
 4401  020540                    2$:
 4402  020540  022737  000012  020376    CMP     #10.,$TKCNT     ;;IS THE QUEUE FULL?
 4403  020546  001004                    BNE     3$              ;;BRANCH IF NO
 4404  020550  104401  001166            TYPE    ,$BELL          ;;RING THE TTY BELL
 4405  020554  005726                    TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
 4406  020556  000451                    BR      5$              ;;EXIT
 4407  020560  021627  000023    3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
 4408  020564  001021                    BNE     32$             ;;BRANCH IF NO
 4409  020566  005077  160352            CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 4410  020572  005726                    TST     (SP)+           ;;CLEAN CHAR OFF STACK
 4411  020574  105777  160344    31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
 4412  020600  100375                    BPL     31$             ;;LOOP UNTIL ITS THERE
 4413  020602  117746  160340            MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
 4414  020606  042716  177600            BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
 4415  020612  022627  000021            CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 4416  020616  001366                    BNE     31$             ;;BRANCH IF NO
 4417  020620  012777  000100  160316    MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
 4418  020626  000002                    RTI                     ;;RETURN
 4419  020630  005237  020376    32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
 4420  020634  021627  000140            CMP     (SP),#140       ;;IS IT UPPER CASE?
 4421  020640  002405                    BLT     4$              ;;BRANCH IF YES
 4422  020642  021627  000175            CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
 4423  020646  003002                    BGT     4$              ;;BRANCH IF YES
 4424  020650  042716  000040            BIC     #40,(SP)        ;;MAKE IT UPPER CASE
 4425  020654  112677  177520    4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
 4426  020660  005237  020400            INC     $TKQIN          ;;UPDATE THE POINTER
 4427  020664  023727  020400  020416    CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
 4428  020672  001003                    BNE     5$              ;;BRANCH IF NO
 4429  020674  012737  020404  020400    MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
 4430  020702  000002            5$:     RTI                     ;;RETURN
 4431
 4432                                    ;;**********************************************************************
 4433                                    ;;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 4434                                    ;;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
 4435                                    ;;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
 4436                                    ;;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
 4437  020704  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
 4438  020712  001124                    BNE     15$             ;;EXIT IF NOT
 4439  020714  105777  160224            TSTB    @$TKS           ;;IS A CHAR WAITING?
 4440  020720  100121                    BPL     15$             ;;IF NOT, EXIT
 4441  020722  117746  160220            MOVB    @$TKB,-(SP)     ;;YES
 4442  020726  042716  177600            BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
 4443  020732  021627  000007            CMP     (SP),#7         ;;IS IT A CONTROL-G?
 4444  020736  001300                    BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
 4445                                                            ;;AND EXIT
```

```
                                        ;;*********************************************************************
                                        ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                        ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                        ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
020740  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
020746  001674                        BEQ     2$              ;;BRANCH IF YES
020750  005726                        TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
020752  004737  020416                JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
020756  005077  160162                CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
020762  112737  000001  001135        MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

020770  104401  021616                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G (↑G)
020774  104401  021623        $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
021000  013746  000176                MOV     $SWREG,-(SP)    ;;SAVE SWREG FOR TYPEOUT
021004  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
021006  104401  021634                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
021012  005046                19$:    CLR     -(SP)           ;;CLEAR COUNTER
021014  005046                        CLR     -(SP)           ;;THE NEW SWR
021016  105777  160122        7$:     TSTB    @$TKS           ;;CHAR THERE?
021022  100375                        BPL     7$              ;;IF NOT TRY AGAIN

021024  117746  160116                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
021030  042716  177600                BIC     #↑C177,(SP)     ;;MAKE IT 7-BIT ASCII

021034  021627  000003                CMP     (SP),#3         ;;IS IT A CONTROL-C?
021040  001015                        BNE     9$              ;;BRANCH IF NOT
021042  104401  021604                TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (↑C)
021046  062706  000006                ADD     #6,SP           ;;CLEAN UP STACK
021052  123727  001135  000001        CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
021060  001003                        BNE     8$              ;;BRANCH IF NO
021062  012777  000100  160054        MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
021070  000137  003676        8$:     JMP     START2          ;;CONTROL-C RESTART


021074  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
021100  001005                        BNE     10$             ;;BRANCH IF NOT
021102  104401  021611                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (↑U)
021106  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
021112  000737                        BR      19$             ;;LET'S TRY IT AGAIN


021114  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
021120  001022                        BNE     16$             ;;BRANCH IF NO
021122  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
021126  001403                        BEQ     11$             ;;BRANCH IF YES
021130  016677  000002  160002        MOV     2(SP),@SWR      ;;SAVE NEW SWR
021136  062706  000006        11$:    ADD     #6,SP           ;;CLEAR UP STACK
021142  104401  001173        14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
021146  123727  001135  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
021154  001003                        BNE     15$             ;;BRANCH IF NOT
021156  012777  000100  157760        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
021164  000002                15$:    RTI                     ;;RETURN
021166  004737  017654        16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
021172  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
021176  002420                        BLT     18$             ;;BRANCH IF YES
```

```
                                CMP     (SP),#67        ;;CHAR > 7?
                                BGT     18$             ;;BRANCH IF YES
                                BIC     #60,(SP)+       ;;STRIP-OFF ASCII
                                TST     2(SP)           ;;IS THIS THE FIRST CHAR
                                BEQ     17$             ;;BRANCH IF YES
                                ASL     (SP)            ;;NO, SHIFT PRESENT
                                ASL     (SP)            ;;    CHAR OVER TO MAKE
                                ASL     (SP)            ;;    ROOM FOR NEW ONE.
                        17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
                                BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
                                BR      7$              ;;GET THE NEXT ONE
                        18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
                                BR      20$             ;;SIMULATE CONTROL-U
                                .DSABL  LSB

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

                        $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
                                MOV     4(SP),2(SP)     ;;THE PS
                                CLR     4(SP)           ;;GET READY FOR A CHARACTER
                                CLR     -(SP)           ;;PUT NEW PS ON STACK
                                MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
                                RTI                     ;;POP NEW PC AND PS
                        64$:
                        1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
                                BEQ     1$
                                DEC     $TKCNT          ;;DECREMENT THE COUNTER
                                MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
                                INC     $TKQOUT         ;;UPDATE THE POINTER
                                CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
                                BNE     2$              ;;BRANCH IF NO
                                MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
                        2$:     RTI                     ;;RETURN
                        ;;**********************************************************************
                        ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                        ;*CALL:
                        ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                        ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                        ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

                        $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
                                CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
                        1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
                        2$:     CMP     #$TTYIN+10.,R3  ;;BUFFER FULL?
                                BLOS    4$              ;;BR IF YES
                                RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
                                MOVB    (SP)+,(R3)      ;;GET CHARACTER
                        10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
                                BNE     5$              ;;BR IF NO
```

```
4558  021366  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
4559  021370  001007                          BNE     6$              ;;BR IF NO
4560  021372  112737  000134  021570         MOVB    #'\,9$          ;;TYPE A BACK SLASH
4561  021400  104401  021570                  TYPE    9$
4562  021404  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
4563  021410  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
4564  021412  020327  021572                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
4565  021416  103434                          BLO     4$              ;;BR IF YES
4566  021420  111337  021570                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
4567  021424  104401  021570                  TYPE    9$              ;;GO TYPE
4568  021430  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
4569  021432  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
4570  021434  001406                          BEQ     7$              ;;BR IF NO
4571  021436  112737  000134  021570         MOVB    #'\,9$          ;;TYPE A BACK SLASH
4572  021444  104401  021570                  TYPE    9$
4573  021450  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
4574  021452  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
4575  021456  001003                          BNE     8$              ;;BR IF NO
4576  021460  104401  021611                  TYPE    $CNTLU          ;;TYPE A CONTROL "U"
4577  021464  000726                          BR      1$              ;;GO START OVER
4578  021466  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "↑R"?
4579  021472  001011                          BNE     3$              ;;BRANCH IF NO
4580  021474  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
4581  021476  104401  001173                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
4582  021502  104401  021572                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
4583  021506  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
4584  021510  104401  001172          4$:     TYPE    $QUES           ;;TYPE A '?'
4585  021514  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
4586  021516  111337  021570          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
4587  021522  104401  021570                  TYPE    9$
4588  021526  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
4589  021532  001305                          BNE     2$              ;;LOOP IF NOT RETURN
4590  021534  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
4591  021540  104401  001174                  TYPE    $LF             ;;TYPE A LINE FEED
4592  021544  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
4593  021546  012603                          MOV     (SP)+,R3        ;;RESTORE R3
4594  021550  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
4595  021552  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
4596  021560  012766  021572  000004          MOV     #$TTYIN,4(SP)
4597  021566  000002                          RTI                     ;;RETURN
4598  021570     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
4599  021571     000                          .BYTE   0               ;;TERMINATOR
4600  021572  000012          $TTYIN: .BLKB   10.             ;;RESERVE 10. BYTES FOR TTY INPUT
4601  021604  041536  005015     000  $CNTLC: .ASCIZ  /↑C/<15><12>    ;;CONTROL "C"
4602  021611     136  006525  000012  $CNTLU: .ASCIZ  /↑U/<15><12>    ;;CONTROL "U"
4603  021616  043536  005015     000  $CNTLG: .ASCIZ  /↑G/<15><12>    ;;CONTROL "G"
4604  021623     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
4605  021630  036440  000040
4606  021634  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
4607  021642  020075     000
4608          021646          .EVEN
4609
4610                          .SBTTL  SCOPE HANDLER ROUTINE
4611
4612                          ;;**********************************************************************
4613                          ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
```

```
                                        ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                        ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW14=1          LOOP ON TEST
                                        ;*SW11=1          INHIBIT ITERATIONS
                                        ;*SW09=1          LOOP ON ERROR
                                        ;*CALL
                                        ;*      SCOPE             ;;SCOPE=IOT

021646                                  $SCOPE:
021646  104407                                 CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
021650  032777  040000  157262          1$:    BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
021656  001101                                 BNE     $OVER           ;;YES IF SW14=1
                                        .#####START OF CODE FOR THE XOR TESTER#####
021660  000416                          $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
021662  013746  000004                          MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
021666  012737  021706  000004                 MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
021674  005737  177060                         TST     @#177060        ;;TIME OUT ON XOR?
021700  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
021704  000453                                 BR      $SVLAD          ;;GO TO THE NEXT TEST
021706  022626                          5$:    CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
021710  012637  000004                         MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
021714  000413                                 BR      7$              ;;LOOP ON THE PRESENT TEST
021716                                  6$:;#####END OF CODE FOR THE XOR TESTER#####
021716  105737  001103                  2$:    TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
021722  001421                                 BEQ     3$              ;;BR IF NO
021724  123737  001115  001103                CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
021732  101015                                 BHI     3$              ;;BR IF NO
021734  032777  001000  157176                BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
021742  001404                                 BEQ     4$              ;;BR IF NO
021744  013737  001110  001106          7$:    MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
021752  000443                                 BR      $OVER
021754  105037  001103                  4$:    CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
021760  005037  001162                         CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
021764  000415                                 BR      1$              ;;ESCAPE TO THE NEXT TEST
021766  032777  004000  157144          3$:    BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
021774  001011                                 BNE     1$              ;;BR IF YES
021776  005737  001100                         TST     $PASS           ;;IF FIRST PASS OF PROGRAM
022002  001406                                 BEQ     1$              ;;      INHIBIT ITERATIONS
022004  005237  001104                         INC     $ICNT           ;;INCREMENT ITERATION COUNT
022010  023737  001162  001104                CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
022016  002021                                 BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
022020  012737  000001  001104          1$:    MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
022026  013737  022076  001162                MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
022034  105237  001102                  $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
022040  011637  001106                         MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
022044  011637  001110                         MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
022050  005037  001164                         CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
022054  112737  000001  001115                MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
022062  013777  001102  157052          $OVER: MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
022070  013716  001106                         MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
022074  000002                                 RTI                     ;;FIXES PS
022076  000001                          $MXCNT: 1                       ;;MAX. NUMBER OF ITERATIONS

                                        .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
```

```
                                ;;*****************************************************************
                                ;*SAVE R0-R5
                                ;*CALL:
                                ;*      SAVREG
                                ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                ;*
                                ;*TOP---(+16)
                                ;* +2---(+18)
                                ;* +4---R5
                                ;* +6---R4
                                ;* +8---R3
                                ;*+10---R2
                                ;*+12---R1
                                ;*+14---R0

022100                          $SAVREG:
022100  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
022102  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
022104  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
022106  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
022110  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
022112  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
022114  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
022120  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
022124  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
022130  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
022134  000002                          RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*      RESREG
022136                          $RESREG:
022136  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
022142  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
022146  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
022152  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
022156  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
022160  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
022162  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
022164  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
022166  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
022170  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
022172  000002                          RTI

                                .SBTTL  ROUTINE TO SIZE MEMORY

                                ;;*****************************************************************
                                ;*CALL:
                                ;*      JSR     PC,$SIZE
                                ;*      RETURN
                                ;*$LSTAD WILL CONTAIN:
                                ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
                                ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
                                ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
                                ;*$KT11 IS THE MEMORY MANAGEMENT KEY
```

```
                                        ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
                                        ;*      MUST BE SETUP BEFORE THE CALL
                                        ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
                                        ;*      DETERMINED BY ROUTINE

022174  010046                $SIZE:  MOV     R0,-(SP)        ;;SAVE R0 ON THE STACK
022176  010146                        MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
022200  010246                        MOV     R2,-(SP)        ;;SAVE R2 ON THE STACK
022202  010346                        MOV     R3,-(SP)        ;;SAVE R3 ON THE STACK
022204  013746  000004                MOV     @#ERRVEC,-(SP)  ;;SAVE PRESENT ERROR VECTOR PS & PC
022210  013746  000006                MOV     @#ERRVEC+2,-(SP)
022214  010600                        MOV     SP,R0           ;;SAVE THE STACK POINTER
                                ;;SET THE ERRVEC PS TO THE PRESENT PS
022216  104400                        TRAP                    ;;PUSH OLD PSW AND PC ON STACK
022220  012637  000006                MOV     (SP)+,@#ERRVEC+2        ;;SAVE THE PSW IN @#ERRVEC+2
022224  012701  003776                MOV     #3776,R1        ;;SETUP ADDRESS
022230  105727                        TSTB    (PC)+           ;;USE MEMORY MANAGEMENT?
022232  000200                $KT11:  .WORD   200             ;;SET TO USE MEMORY MANAGEMENT
022234  100062                        BPL     $CORE           ;;BR IF NO
022236  012737  022374  000004        MOV     #$KTNEX,@#ERRVEC        ;;SET FOR TIMEOUT
022244  005737  177572                TST     @#SR0           ;;KT11 ARE YOU THERE?
022250  052737  100000  022232        BIS     #100000,$KT11   ;;YES--SET KT11 KEY
022256  005046                        CLR     -(SP)           ;;INITIALIZE FOR "PAR" LOADING
022260  012702  172340                MOV     #KIPAR0,R2      ;;ADDRESS OF FIRST "PAR"
022264  012703  000010                MOV     #^D8,R3         ;;LOAD EIGHT "PAR.'S" AND EIGHT "PDR.'S"
022270  012762  077406  177740  1$:   MOV     #77406,-40(R2)  ;;PDR = 4K, UP, READ/WRITE
022276  011622                        MOV     (SP),(R2)+      ;;LOAD "PAR"
022300  062716  000200                ADD     #200,(SP)       ;;UPDATE FOR NEXT "PAR"
022304  077307                        SOB     R3,1$           ;;LOOP UNTIL ALL EIGHT ARE LOADED
022306  012742  177600                MOV     #177600,-(R2)   ;;SETUP KIPAR7 FOR I/O
022312  005042                        CLR     -(R2)           ;;SETUP KIPAR6 FOR TESTING
022314  012737  022332  000004        MOV     #2$,@#ERRVEC    ;;CATCH TIMEOUT IF NO SR3
022322  012737  000020  172516        MOV     #20,@#SR3       ;;ENABLE 22 BIT MODE
022330  000401                        BR      3$              ;;THIS PDP-11 HAS A SR3 REGISTER
022332  022626                2$:     CMP     (SP)+,(SP)+     ;;CLEAN OFF THE STACK--NO SR3
022334  005237  177572        3$:     INC     @#SR0           ;;TURN ON MEMORY MANAGEMENT
022340  012737  022364  000004        MOV     #$KTOUT,@#ERRVEC        ;;SET FOR TIME OUT
022346  005737  143776        4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
022352  062712  000040                ADD     #40,(R2)        ;;MAKE A 1K STEP
022356  023712  172356                CMP     @#KIPAR7,(R2)   ;;LAST ONE?
022362  101371                        BHI     4$              ;;NO--TRY IT
022364  011202                $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
022366  005037  177572                CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
022372  000421                        BR      $SIZEX
022374  042737  100000  022232  $KTNEX: BIC   #100000,$KT11   ;;KT11 NON-EXISTENT
022402  012737  022432  000004  $CORE: MOV     #$CROUT,@#ERRVEC        ;;SET FOR TIMEOUT
022410  005002                        CLR     R2              ;;SET UP BANK
022412  062701  004000        1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
022416  062702  000040                ADD     #40,R2          ;;1K STEP
022422  005711                        TST     (R1)            ;;TRAP ON TIME OUT
022424  022701  177776                CMP     #177776,R1      ;;LAST ONE
022430  001370                        BNE     1$              ;;NO--TRY AGAIN
022432  162701  004000        $CROUT: SUB     #4000,R1
022436  162702  000040        $SIZEX: SUB     #40,R2          ;;DROP BACK
022442  010006                        MOV     R0,SP           ;;RESTORE THE STACK
022444  012637  000006                MOV     (SP)+,@#ERRVEC+2        ;;RESTORE ERROR VECTOR
```

```
                                        MOV     (SP)+,@#ERRVEC
                                        MOV     R1,$LSTAD       ;;LAST ADDRESS
                                        MOV     R2,$LSTBK       ;;LAST BANK
                                        MOV     (SP)+,R3        ;;RESTORE R3
                                        MOV     (SP)+,R2        ;;RESTORE R2
                                        MOV     (SP)+,R1        ;;RESTORE R1
                                        MOV     (SP)+,R0        ;;RESTORE R0
                                        RTS     PC
                                $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
                                $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK

                                .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

                                ;;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)
                                        $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)

                                        $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING

                                        $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
                                        $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
                                        $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
                                        $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
                                        $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
```

```
                                        ;THIS ROUTINE CLEARS THE RP11 AND DETERMINES WHICH DRIVES ARE AVAILABLE.
                                        ;THE TABLE 'DRVSTA' IS LOADED TO REFLECT THE SYSTEM STATUS.

022566  104412                  RPINIT: SAVREG                          ;SAVE R0-R5
022570  013701  001326                  MOV     RPVEC,R1                ;VECTOR ADDRESS
022574  005021                          CLR     (R1)+                   ;SET INTERRUPT ADDRESS TO ZERO
022576  013711  001332                  MOV     RPPRIO,(R1)             ;RP11 PRIORITY
022602  005037  001304                  CLR     DRVTYP                  ;CLEAR DRIVE TYPE STORAGE
022606  005037  001370                  CLR     DRVSTA                  ;SET DRIVE STATUS TO OFFLINE
022612  005037  001372                  CLR     DRVSTA+2                ;SET DRIVE STATUS TO OFFLINE
022616  005037  001374                  CLR     DRVSTA+4                ;SET DRIVE STATUS TO OFFLINE
022622  005037  001376                  CLR     DRVSTA+6                ;SET DRIVE STATUS TO OFFLINE
022626  013704  001324                  MOV     RPADR,R4                ;PUT RP11 ADDRESS INTO R4
022632  012764  000001  000004          MOV     #CLEAR,RPCS(R4)         ;CLEAR THE RP11
022640  012764  000001  000004          MOV     #CLEAR,RPCS(R4)         ;CLEAR THE RP11 AGAIN
022646  012701  000007                  MOV     #7,R1                   ;'DRVSTA' TABLE INDEX
022652  004737  022666          1$:     JSR     PC,DRVINT               ;CHECK THE DRIVE'S STATUS
022656  005301                          DEC     R1                      ;DECREMENT THE TABLE INDEX
022660  100374                          BPL     1$                      ;BR IF NOT FINISHED
022662  104413                          RESREG                          ;RESTORE R0-R5
022664  000207                          RTS     PC                      ;RETURN

                                        ;ROUTINE TO CHECK THE DRIVE'S STATUS.  DRIVE NUMBER MUST BE IN R1

022666  110164  000005          DRVINT: MOVB    R1,RPCS+1(R4)   ;SELECT DRIVE
022672  032764  001000  000000          BIT     #SUFU,RPDS(R4)  ;SEE IF DRIVE UNSAFE
022700  001404                          BEQ     1$              ;BR IF NOT UNSAFE
022702  112761  177777  001370          MOVB    #-1,DRVSTA(R1)  ;SET INDICATOR TO 'UNSAFE'
022710  000417                          BR      3$              ;EXIT
022712  032764  040000  000000  1$:     BIT     #SUOL,RPDS(R4)  ;IS DRIVE ONLINE ?
022720  001422                          BEQ     4$              ;BR IF NOT
022722  032764  004000  000000          BIT     #SUSI,RPDS(R4)  ;SEEK INCOMPLETE SET ?
022730  001004                          BNE     2$              ;BR IF SET
022732  032764  100000  000000          BIT     #SURDY,RPDS(R4) ;IS DRIVE READY ?
022740  001412                          BEQ     4$              ;BR IF NOT
022742  112761  000001  001370  2$:     MOVB    #1,DRVSTA(R1)   ;SET DRIVE INDICATOR TO 'ONLINE'
022750  032764  020000  000000  3$:     BIT     #SURP03,RPDS(R4) ;IS THE DRIVE AN RP03 ?
022756  001403                          BEQ     4$              ;BR IF NOT
022760  156137  001360  001304          BISB    ATABIT(R1),DRVTYP ;SET RP03 INDICATOR
022766  000207                  4$:     RTS     PC              ;RETURN

                                        ;DO THE REQUESTED COMMAND.  THE RP11 ADDRESS MUST BE IN R4

022770  104412                  RP11:   SAVREG                  ;SAVE R0 - R5
022772  113764  002500  000005          MOVB    DPB,RPCS+1(R4)  ;SELECT THE DRIVE
023000  005001                          CLR     R1              ;CLEAR R1
023002  113701  002500                  MOVB    DPB,R1          ;DRIVE ADDRESS
023006  004737  023120                  JSR     PC,CIR          ;SEE IF DRIVE IS READY
023012  013746  002504                  MOV     $WC,-(SP)       ;COMPLEMENT THE WORD COUNT
023016  005416                          NEG     (SP)            ;COMPLEMENT
023020  012664  000006                  MOV     (SP)+,RPWC(R4)  ;LOAD THE WORD COUNT REGISTER
023024  013764  002506  000010          MOV     $BUF,RPBA(R4)   ;BUFFER ADDRESS
023032  013764  002510  000012          MOV     $CYL,RPCA(R4)   ;CYLINDER ADDRESS
023040  013764  002512  000014          MOV     $SEC,RPDA(R4)   ;SECTOR/TRACK ADDRESS
023046  012737  024000  023060          MOV     #24000,2$               ;SET UP FOR....
```

```
 4894  023054  005337  023060          1$:     DEC     2$                      ;...A DELAY OF
 4895  023060  000000                  2$:     0                               ;...AT LEAST
 4896  023062  001374                          BNE     1$                      ;...2S MILLISECONDS.
 4897  023064  113764  002500  000005          MOVB    DPB,RPCS+1(R4)  ;SELECT THE DRIVE
 4898  023072  153764  002503  000005          BISB    DPB+3,RPCS+1(R4)  ;SET 'MODE' & 'HDR' BITS
 4899  023100  013746  002502                  MOV     DPB+2,-(SP)     ;COMMAND CODE
 4900  023104  153716  002501                  BISB    DPB+1,(SP)      ;'MEX' BITS
 4901  023110  112664  000004                  MOVB    (SP)+,RPCS(R4)  ;START THE COMMAND
 4902  023114  104413                          RESREG                  ;RESTORE R0 - R5
 4903  023116  000207                          RTS     PC              ;RETURN
 4904
 4905                                  ;SEE IF DRIVE IS STILL ONLINE
 4906
 4907  023120                          CIR:
 4908  023120  012737  023152  001164          MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
 4909  023126  004737  022666                  JSR     PC,DRVINT       ;CHECK ON THE DRIVE'S STATUS
 4910  023132  004737  023240                  JSR     PC,SAVRP        ;STORE RP11 REGISTERS IN CASE OF ERROR
 4911  023136  105761  001370                  TSTB    DRVSTA(R1)      ;IS THE DRIVE STILL ONLINE ?
 4912  023142  003014                          BGT     3$              ;BR IF IT IS
 4913  023144  002401                          BLT     1$              ;BR IF DRIVE IS UNSAFE
 4914  023146  104076                          ERROR   76              ;DRIVE HAS GONE OFFLINE
 4915  023150  104077                  1$:     ERROR   77              ;DRIVE HAS BECOME UNSAFE
 4916  023152  032777  001000  155760  2$:     BIT     #SWD9,@SWR      ;LOOP ON THE ERROR ?
 4917  023160  001357                          BNE     CIR             ;BR IF LOOP
 4918  023162  156137  001360  001310          BISB    ATABIT(R1),DRVBAD  ;SETUP TO DEASSIGN THE DRIVE
 4919  023170  000137  016616                  JMP     @#$EOP          ;GO TO THE END OF PASS ROUTINE
 4920  023174  005037  001164          3$:     CLR     $ESCAPE         ;CLEAR THE 'ESCAPE' FLAG
 4921  023200  000207                          RTS     PC              ;RETURN
 4922
 4923                                  ;THIS ROUTINE WILL PROVIDE A STALL IN MILLISECONDS FOR A SPECIFIC
 4924                                  ;AMOUNT OF TIME.  STALL VALUE IS STORED IN 'STALLT'.
 4925                                  ;       JSR     PC,STALL
 4926
 4927  023202  013746  001220          STALL:  MOV     STALLT,-(SP)    ;PICKUP THE STALL TIME
 4928  023206  005046                          CLR     -(SP)           ;CLEAR TEMP. LOCATION
 4929  023210  162766  000001  000002  1$:     SUB     #1,2(SP)        ;MORE STALL REQUIRED?
 4930  023216  103406                          BLO     3$              ;NO--BRANCH
 4931  023220  012716  000454                  MOV     #300.,(SP)      ;STALL FOR ABOUT 1 MILLISECOND
 4932  023224  005700                  2$:     TST     R0              ;NOP TO KILL TIME
 4933  023226  005316                          DEC     (SP)            ;COUNT
 4934  023230  001375                          BNE     2$              ;LOOP IF MORE COUNTS NEEDED
 4935  023232  000766                          BR      1$
 4936  023234  022626                  3$:     CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
 4937  023236  000207                          RTS     PC              ;EXIT
 4938
 4939                                  ;THIS ROUTINE STORES THE RP11E REGISTERS FOR USE BY THE PROGRAM.
 4940                                  ;THE PROGRAM DOES NOT USE THE 'LIVE' REGISTERS FOR ERROR DETERMINATION.
 4941                                  ;CALL
 4942                                  ;       JSR     PC,SAVRP
 4943                                  ;       RETURN
 4944
 4945  023240  010446                  SAVRP:  MOV     R4,-(SP)        ;SAVE R4
 4946  023242  013704  001324                  MOV     RPADR,R4        ;RP11 ADDRESS
 4947  023246  016437  000000  001334          MOV     RPDS(R4),$RPDS  ;STORE RPDS
 4948  023254  016437  000002  001336          MOV     RPER(R4),$RPER  ;STORE RPER
 4949  023262  016437  000004  001340          MOV     RPCS(R4),$RPCS  ;STORE RPCS
```

```
 4950  023270  016437  000006  001342          MOV     RPWC(R4),$RPWC  ;STORE RPWC
 4951  023276  016437  000010  001344          MOV     RPBA(R4),$RPBA  ;STORE RPBA
 4952  023304  016437  000012  001346          MOV     RPCA(R4),$RPCA  ;STORE RPCA
 4953  023312  016437  000014  001350          MOV     RPDA(R4),$RPDA  ;STORE RPDA
 4954  023320  016437  000016  001352          MOV     RPM1(R4),$RPM1  ;STORE RPM1
 4955  023326  016437  000024  001354          MOV     SUCA(R4),$SUCA  ;STORE SUCA
 4956  023334  016437  000026  001356          MOV     SILO(R4),$SILO  ;STORE SILO
 4957  023342  012604                          MOV     (SP)+,R4        ;RESTORE R4
 4958  023344  000207                          RTS     PC              ;RETURN
 4959
 4960                                  ;ROUTINE TO MOVE THE LOADER FROM HIGH MEMORY TO JUST ABOVE THE PROGRAM
 4961
 4962  023346  105737  000041        MOVLDR: TSTB    41              ;WHO LOADED THE PROGRAM
 4963  023352  001005                          BNE     1$              ;BR IF NOT 'ABS'
 4964  023354  012702  000144                  MOV     #100.,R2        ;'ABS' LOADER SIZE
 4965  023360  012701  044166                  MOV     #BUFFER+200.,R1 ;RELOCATE TO HERE
 4966  023364  000404                          BR      2$              ;FINISH
 4967  023366  012702  002734          1$:     MOV     #1500.,R2       ;'XXDP' SIZE
 4968  023372  012701  051546                  MOV     #BUFFER+3000.,R1  ;RELOCATE TO HERE (UPPER BOUNDARY)
 4969  023376  012703  157776          2$:     MOV     #157776,R3      ;TRY THIS UPPER MEMORY ADDRESS
 4970  023402  005737  022232                  TST     $KT11           ;MEMORY MANAGEMENT ?
 4971  023406  100402                          BMI     3$              ;BR IF YES
 4972  023410  013703  022476                  MOV     $LSTAD,R3       ;USE ACTUAL HIGH ADDRESS
 4973  023414  005046                  3$:     CLR     -(SP)           ;CLEAR THE STACK
 4974  023416  016616  000002                  MOV     2(SP),(SP)      ;MOVE THE RETURN ADDRESS DOWN THE STACK
 4975  023422  010166  000002                  MOV     R1,2(SP)        ;BEGINNING ADDRESS OF BUFFER
 4976  023426  005737  001322                  TST     LDRFLG          ;LOADER ALREADY RELOCATED ?
 4977  023432  001006                          BNE     5$              ;BR IF YES
 4978  023434  014341                  4$:     MOV     -(R3),-(R1)     ;MOVE A WORD
 4979  023436  005302                          DEC     R2              ;COUNT IT
 4980  023440  001375                          BNE     4$              ;BR IF MORE TO GO
 4981  023442  012737  177777  001322          MOV     #-1,LDRFLG      ;SET THE LOADER MOVED FLAG
 4982  023450  000207                  5$:     RTS     PC              ;RETURN
 4983
 4984                                  ;ROUTINE TO RELOCATE THE LOADER TO UPPER MEMORY
 4985
 4986  023452  104412                  RESLDR: SAVREG                  ;SAVE R0 - R5
 4987  023454  005737  001322                  TST     LDRFLG          ;LOADER MOVED ?
 4988  023460  001430                          BEQ     5$              ;BR IF NOT
 4989  023462  105737  000041                  TSTB    41              ;'ABS' LOADER
 4990  023466  001005                          BNE     1$              ;BR IF NOT
 4991  023470  012702  000144                  MOV     #100.,R2        ;'ABS' LOADER SIZE
 4992  023474  012701  044166                  MOV     #BUFFER+200.,R1 ;RELOCATE FROM HERE
 4993  023500  000404                          BR      3$              ;FINISH
 4994  023502  012702  002734          1$:     MOV     #1500.,R2       ;'XXDP' SIZE
 4995  023506  012701  051546                  MOV     #BUFFER+3000.,R1  ;RELOCATE FROM HERE (UPPER BOUNDARY)
 4996  023512  012703  157776          3$:     MOV     #157776,R3      ;TRY THIS UPPER MEMORY ADDRESS
 4997  023516  005737  022232                  TST     $KT11           ;MEMORY MANAGEMENT ?
 4998  023522  100402                          BMI     4$              ;BR IF YES
 4999  023524  013703  022476                  MOV     $LSTAD,R3       ;USE ACTUAL HIGH ADDRESS
 5000  023530  014143                  4$:     MOV     -(R1),-(R3)     ;MOVE A WORD
 5001  023532  005302                          DEC     R2              ;COUNT IT
 5002  023534  001375                          BNE     4$              ;BR IF MORE TO GO
 5003  023536  005037  001322                  CLR     LDRFLG          ;CLEAR THE LOADER MOVED FLAG
 5004  023542  104413                  5$:     RESREG                  ;RESTORE R0 - R5
 5005  023544  000207                          RTS     PC              ;RETURN
```

```
                                        ;ROUTINE TO WAIT FOR DRIVE READY
023546  032764  100000  000000  DRVRDY: BIT     #SURDY,RPDS(R4)  ;DRIVE READY ?
023554  001042                  BNE     6$               ;BR IF READY
023556  013746  001212          MOV     TIMOUT,-(SP)     ;WAIT TIME: APPROX 1 SEC
023562  005046                  CLR     -(SP)            ;CLEAR WORKING LOCATION
023564  162766  000001  000002  1$:     SUB     #1,2(SP) ;ANY TIME LEFT IN WAIT LOOP ?
023572  001411                  BEQ     3$               ;BR IF NOT
023574  012716  000144          MOV     #100.,(SP)       ;VALUE FOR APPROX 1 MS
023600  032764  100000  000000  2$:     BIT     #SURDY,RPDS(R4) ;DRIVE READY ?
023606  001023                  BNE     5$               ;BR IF READY
023610  005316                  DEC     (SP)             ;COUNT SOME TIME
023612  001372                  BNE     2$               ;BR IF NOT FINISHED
023614  000763                  BR      1$               ;COUNT MAJOR TIME AGAIN
023616  004737  023240          3$:     JSR     PC,SAVRP         ;STORE RP11 REGISTERS FOR ERROR MESSAGE
023622  012737  023632  001164          MOV     #4$,$ESCAPE      ;ESCAPE TO 4$ ON ERROR
023630  104100                  ERROR   100              ;DRIVE TIMED OUT
023632  005037  001164          4$:     CLR     $ESCAPE          ;CLEAR THE 'ESCAPE' FLAG
023636  062706  000004                  ADD     #4,SP            ;CORRECT  THE STACK POINTER
023642  032777  001000  155270          BIT     #SW09,@SWR       ;LOOP ON THE ERROR ?
023650  001336                  BNE     DRVRDY           ;LOOP IF SET
023652  000177  155322          JMP     @BYPASS          ;GO THE 'BYPASS' ADDRESS
023656  062706  000004          5$:     ADD     #4,SP            ;CORRECT THE STACK POINTER
023662  000207                  6$:     RTS     PC               ;RETURN

                                        ;ROUTINE TO WAIT FOR CONTROLLER READY

023664  032764  000200  000004  CONRDY: BIT     #RDY,RPCS(R4)    ;CONTROLLER READY ?
023672  001042                  BNE     6$               ;BR IF READY
023674  013746  001212          MOV     TIMOUT,-(SP)     ;WAIT TIME: APPROX 1 SEC
023700  005046                  CLR     -(SP)            ;CLEAR WORKING LOCATION
023702  162766  000001  000002  1$:     SUB     #1,2(SP) ;ANY TIME LEFT IN WAIT LOOP ?
023710  001411                  BEQ     3$               ;BR IF NOT
023712  012716  000144          MOV     #100.,(SP)       ;VALUE FOR APPROX 1 MS
023716  032764  000200  000004  2$:     BIT     #RDY,RPCS(R4)    ;CONTROLLER READY ?
023724  001023                  BNE     5$               ;BR IF READY
023726  005316                  DEC     (SP)             ;COUNT SOME TIME
023730  001372                  BNE     2$               ;BR IF NOT FINISHED
023732  000763                  BR      1$               ;COUNT MAJOR TIME AGAIN
023734  004737  023240          3$:     JSR     PC,SAVRP         ;STORE RP11 REGISTERS FOR ERROR MESSAGE
023740  012737  023750  001164          MOV     #4$,$ESCAPE      ;ESCAPE TO 4$ ON ERROR
023746  104101                  ERROR   101              ;CONTROLLER TIMED OUT
023750  005037  001164          4$:     CLR     $ESCAPE          ;CLEAR THE 'ESCAPE' FLAG
023754  062706  000004                  ADD     #4,SP            ;CORRECT THE STACK POINTER
023760  032777  001000  155152          BIT     #SW09,@SWR       ;LOOP ON THE ERROR ?
023766  001336                  BNE     CONRDY           ;BR IF LOOPING
023770  000177  155204          JMP     @BYPASS          ;'BYPASS' THE TEST
023774  062706  000004          5$:     ADD     #4,SP            ;CORRECT THE STACK POINTER
024000  000207                  6$:     RTS     PC               ;RETURN

                                        ;ROUTINE TO DETERMINE THE NUMBER OF SECTORS ON THE DRIVE BEING TESTED

024002  010046          SECNBR: MOV     R0,-(SP)         ;SAVE R0
024004  012737  117230  001244          MOV     #40600.,PAKSIZ   ;START WITH THE NUMBER OF SECTORS ON
```

```
5062  024012  005037  001246                  CLR     PAKSIZ+2            ;AN RP02
5063  024016  005000                          CLR     R0                  ;USE R0 AS AN INDEX
5064  024020  113700  002500                  MOVB    DPB,R0              ;GET THE DRIVE NUMBER
5065  024024  136037  001360  001304          BITB    ATABIT(R0),DRVTYP   ;IS THE DRIVE AN RP03 ?
5066  024032  001404                          BEQ     1$                  ;BR IF NOT
5067  024034  006337  001244                  ASL     PAKSIZ              ;DOUBLE THE SECTOR COUNT
5068  024040  006137  001246                  ROL     PAKSIZ+2            ;GET THE CARRY
5069  024044  012600                  1$:     MOV     (SP)+,R0            ;RESTORE R0
5070  024046  000207                          RTS     PC                  ;RETURN
5071
5072                                  ;ROUTINE TO SELECT A PATTERN
5073
5074  024050  010046                  PATSEL: MOV     R0,-(SP)            ;SAVE R0
5075  024052  005237  001204          1$:     INC     PATNUM              ;INCREMENT THE PATTERN NUMBER
5076  024056  022737  000020  001204          CMP     #16.,PATNUM         ;NUMBER AT MAXIMUM
5077  024064  101002                          BHI     2$                  ;BR IF NOT
5078  024066  005037  001204                  CLR     PATNUM              ;START AT ZERO AGAIN
5079  024072  013700  001204          2$:     MOV     PATNUM,R0           ;SEE IF THE PATTERN IS IN USE
5080  024076  006300                          ASL     R0                  ;CONVERT INTO A TABLE INDEX
5081  024100  036037  001400  001300          BIT     BITS(R0),PATTRN     ;INUSE INDICATOR SET FOR THIS PATTERN ?
5082  024106  001761                          BEQ     1$                  ;BR IF NOT
5083  024110  012600                          MOV     (SP)+,R0            ;RESTORE R0
5084  024112  000207                          RTS     PC                  ;RETURN
5085
5086                                  ;ROUTINE TO INCREMENT THE DISK ADDRESS
5087                                  ;       RETURN + 2 WHEN THE END OF THE PACK IS REACHED
5088
5089  024114  005737  001260          INCADR: TST     OPRSEL              ;USING AN OPERATOR SPECIFIED ADDRESS ?
5090  024120  001066                          BNE     5$                  ;BR IF YES
5091  024122  005737  001246                  TST     PAKSIZ+2            ;SECTORS LEFT ?
5092  024126  003017                          BGT     1$                  ;BR IF AN UPPER COUNT
5093  024130  005737  001244                  TST     PAKSIZ              ;LOWER COUNT AT ZERO ?
5094  024134  001460                          BEQ     5$                  ;BR IF IT IS
5095  024136  023737  001244  001254          CMP     PAKSIZ,BLKS14       ;ENOUGH SECTORS LEFT FOR A FULL TRANSFER ?
5096  024144  103010                          BHIS    1$                  ;BR IF THERE ARE
5097  024146  013737  001244  002504          MOV     PAKSIZ,$WC          ;USE THE RESIDUE COUNT FOR THIS TRANSFER
5098  024154  000337  002504                  SWAB    $WC                 ;CONVERT TO A WORD COUNT
5099  024160  005037  001244                  CLR     PAKSIZ              ;FORCE COUNT TO ZERO
5100  024164  000410                          BR      2$                  ;GO AND INCREMENT THE ADDRESS
5101  024166  013737  001256  002504  1$:     MOV     WC14,$WC            ;USE THE STANDARD WORD COUNT
5102  024174  163737  001254  001244          SUB     BLKS14,PAKSIZ       ;DECREMENT THE TOTAL SECTORS AVAILABLE
5103  024202  005637  001246                  SBC     PAKSIZ+2            ;SUBTRACT ANY CARRY
5104  024206  005046                  2$:     CLR     -(SP)               ;CLEAR THE STACK
5105  024210  113716  002512                  MOVB    $SEC,(SP)           ;GET CURRENT SECTOR
5106  024214  063716  001250                  ADD     INCSEC,(SP)         ;ADD THE SECTOR INCREMENT
5107  024220  121627  000012                  CMPB    (SP),#10.           ;EXCEED THE MAXIMUM ?
5108  024224  002404                          BLT     3$                  ;BR IF NOT
5109  024226  162716  000012                  SUB     #10.,(SP)           ;CORRECT THE RESIDUE
5110  024232  105237  002513                  INCB    $TRK                ;INCREMENT THE TRACK
5111  024236  111637  002512          3$:     MOVB    (SP),$SEC           ;NEW SECTOR ADDRESS
5112  024242  113716  002513                  MOVB    $TRK,(SP)           ;GET CURRENT TRACK VALUE
5113  024246  063716  001252                  ADD     INCTRK,(SP)         ;ADD THE TRACK INCREMENT
5114  024252  121627  000024                  CMPB    (SP),#20.           ;EXCEED THE MAXIMUM ?
5115  024256  002404                          BLT     4$                  ;BR IF NOT
5116  024260  162716  000024                  SUB     #20.,(SP)           ;CORRECT THE SIZE
5117  024264  005237  002510                  INC     $CYL                ;INCREMENT THE CYLINDER ADDRESS
```

```
 5118  024270  112637  002513  4$:     MOVB    (SP)+,$TRK      ;NEW TRACK ADDRESS
 5119  024274  000402          5$:     BR      6$              ;EXIT
 5120  024276  062716  000002          ADD     #2,(SP)         ;INCREMENT RETURN - AT END
 5121  024302  000207          6$:     RTS     PC              ;RETURN
 5122
 5123                          ;ROUTINE TO SETUP A BUFFER WITH THE SEQUENTIAL HEADERS FOR CYLINDER 0
 5124                          ;       TRACK 0
 5125
 5126  024304  104412          LODSEC: SAVREG                  ;SAVE R0 - R5
 5127  024306  013701  002506          MOV     $BUF,R1         ;BUFFER ADDRESS
 5128  024312  005003          1$:     CLR     R3              ;SECTOR COUNTER
 5129  024314  005021                  CLR     (R1)+           ;UPPER WORD OF HEADER
 5130  024316  005021                  CLR     (R1)+           ;CYLINDER/TRACK WORD OF HEADER
 5131  024320  010321                  MOV     R3,(R1)+        ;SECTOR ADDRESS
 5132  024322  005203                  INC     R3              ;INCREMENT THE SECTOR ADDRESS
 5133  024324  022703  000012          CMP     #10.,R3         ;FINISHED ?
 5134  024330  001371                  BNE     1$              ;BR IF NOT
 5135  024332  104413                  RESREG                  ;RESTORE R0 - R5
 5136  024334  000207                  RTS     PC              ;RETURN
 5137
 5138                          ;FILL THE BUFFER WITH THE SECTOR ADDRESS (CONTAINED IN '$SEC')
 5139
 5140  024336  104412          FILSEC: SAVREG                  ;STORE R0-R5
 5141  024340  005003                  CLR     R3              ;CLEAR THE INDEX
 5142  024342  113703  002512          MOVB    $SEC,R3         ;SECTOR ADDRESS
 5143  024346  001002                  BNE     1$              ;BR IF NOT SECTOR ZERO
 5144  024350  012703  000012          MOV     #10.,R3         ;USE 10. AS VALUE FOR SECTOR ZERO
 5145  024354  013702  002504  1$:     MOV     $WC,R2          ;WORD COUNT
 5146  024360  013701  002506          MOV     $BUF,R1         ;BUFFER START ADDRESS
 5147  024364  010321          2$:     MOV     R3,(R1)+        ;MOVE DATA INTO THE BUFFER
 5148  024366  005302                  DEC     R2              ;DECREMENT THE COUNTER
 5149  024370  001375                  BNE     2$              ;BR IF MORE TO DO
 5150  024372  104413                  RESREG                  ;RESTORE R0-R5
 5151  024374  000207                  RTS     PC              ;RETURN
 5152
 5153                          ;ROUTINE TO FILL THE BUFFER WITH THE TRACK ADDRESS (CONTAINED IN '$TRK')
 5154
 5155  024376  104412          FILTRK: SAVREG                  ;STORE R0-R5
 5156  024400  012701  043657          MOV     #BUFFER+1,R1    ;BUFFER ADDRESS
 5157  024404  013702  002504          MOV     $WC,R2          ;WORD COUNT
 5158  024410  005003                  CLR     R3              ;CLEAR THE DATA STORAGE
 5159  024412  113711  002513  1$:     MOVB    $TRK,(R1)       ;MOVE TRACK ADDRESS INTO THE BUFFER
 5160  024416  062701  000002          ADD     #2,R1           ;INCREMENT THE BUFFER POINTER
 5161  024422  005302                  DEC     R2              ;DECREMENT THE COUNTER
 5162  024424  001372                  BNE     1$              ;BR IF MORE TO DO
 5163  024426  104413                  RESREG                  ;RESTORE R0-R5
 5164  024430  000207                  RTS     PC              ;RETURN
 5165
 5166                          ;THIS ROUTINE WILL MOVE THE 16 WORDS OF THE
 5167                          ;DESIRED PATTERN INTO THE DATA BUFFER.
 5168
 5169  024432  104412          SETBUF: SAVREG                  ;SAVE R0 - R5
 5170  024434  013700  001204          MOV     PATNUM,R0       ;GET THE PATTERN NUMBER
 5171  024440  006300                  ASL     R0              ;CONVERT NUMBER TO INDEX
 5172  024442  013701  002506          MOV     $BUF,R1         ;FIRST ADDRESS
 5173  024446  013702  002504          MOV     $WC,R2          ;WORD COUNT
```

```
5174  024452  016003  001440          1$:     MOV     PAT.PT(R0),R3   ;PICKUP PATTERN POINTER
5175  024456  012704  000020                  MOV     #16.,R4         ;PATTERN COUNTER
5176  024462  012321                  2$:     MOV     (R3)+,(R1)+     ;MOVE WORD 'N' INTO DATA BUFFER
5177  024464  005302                          DEC     R2              ;DECREMENT THE COUNTER
5178  024466  001403                          BEQ     3$              ;BR IF DONE
5179  024470  005304                          DEC     R4              ;DECREMENT THE PATTERN COUNTER
5180  024472  001767                          BEQ     1$              ;START PATTERN AGAIN
5181  024474  000772                          BR      2$              ;CONTINUE
5182  024476  104413                  3$:     RESREG                  ;RESTORE R0 - R5
5183  024500  000207                          RTS     PC              ;RETURN
5184
5185                                  ;THIS ROUTINE LOADS THE PHYSICAL TRANSFER ADDRESS INTO '$BUF'
5186
5187  024502  013737  001224  002506  GETBUF: MOV     ACTMEM,$BUF     ;LOWER 16 BITS OF PHYSICAL ADDRESS
5188  024510  013746  001226                  MOV     ACTMEM+2,-(SP)  ;UPPER ADDRESS BITS ON THE STACK
5189  024514  042716  177774                  BIC     #^C3,(SP)       ;LEAVE ONLY THE LOWER 2 BITS
5190  024520  006316                          ASL     (SP)            ;ALIGN UPPER ADDRESS BITS
5191  024522  006316                          ASL     (SP)            ;ALIGN UPPER ADDRESS BITS
5192  024524  006316                          ASL     (SP)            ;ALIGN UPPER ADDRESS BITS
5193  024526  006316                          ASL     (SP)            ;ALIGN UPPER ADDRESS BITS
5194  024530  112637  002501                  MOVB    (SP)+,DPB+1     ;LOAD 'MEX' BITS
5195  024534  013746  002504                  MOV     $WC,-(SP)       ;PUT WORD COUNT ON THE STACK
5196  024540  006316                          ASL     (SP)            ;DOUBLE THE WORD COUNT
5197  024542  062637  001224                  ADD     (SP)+,ACTMEM    ;FORM NEXT STARTING ADDRESS
5198  024546  005537  001226                  ADC     ACTMEM+2        ;ADD ANY CARRY
5199  024552  023737  001226  001236          CMP     ACTMEM+2,HIMEM+2  ;EXCEEDING HIGHEST BANK ADDRESS
5200  024560  101005                          BHI     1$              ;BR IF HIGHER
5201  024562  103413                          BLO     2$              ;BR IF LOWER
5202  024564  023737  001224  001234          CMP     ACTMEM,HIMEM    ;CHECK LOWER 16 BITS
5203  024572  103407                          BLO     2$              ;BR IF LOWER
5204  024574  013737  001240  001224  1$:     MOV     LOMEM,ACTMEM    ;RESET ADDRESS
5205  024602  013737  001242  001226          MOV     LOMEM+2,ACTMEM+2  ;UPPER ADDRESS BITS
5206  024610  000734                          BR      GETBUF          ;TRY AGAIN
5207  024612  000207                  2$:     RTS     PC              ;RETURN
5208
5209                                  ;THIS ROUTINE COMPARES A 16 WORD DATA PATTERN AGAINST THE
5210                                  ;DATA BUFFER.  THE PATTERN NUMBER IS IN 'PATNUM'
5211
5212  024614  104412                  DATCMP: SAVREG                  ;SAVE R0-R5
5213  024616  005737  001222                  TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
5214  024622  001510                          BEQ     2$              ;BR IF NOT
```

```
5215  024624  005037  172340                      CLR     KIPAR0          ;LOAD ADDRESS REGISTER 0
5216  024630  012737  000200  172342              MOV     #200,KIPAR1     ;LOAD ADDRESS REGISTER 1
5217  024636  012737  000400  172344              MOV     #400,KIPAR2     ;LOAD ADDRESS REGISTER 2
5218  024644  012737  177600  172356              MOV     #177600,KIPAR7  ;ACCESS TO I/O PAGE
5219  024652  005046                              CLR     -(SP)           ;MAKE ROOM ON THE STACK
5220  024654  005046                              CLR     -(SP)           ;MAKE ROOM ON THE STACK
5221  024656  013766  002506  000002              MOV     $BUF,2(SP)      ;LOWER 16 BITS OF STARTING ADDRESS
5222  024664  113716  002501                      MOVB    DPB+1,(SP)      ;UPPER ADDRESS BITS
5223  024670  006216                              ASR     (SP)            ;RIGHT JUSTIFY THE ADDRESS BITS
5224  024672  006216                              ASR     (SP)            ;RIGHT JUSTIFY THE ADDRESS BITS
5225  024674  006216                              ASR     (SP)            ;RIGHT JUSTIFY THE ADDRESS BITS
5226  024676  006216                              ASR     (SP)            ;RIGHT JUSTIFY THE ADDRESS BITS
5227  024700  012703  000006                      MOV     #6,R3           ;LOAD COUNTER
5228  024704  006216                      1$:     ASR     (SP)            ;CONVERT ADDRESS TO BLOCK ADDRESS
5229  024706  006066  000002                      ROR     2(SP)           ;LOWER 16 BITS
5230  024712  005303                              DEC     R3              ;DECREMENT THE COUNTER
5231  024714  001373                              BNE     1$              ;CONTINUE UNTIL FINISHED
5232  024716  042766  000177  000002              BIC     #177,2(SP)      ;CLEAR LOWER BLOCK ADDRESS BITS
5233  024724  016637  000002  172346              MOV     2(SP),KIPAR3    ;LOAD ADDRESS REGISTER 3
5234  024732  062766  000200  000002              ADD     #200,2(SP)      ;INCREMENT BLOCK ADDRESS BY 4K
5235  024740  016637  000002  172350              MOV     2(SP),KIPAR4    ;LOAD ADDRESS REGISTER # 4
5236  024746  012737  077406  172300              MOV     #77406,KIPDR0   ;LOAD DESCRIPTOR REGISTER # 0
5237  024754  012737  077406  172302              MOV     #77406,KIPDR1   ;LOAD DESCRIPTOR REGISTER # 1
5238  024762  012737  077406  172304              MOV     #77406,KIPDR2   ;LOAD DESCRIPTOR REGISTER # 2
5239  024770  012737  077406  172306              MOV     #77406,KIPDR3   ;LOAD DESCRIPTOR REGISTER # 3
5240  024776  012737  077406  172310              MOV     #77406,KIPDR4   ;LOAD DESCRIPTOR REGISTER # 4
5241  025004  012737  077406  172316              MOV     #77406,KIPDR7   ;LOAD DESCRIPTOR REGISTER # 7
5242  025012  023727  002504  010000              CMP     $WC,#4096.      ;WAS TRANSFER GREATER THAN 4K ?
5243  025020  101411                              BLOS    2$              ;BR IF NOT
5244  025022  062766  000200  000002              ADD     #200,2(SP)      ;INCREMENT THE ADDRESS BY 4K
5245  025030  016637  000002  172352              MOV     2(SP),KIPAR5    ;LOAD ADDRESS REGISTER # 5
5246  025036  012737  077406  172312              MOV     #77406,KIPDR5   ;LOAD DESCRIPTOR REGISTER # 5
5247  025044  013701  002506              2$:     MOV     $BUF,R1         ;LOAD STARTING ADDRESS
5248  025050  005737  001222                      TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
5249  025054  001425                              BEQ     CMPAR           ;BR IF NOT
5250  025056  042701  140000                      BIC     #140000,R1      ;CHANGE PHYSICAL BASE TO VIRTUAL BASE
5251  025062  052701  060000                      BIS     #60000,R1       ;START AT VIRTUAL 12K
5252  025066  062706  000004                      ADD     #4,SP           ;CORRECT THE STACK POINTER
5253  025072  012737  025110  000004              MOV     #3$,ERRVEC      ;CHANGE THE ERROR VECTOR
5254  025100  012737  000020  172516              MOV     #20,@#SR3       ;ENABLE 22 BIT ADDRESSING MODE
5255  025106  000402                              BR      4$              ;BR IF AN 11/70
5256  025110  062706  000004              3$:     ADD     #4,SP           ;CORRECT THE STACK POINTER
5257  025114  012737  000006  000004      4$:     MOV     #ERRVEC+2,ERRVEC ;RESTORE THE ERROR VECTOR
5258  025122  012737  000001  177572              MOV     #1,@#SR0        ;ENABLE MEMORY MANAGEMENT
5259  025130  013737  001342  025420      CMPAR:  MOV     $RPWC,CMCNT     ;WORD COUNT TO WORKING LOCATION
5260  025136  063737  002504  025420              ADD     $WC,CMCNT       ;CALCULATE ACTUAL WORDS TRANSFERED
5261  025144  013737  002510  025422              MOV     $CYL,CMCYL      ;CYLINDER ADDRESS
5262  025152  113737  002513  025426              MOVB    $TRK,CMTRK      ;TRACK ADDRESS
5263  025160  113737  002512  025424              MOVB    $SEC,CMSEC      ;SECTOR ADDRESS
5264  025166  013737  001320  025416      CMSTR:  MOV     CMP.CT,LIMIT    ;DISPLAY LIMIT
5265  025174  005237  025416                      INC     LIMIT           ;CONVERT PARAMETER INTO LIMIT VALUE
5266  025200  010137  025430                      MOV     R1,CMBUF        ;STARTING ADDRESS OF SECTOR BUFFER
5267  025204  005037  025412                      CLR     FRSTER          ;CLEAR 'FIRST ERROR' INDICATOR
5268  025210  005037  025414                      CLR     ERCTR           ;CLEAR ERROR COUNTER
5269  025214  023727  025420  000400              CMP     CMCNT,#256.     ;IS BUFFER SIZE GREATER THAN ONE SECTOR ?
5270  025222  101003                              BHI     1$              ;BR IF IT IS
```

```
5271  025224  013702  025420              MOV     CMCNT,R2        ;LESS THAN, USE REMAINING BUFFER
5272  025230  000402                      BR      2$              ;
5273  025232  012702  000400        1$:   MOV     #256.,R2        ;COMPARE SECTOR
5274  025236  160237  025420        2$:   SUB     R2,CMCNT        ;DECREMENT WORD COUNT
5275  025242  013704  001204              MOV     PATNUM,R4       ;PATTERN NUMBER
5276  025246  006304                      ASL     R4              ;GENERATE INTO AN INDEX
5277  025250  012703  000020        CMDAT:MOV     #16.,R3         ;R3 IS PATTERN POSITION COUNTER
5278  025254  016405  001440              MOV     PAT.PT(R4),R5   ;PATTERN ADDRESS
5279  025260  022125              1$:   CMP     (R1)+,(R5)+     ;COMPARE BUFFER WITH PATTERN
5280  025262  001406                      BEQ     2$              ;BR IF EQUAL
5281  025264  032777  000010  153646      BIT     #SWD3,@SWR      ;SWITCH 3 SET ?
5282  025272  001002                      BNE     2$              ;BR IF NOT
5283  025274  004737  025432              JSR     PC,CMPRT        ;TYPE THE ERROR
5284  025300  005302              2$:   DEC     R2              ;DECREMENT SIZE COUNT
5285  025302  001403                      BEQ     3$              ;BR WHEN AT END
5286  025304  005303                      DEC     R3              ;DECREMENT PATT POS COUNT
5287  025306  001364                      BNE     1$              ;BR IF NOT AT END OF PATT
5288  025310  000757                      BR      CMDAT           ;RESTART THE PATTERN
5289  025312  004737  025630        3$:   JSR     PC,ENDCMP       ;PRINT LAST LINE (IF ERRORS)
5290  025316  005737  025420              TST     CMCNT           ;AT END OF BUFFER
5291  025322  003424                      BLE     CMPRX           ;BR IF AT END
5292  025324  005237  025424              INC     CMSEC           ;INCREMENT SECTOR
5293  025330  023727  025424  000012      CMP     CMSEC,#10.      ;SECTOR GREATER THAN MAX ?
5294  025336  103713                      BLO     CMSTR           ;BR IF NOT GREATER THAN MAX
5295  025340  005037  025424              CLR     CMSEC           ;CLEAR SECTOR ADDRESS
5296  025344  005237  025426              INC     CMTRK           ;INCREMENT TRACK
5297  025350  023727  025426  000024      CMP     CMTRK,#20.      ;TRACK GREATER THAN MAX ?
5298  025356  103703                      BLO     CMSTR           ;BR IF NOT GREATER
5299  025360  005037  025426              CLR     CMTRK           ;RESET TRACK ADDRESS
5300  025364  005237  025422              INC     CMCYL           ;INCREMENT CYLINDER ADDRESS
5301  025370  000137  025166              JMP     CMSTR           ;CONTINUE WITH COMPARE
5302  025374  104413              CMPRX:RESREG                    ;RESTORE R0 - R5
5303  025376  005737  001222              TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
5304  025402  001402                      BEQ     1$              ;BR IF NOT
5305  025404  005037  177572              CLR     @#SR0           ;TURN OFF MEMORY MANAGEMENT
5306  025410  000207              1$:   RTS     PC
5307
5308  025412  000000              FRSTER:.WORD  0               ;FIRST ERROR INDICATOR
5309  025414  000000              ERCTR: .WORD  0               ;NUMBER OF ERRORS
5310  025416  000000              LIMIT: .WORD  0               ;DISPLAY LIMIT
5311  025420  000000              CMCNT: .WORD  0               ;WORD COUNT
5312  025422  000000              CMCYL: .WORD  0               ;CYLINDER ADDRESS
5313  025424  000000              CMSEC: .WORD  0               ;SECTOR ADDRESS
5314  025426  000000              CMTRK: .WORD  0               ;TRACK ADDRESS
5315  025430  000000              CMBUF: .WORD  0               ;BEGINNING ADDRESS OF SECTOR
5316
5317                              ;TYPE DATA COMPARE ERRORS
5318
5319  025432  005737  025412        CMPRT:TST     FRSTER          ;FIRST ERROR?
5320  025436  001037                      BNE     CMPRT1          ;BR IF NOT
5321  025440  005737  001222              TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
5322  025444  001011                      BNE     1$              ;BR IF ACTIVE
5323  025446  012737  025446  001116      MOV     #.,$ERRPC       ;PC FOR ERROR MESSAGE
5324  025454  112737  000102  001114      MOVB    #102,$ITEMB     ;COMPARSION ERROR MESSAGE NUMBER
5325  025462  004737  017254              JSR     PC,TYPERR       ;REPORT THE ERROR
5326  025466  000412                      BR      2$              ;CONTINUE
```

```
5327  025470  005037  177572          1$:     CLR     @#SR0           ;TURN OFF MEMORY MANAGEMENT
5328  025474  012737  025474  001116          MOV     #.,$ERRPC       ;PC FOR ERROR MESSAGE
5329  025502  112737  000103  001114          MOVB    #103,$ITEMB     ;COMPARSION ERROR MESSAGE WITH MM REGISTERS
5330  025510  004737  017254                  JSR     PC,TYPERR       ;REPORT THE COMPARSION ERROR
5331  025514  012737  177777  025412  2$:     MOV     #-1,FRSTER      ;SET FIRST ERROR INDICATOR
5332  025522  005737  001222                  TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
5333  025526  001403                          BEQ     CMPRT1          ;BR IF OT
5334  025530  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEMORY MANAGEMENT AGAIN
5335  025536  005737  025416          CMPRT1: TST     LIMIT           ;TYPEOUT LIMIT REACHED ?
5336  025542  001403                          BEQ     1$              ;BR IF IT HAS
5337  025544  005337  025416                  DEC     LIMIT           ;DECREMENT LIMIT COUNTER
5338  025550  001004                          BNE     2$              ;BR IF NOT AT LIMIT
5339  025552  032777  000020  153360  1$:     BIT     #SW04,@SWR      ;PRINT ALL DATA COMPARE ERRORS ?
5340  025560  001420                          BEQ     3$              ;BR IF NOT
5341  025562  010137  001122          2$:     MOV     R1,$BDADR       ;ADDRESS OF BAD WORD
5342  025566  162737  000002  001122          SUB     #2,$BDADR       ;ADJUST ADDRESS
5343  025574  016537  177776  001124          MOV     -2(R5),$GDDAT   ;GOOD DATA
5344  025602  016137  177776  001126          MOV     -2(R1),$BDDAT   ;BAD DATA
5345  025610  112737  000104  001114          MOVB    #104,$ITEMB     ;ERROR TABLE INDEX
5346  025616  004737  017254                  JSR     PC,TYPERR       ;DISPLAY WORD THAT DIDN'T COMPARE
5347  025622  005237  025414          3$:     INC     ERCTR           ;COUNT THE ERROR
5348  025626  000207                          RTS     PC              ;RETURN
5349
5350                                  ;LAST LINE OF COMPARE ERROR REPORTING
5351
5352  025630  005737  025414          ENDCMP: TST     ERCTR           ;SEE IF ANY ERRORS
5353  025634  001406                          BEQ     2$              ;BR IF NONE
5354  025636  012737  025646  001164          MOV     #1$,$ESCAPE     ;;ESCAPE TO 1$ ON ERROR
5355  025644  104105                          ERROR   105             ;NUMBER OF COMPARE ERRORS
5356  025646  005037  001164          1$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
5357  025652  000207                  2$:     RTS     PC              ;RETURN
5358
5359                                  ;ROUTINE TO CLEAR THE RP11E
5360                                  ;
5361                                  ;THE FIRST 'CLEAR' TERMINATES ANY ORDER WHICH IS IN PROGRESS.  THE SECOND
5362                                  ;'CLEAR' CLEARS THE 'PROGRAM ERROR' WHICH MAY HAVE BEEN SET BY THE FIRST
5363                                  ;'CLEAR'.
5364                                  ;
5365                                  ;NOTE:  R4 MUST CONTAIN THE BASE ADDRESS OF THE RP11E
5366
5367  025654  012764  000001  000004  CLRP:   MOV     #CLEAR,RPCS(R4) ;CLEAR THE RP11
5368  025662  051616                  BIS     (SP),(SP)       ;DELAY FOR...
5369  025664  051616                  BIS     (SP),(SP)       ;...AT LEAST
5370  025666  051616                  BIS     (SP),(SP)       ;...4 MICROSECONDS.
5371  025670  012764  000001  000004          MOV     #CLEAR,RPCS(R4) ;CLEAR THE RP11
5372  025676  000207                          RTS     PC              ;RETURN
5373
5374                                  ;ROUTINE TO SIZE MEMORY AND TO SETUP FOR TEST 14
5375
5376  025700  012737  000200  022232  SIZMEM: MOV     #BIT07,$KT11    ;TELL THE 'SIZE' ROUTINE TO USE MM
5377  025706  004737  022174  6$:     JSR     PC,$SIZE        ;FIND OUT HOW MUCH MEMORY AND IF THE
5378                                                          ;SYSTEM HAS A KT11
5379  025712  005737  022232                  TST     $KT11           ;DOES THE SYSTEM HAVE A KT11 ?
5380  025716  100060                          BPL     2$              ;BR IF NOT
5381  025720  023727  022500  001540          CMP     $LSTBK,#1540    ;MORE THAN 28K ON THE SYSTEM ?
5382  025726  101003                          BHI     1$              ;BR IF MORE THAN 28K
```

```
5383  025730  005037  022232                    CLR     $KT11           ;CLEAR MEMORY MANAGEMENT INDICATOR
5384  025734  000764                            BR      6$              ;DO IT AGAIN WITH MEM. MAN. DESELECTED
5385  025736  012737  157776  001302  1$:       MOV     #157776,MEMSIZ  ;ADDRESS OF LAST NON MEMORY MANAGEMENT LOCATION
5386  025744  013737  022500  001230            MOV     $LSTBK,MAXMEM   ;HIGH MEMORY BANK ADDRESS
5387  025752  005037  001232                    CLR     MAXMEM+2        ;CLEAR UPPER MEMORY ADDRESS BITS
5388  025756  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
5389  025762  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
5390  025766  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
5391  025772  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
5392  025776  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
5393  026002  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
5394  026006  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
5395  026012  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
5396  026016  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
5397  026022  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
5398  026026  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
5399  026032  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
5400  026036  063737  022476  001230            ADD     $LSTAD,MAXMEM   ;ADD THE SIZE OF THE LAST BANK
5401  026044  005537  001232                    ADC     MAXMEM+2        ;HANDLE ANY CARRY
5402  026050  012737  020000  001264            MOV     #8192.,MAXWC    ;SETUP FOR MAXIMUM TRANSFER SIZE
5403  026056  000433                            BR      5$              ;EXIT
5404  026060  013737  022476  001302  2$:       MOV     $LSTAD,MEMSIZ   ;HIGH NON MEMORY MANAGEMENT ADDRESS
5405  026066  013737  022476  001230            MOV     $LSTAD,MAXMEM   ;HIGH NON MEMORY MANAGEMENT ADDRESS
5406  026074  005037  001232                    CLR     MAXMEM+2        ;DON'T NEED THE UPPER WORD
5407  026100  013737  001230  001264            MOV     MAXMEM,MAXWC    ;CONVERT MEMORY SIZE TO WORD COUNT
5408  026106  162737  043656  001264            SUB     #BUFFER,MAXWC   ;SUBTRACT STARTING ADDRESS OF BUFFER
5409  026114  000241                            CLC                     ;CLEAR THE 'C' BIT
5410  026116  006037  001264                    ROR     MAXWC           ;CHANGE TO WORD COUNT
5411  026122  105737  000041                    TSTB    41              ;SEE WHICH LOADER
5412  026126  001004                            BNE     4$              ;BR IF 'XXDP'
5413  026130  162737  000144  001264  3$:       SUB     #100.,MAXWC     ;'ABS' LOADER SIZE (PLUS A LITTLE BIT)
5414  026136  000403                            BR      5$              ;EXIT
5415  026140  162737  002734  001264  4$:       SUB     #1500.,MAXWC    ;'XXDP' SIZE
5416  026146  000207                  5$:       RTS     PC              ;RETURN
5417
5418                                    ;THIS ROUTINE IS USED TO ENSURE THAT THE BUS ADDRESS
5419                                    ;OF THE RP11 IS SETUP TO READ THE PROPER VALUE.
5420                                    ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
5421                                    ;REQUIRED.
5422                                    ;NOTE: THIS ROUTINE DESTROYS R0 - R4
5423
5424  026150  005737  001216            GETADR: TST     BUSADR          ;INPUT FROM TTY REQUESTED?
5425  026154  001425                            BEQ     3$              ;NO--BRANCH
5426  026156  005037  001216                    CLR     BUSADR          ;YES--CLEAR THE REQUEST FLAG
5427  026162  104401  001173            1$:     TYPE    ,$CRLF          ;CR-LF
5428  026166  104401  032162                    TYPE    ,MRPADR         ;'RPADR='
5429  026172  013746  001324                    MOV     RPADR,-(SP)     ;;SAVE RPADR FOR TYPEOUT
5430                                            ;;RP11 ADDRESS
5431  026176  104403                            TYPOS                   ;;GO TYPE--OCTAL ASCII
5432  026200     006                            .BYTE   6               ;;TYPE 6 DIGIT(S)
5433  026201     001                            .BYTE   1               ;;TYPE LEADING ZEROS
5434  026202  104401  031322                    TYPE    ,SLASH          ;'/'
5435  026206  104411                            RDLIN                   ;GET THE ENTRY
5436  026210  012601                            MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
5437  026212  004037  027136                    JSR     R0,CK.NUM       ;CHECK THE NUMBER
5438  026216  026162                            1$                      ;ILLEGAL INPUT
```

```
5439  026220  026224                          2$                      ;TERMINATED WITH A 'CR'
5440  026222  026230                          3$                      ;NO ENTRY - 'CR' ONLY
5441  026224  010237  001324          2$:     MOV     R2,RPADR        ;STORE THE ADDRESS
5442  026230  013701  000004          3$:     MOV     ERRVEC,R1       ;SAVE THE ERROR VECTOR
5443  026234  012737  026254  000004          MOV     #4$,ERRVEC      ;SETUP FOR TRAP
5444  026242  005777  153056                  TST     @RPADR          ;CHECK FOR RP11
5445  026246  010137  000004                  MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
5446  026252  000207                          RTS     PC              ;RETURN
5447  026254  010137  000004          4$:     MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
5448  026260  022626                          CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
5449  026262  012737  026272  001164          MOV     #5$,$ESCAPE     ;ESCAPE TO 5$ ON ERROR
5450  026270  104001                          ERROR   1               ;REPORT THE ERROR
5451  026272  005037  001164          5$:     CLR     $ESCAPE         ;CLEAR ERROR ESCAPE ADDRESS
5452  026276  005737  000042                  TST     @#42            ;IS THERE A MONITOR?
5453  026302  001727                          BEQ     1$              ;NO--GO ASK FOR ADDRESS
5454  026304  005037  016776                  CLR     $EOPCT          ;NO PASSES
5455  026310  000137  016616                  JMP     $EOP            ;GO TO END OF PROGRAM
5456
5457                                  ;THIS ROUTINE ALLOWS THE OPERATOR TO CONTROL THE PROGRAM
5458                                  ;       (CONVERSATION MODE)
5459
5460  026314  104401  031324          ENTPRM: TYPE    ,ENTNM          ;'ENTER TEST NUMBER'
5461  026320  104411                          RDLIN                   ;GET THE ENTRY
5462  026322  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
5463  026324  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
5464  026330  026314                          ENTPRM                  ;ILLEGAL INPUT
5465  026332  026336                          1$                      ;TERMINATED WITH A 'CR'
5466  026334  026362                          2$                      ;NO ENTRY - 'CR' ONLY
5467  026336  022702  000020          1$:     CMP     #STN,R2         ;VALID NUMBER ?
5468  026342  101764                          BLOS    ENTPRM          ;BR IF NOT
5469  026344  006302                          ASL     R2              ;CONVERT INTO A BIT INDEX
5470  026346  016237  001400  001276          MOV     BITS(R2),TSTNMS ;TEST SELECTION BIT
5471  026354  006202                          ASR     R2              ;CHANGE BACK INTO A WORD COUNT
5472  026356  010246                          MOV     R2,-(SP)        ;SAVE THE TEST NUMBER ENTRY
5473  026360  000405                          BR      ENTPR1          ;GET THE DRIVE NUMBER
5474  026362  012737  017777  001276  2$:     MOV     #17777,TSTNMS   ;SELECT ALL TESTS
5475  026370  012746  177777                  MOV     #-1,-(SP)       ;ALL TESTS SELECTED
5476  026374  005037  001312          ENTPR1: CLR     DRVSEL          ;CLEAR DRIVE SELECTION WORD
5477  026400  004737  022566                  JSR     PC,RPINIT       ;CHECK THE DRIVE STATUS
5478  026404  012737  000312  001210          MOV     #202.,MAXCYL    ;ASSUME ONLY RP02'S
5479  026412  005737  001304                  TST     DRVTYP          ;SEE WHICH TYPE
5480  026416  001403                          BEQ     1$              ;BR IF ONLY RP02'S
5481  026420  012737  000625  001210          MOV     #405.,MAXCYL    ;RP03'S
5482  026426  104401  031352          1$:     TYPE    ,DRVNM          ;ENTER DRIVE NUMBER
5483  026432  104411                          RDLIN                   ;GET THE ENTRY
5484  026434  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
5485  026436  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
5486  026442  026426                          1$                      ;ILLEGAL INPUT
5487  026444  026450                          2$                      ;TERMINATED WITH A 'CR'
5488  026446  026524                          5$                      ;NO ENTRY - 'CR' ONLY
5489  026450  022702  000010          2$:     CMP     #8.,R2          ;VALID ENTRY ?
5490  026454  101764                          BLOS    1$              ;BR IF NOT
5491  026456  116237  001360  001312          MOVB    ATABIT(R2),DRVSEL ;DRIVE SELECTION BIT
5492  026464  105762  001370                  TSTB    DRVSTA(R2)      ;IS DRIVE ONLINE
5493  026470  003035                          BGT     8$              ;BR IF ONLINE
5494  026472  104401  031377                  TYPE    ,DRIVE          ;'DRIVE'
```

```
5495  026476  105762  001370                  TSTB    DRVSTA(R2)      ;CHECK DRIVE STATUS AGAIN
5496  026502  100403                          BMI     3$              ;BR IF UNSAFE
5497  026504  104401  027312                  TYPE    ,OFFLIN         ;'OFFLINE'
5498  026510  000402                          BR      4$              ;CONTINUE
5499  026512  104401  027276          3$:     TYPE    ,UNSAFE         ;'UNSAFE'
5500  026516  104401  001173          4$:     TYPE    ,$CRLF          ;CR-LF
5501  026522  000724                          BR      ENTPR1          ;TRY AGAIN
5502  026524  005001                  5$:     CLR     R1              ;CLEAR R1 - USE IT AS AN INDEX
5503  026526  005716                          TST     (SP)            ;ALL TESTS SELECTED ?
5504  026530  100403                          BMI     6$              ;BR IF THEY WERE
5505  026532  022716  000015                  CMP     #15,(SP)        ;TEST 15, 16, OR 17 SELECTED ?
5506  026536  101733                          BLOS    1$              ;BR IF YES, SPECIFIC DRIVE NUMBER REQUIRED
5507  026540  105761  001370          6$:     TSTB    DRVSTA(R1)      ;IS THE DRIVE ONLINE ?
5508  026544  003403                          BLE     7$              ;BR IF NOT
5509  026546  156137  001360  001312          BISB    ATABIT(R1),DRVSEL ;SET THE SELECT BIT FOR THE DRIVE
5510  026554  005201                  7$:     INC     R1              ;INCREMENT THE DRIVE INDEX
5511  026556  020127  000010                  CMP     R1,#8.          ;REACHED MAXIMUM ?
5512  026562  103766                          BLO     6$              ;BR IF NOT
5513  026564  022726  000014          8$:     CMP     #14,(SP)+       ;IS TEST 14 SELECTED ?
5514  026570  001137                          BNE     ENTPRX          ;BR IF NOT
5515  026572  104401  031406          ENTPR2: TYPE    ,CHNGPM         ;CHANGE PARAMETERS FOR TEST 14 ?
5516  026576  104411                          RDLIN                   ;GET THE RESPONSE
5517  026600  123627  000131                  CMPB    @(SP)+,#'Y      ;'Y' ENTERED ?
5518  026604  001131                          BNE     ENTPRX          ;BR IF NOT
5519  026606  104401  031447                  TYPE    ,MXWRDS         ;'MAXIMUM WORD COUNT FOR TEST 14: '
5520  026612  013746  001264                  MOV     MAXWC,-(SP)     ;SAVE MAXWC FOR TYPEOUT
5521                                                                  ;MAXIMUM WORD COUNT FOR TEST 14
5522  026616  104403                          TYPOS                   ;GO TYPE--OCTAL ASCII
5523  026620     005                          .BYTE   5               ;TYPE 5 DIGIT(S)
5524  026621     000                          .BYTE   0               ;SUPPRESS LEADING ZEROS
5525  026622  104401  001173                  TYPE    ,$CRLF          ;CR-LF
5526  026626  104401  031523          1$:     TYPE    ,ENTWC          ;'ENTER WORD COUNT FOR TEST 14'
5527  026632  104411                          RDLIN                   ;GET THE ENTRY
5528  026634  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
5529  026636  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
5530  026642  026626                          1$                      ;ILLEGAL INPUT
5531  026644  026650                          2$                      ;TERMINATED WITH A 'CR'
5532  026646  026670                          ENTPR3                  ;NO ENTRY - 'CR' ONLY
5533  026650  020237  001264          2$:     CMP     R2,MAXWC        ;VALUE TOO LARGE ?
5534  026654  101364                          BHI     1$              ;BR IF TOO LARGE
5535  026656  020237  001262                  CMP     R2,LMT8K        ;NEW WORD COUNT LARGER THAN 8K ?
5536  026662  101361                          BHI     1$              ;BR IF LARGER
5537  026664  010237  001256                  MOV     R2,WC14         ;LOAD NEW WORD COUNT
5538  026670  104401  031565          ENTPR3: TYPE    ,CHNGPT         ;ENTER PATTERN CODE
5539  026674  104411                          RDLIN                   ;READ THE ENTRY
5540  026676  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
5541  026700  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
5542  026704  026670                          ENTPR3                  ;ILLEGAL INPUT
5543  026706  026712                          1$                      ;TERMINATED WITH A 'CR'
5544  026710  026716                          ENTPR4                  ;NO ENTRY - 'CR' ONLY
5545  026712  010237  001300          1$:     MOV     R2,PATTRN       ;PATTERN CODE
5546  026716  104401  031640          ENTPR4: TYPE    ,CHNGAD         ;USE A SPECIFIC DISK ADDRESS ?
5547  026722  104411                          RDLIN                   ;READ THE ENTRY
5548  026724  123627  000131                  CMPB    @(SP)+,#'Y      ;'Y' ENTERED ?
5549  026730  001057                          BNE     ENTPRX          ;BR IF NOT
5550  026732  104401  031713                  TYPE    ,ONEPAT         ;'A SINGLE DATA PATTERN MUST HAVE BEEN SPECIFIED:
```

```
                                                        ;'IF NOT, PATTERN 0 (ALL ZEROS) WILL BE USED'
026736  104401  032054          1$:     TYPE    ,ENTCYL ;'ENTER CYLINDER ADDRESS'
026742  104411                          RDLIN           ;GET THE ENTRY
026744  012601                          MOV     (SP)+,R1 ;STARTING ADDRESS OF INPUT ASCII STRING
026746  004037  027136                  JSR     R0,CK.NUM ;CHECK THE NUMBER
026752  026736                          1$              ;ILLEGAL INPUT
026754  026760                          2$              ;TERMINATED WITH A 'CR'
026756  026736                          1$              ;NO ENTRY - 'CR' ONLY
026760  020237  001210          2$:     CMP     R2,MAXCYL ;TOO LARGE ?
026764  101364                          BHI     1$      ;BR IF TOO LARGE
026766  010237  001266                  MOV     R2,SCYL ;LOAD CYLINDER ADDRESS
026772  104401  032105          3$:     TYPE    ,ENTRK  ;'ENTER TRACK ADDRESS'
026776  104411                          RDLIN           ;GET THE ENTRY
027000  012601                          MOV     (SP)+,R1 ;STARTING ADDRESS OF INPUT ASCII STRING
027002  004037  027136                  JSR     R0,CK.NUM ;CHECK THE NUMBER
027006  026772                          3$              ;ILLEGAL INPUT
027010  027014                          4$              ;TERMINATED WITH A 'CR'
027012  026772                          3$              ;NO ENTRY - 'CR' ONLY
027014  020227  000023          4$:     CMP     R2,#19. ;VALID ENTRY ?
027020  101364                          BHI     3$      ;BR IF NOT
027022  010237  001270                  MOV     R2,STRK ;TRACK ADDRESS
027026  104401  032133          5$:     TYPE    ,ENTSEC ;'ENTER SECTOR ADDRESS'
027032  104411                          RDLIN
027034  012601                          MOV     (SP)+,R1 ;STARTING ADDRESS OF INPUT ASCII STRING
027036  004037  027136                  JSR     R0,CK.NUM ;CHECK THE NUMBER
027042  027026                          5$              ;ILLEGAL INPUT
027044  027050                          6$              ;TERMINATED WITH A 'CR'
027046  027026                          5$              ;NO ENTRY - 'CR' ONLY
027050  020227  000011          6$:     CMP     R2,#9.  ;VALID ENTRY ?
027054  101364                          BHI     5$      ;BR IF NOT
027056  010237  001272                  MOV     R2,SSEC ;LOAD SECTOR ADDRESS
027062  012737  177777  001260          MOV     #-1,OPRSEL ;SET ADDRESS FLAG
027070  000207          ENTPRX: RTS     PC              ;RETURN

                        ;THIS ROUTINE IS USED TO CHECK IF AN ASCII CHARACTER IS A DIGIT
                        ;BETWEEN 0 AND 7.
                        ;
                                MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                        ;       JSR     R0,CK.OCT       ;CHECK THE CHARACTER
                        ;       RETURN1                 ;CHARACTER IS IN R2 AS A
                        ;                               ;OCTAL DIGIT
                        ;       RETURN2                 ;CHARACTER IS NOT BETWEEN 0-7

027072  121127  000060  CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
027076  103407                  BLO     1$              ;YES -- BRANCH
027100  121127  000067          CMPB    (R1),#'7        ;GREATER THAN SEVEN?
027104  101004                  BHI     1$              ;YES -- BRANCH
027106  111102                  MOVB    (R1),R2         ;GET THE CHARACTER
027110  042702  177770          BIC     #^C7,R2         ;STRIP AWAY THE ASCII
027114  000401                  BR      2$              ;BYPASS RETURN ADJUST
027116  005720          1$:     TST     (R0)+           ;ADJUST FOR RETURN
027120  000200          2$:     RTS     R0              ;RETURN

                        ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
                        ;DETERMINE WHAT IT IS.
                        ;
```

```
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
                                        ;       JSR     RO,CK.CHR       ;CHECK CHARACTER
                                        ;       RETURN  ADR1            ;UNKNOWN CHARACTER
                                        ;       RETURN  ADR2            ;CARRIAGE RETURN # (R1)=ADR+1

027122  105711                  CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
027124  001002                          BNE     1$              ;NO -- BRANCH
027126  005720                          TST     (R0)+           ;CARRIAGE RETURN
027130  005201                          INC     R1              ;MOVE POINTER TO NEXT CHARACTER
027132  011000                  1$:     MOV     (R0),R0         ;UNKNOWN CHARACTER
027134  000200                          RTS     R0              ;RETURN

                                        ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
                                        ;AND FORMS AN OCTAL NUMBER IN R2
                                        ;
                                        ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
                                        ;       JSR     R0,CK.NUM       ;GO FORM THE NUMBER
                                        ;       RETURN  ADR1            ;ILLEGAL CHARACTER IN THE INPUT STRING
                                        ;       RETURN  ADR2            ;"CR"--R2=NUMBER
                                        ;       RETURN  ADR3            ;'CR' WAS FIRST ENTRY

027136  010346                  CK.NUM: MOV     R3,-(SP)        ;SAVE R3
027140  005003                          CLR     R3              ;START NUMBER AT ZERO
027142  004037  027072                  JSR     R0,CK.OCT       ;OCTAL DIGIT?
027146  000404                          BR      1$              ;YES--BRANCH
027150  004037  027122                  JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
027154  027230                          6$                      ;ILLEGAL CHARACTER
027156  027222                          8$                      ;CARRIAGE RETURN
027160  005201                  1$:     INC     R1              ;MOVE TO NEXT CHARACTER
027162  006303                          ASL     R3              ;FOR THE OCTAL NUMBER IN R3
027164  103421                          BCS     6$              ;DON'T LET IT GET TO BIG
027166  006303                          ASL     R3
027170  103417                          BCS     6$
027172  006303                          ASL     R3
027174  103415                          BCS     6$
027176  060203                          ADD     R2,R3
027200  004037  027072                  JSR     R0,CK.OCT       ;IS THIS AN OCTAL DIGIT?
027204  000765                          BR      1$              ;YES--MAKE IT PART OF THE NUMBER
027206  010302                  2$:     MOV     R3,R2           ;SAVE THE OCTAL NUMBER
027210  005003                          CLR     R3              ;START WITH ZERO INDEX
027212                          3$:
027212  004037  027122                  JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
027216  027230                          6$                      ;ILLEGAL CHARACTER
027220  027224                          5$                      ;CARRIAGE RETURN
027222  005723                  8$:     TST     (R3)+           ;'CR' ONLY
027224  005723                  5$:     TST     (R3)+           ;"CR"
027226  060300                          ADD     R3,R0           ;YES--SAVE THE OCTAL NUMBER
027230  012603                  6$:     MOV     (SP)+,R3        ;RESTORE R3
027232  011000                          MOV     (R0),R0         ;PICKUP EXIT ADDRESS
027234  000200                          RTS     R0              ;RETURN

                                .SBTTL  TELETYPE MESSAGES

027236  005015  042012  044522  DRSTAT: .ASCIZ  <15><12><12>@DRIVE STATUS:@<15><12><12>
027244  042526  051440  040524
```

```
 5663  027252  052524  035123  005015
 5664  027260  000012
 5665  027262  020040  020040  047440  ONLINE: .ASCIZ  @     ONLINE@
 5666  027270  046116  047111  000105
 5667  027276  020040  020040  052440  UNSAFE: .ASCIZ  @     UNSAFE@
 5668  027304  051516  043101  000105
 5669  027312  020040  020040  047440  OFFLIN: .ASCIZ  @     OFFLINE@
 5670  027320  043106  044514  042516
 5671  027336      000
 5672  027327      040  051040  030120  RP02:   .ASCIZ  @  RP02@
 5673  027334  000062
 5674  027336  020040  050122  031460  RP03:   .ASCIZ  @  RP03@
 5675  027344      000
 5676  027345      104  044522  042526  DRVTST: .ASCIZ  @DRIVE(S) TO BE USED @
 5677  027352  051450  020051  047524
 5678  027360  041040  020105  051525
 5679  027366  042105  000040
 5680  027372  047516  042516      000  NONE:   .ASCIZ  @NONE@
 5681  027377      054      000  COMMA:  .ASCIZ  @,@
 5682  027401      015  005012  042524  TSTING: .ASCIZ  <15><12><12>@TESTING WITH DRIVE @
 5683  027406  052123  047111  020107
 5684  027414  044527  044124  042040
 5685  027422  044522  042526  000040
 5686  027430  020040      000          BLNKS2: .ASCIZ  @  @
 5687
 5688  027433      015  005012  047520  MSG15A: .ASCIZ  <15><12><12>@POWER FAIL TEST@
 5689  027440  042527  020122  040506
 5690  027446  046111  052040  051505
 5691  027454  000124
 5692  027456  005015  052012  051125  MSG15B: .ASCII  <15><12><12>@TURN CPU POWER  O F F  AFTER THIS MESSAGE, WAIT@<15><12>
 5693  027464  020116  050103  020125
 5694  027472  047520  042527  020122
 5695  027500  047440  043040  043040
 5696  027506  020040  043101  042524
 5697  027514  020122  044124  051511
 5698  027522  046440  051505  040523
 5699  027530  042507  020054  040527
 5700  027536  052111  005015
 5701  027542  020065  042523  047503          .ASCIZ  @5 SECONDS, THEN TURN CPU POWER  O N  AGAIN@<15><12>
 5702  027550  042116  026123  052040
 5703  027556  042510  020116  052524
 5704  027564  047122  041440  052520
 5705  027572  050040  053517  051105
 5706  027600  020040  020117  020116
 5707  027606  040440  040507  047111
 5708  027614  005015      000
 5709  027617      015  042412  042116  MSG15C: .ASCIZ  <15><12>@END OF POWER FAIL TEST@<15><12>
 5710  027624  047440  020106  047520
 5711  027632  042527  020122  040506
 5712  027640  046111  052040  051505
 5713  027646  006524  000012
 5714  027652  005015  050012  047522  MSG15D: .ASCIZ  <15><12><12>@PROGRAM WILL LOOP, WAITING FOR DRIVE TO COME ONLINE@<15><12
 5715  027660  051107  046501  053440
 5716  027666  046111  020114  047514
 5717  027674  050117  020054  040527
 5718  027702  052111  047111  020107
```

```
5719   027710  047506  020122  051104
5720   027716  053111  020105  047524
5721   027724  041440  046517  020105
5722   027732  047117  044514  042516
5723   027740  005015      000
5724
5725   027743     015  005012  042510  MSG16A: .ASCII  <15><12><12>@HEAD ALIGNMENT ROUTINE@
5726   027750  042101  040440  044514
5727   027756  047107  042515  052116
5728   027764  051040  052517  044524
5729   027772  042516
5730   027774  005015  041412  047117          .ASCII  <15><12><12>@CONNECT A JUMPER BETWEEN PINS J13M2 & J13C2 ON THE RP11E BA
5731   030002  042516  052103  040440
5732   030010  045040  046525  042520
5733   030016  020122  042502  053524
5734   030024  042505  020116  044520
5735   030032  051516  045040  031461
5736   030040  031115  023040  045040
5737   030046  031461  031103  047440
5738   030054  020116  044124  020105
5739   030062  050122  030461  020105
5740   030070  040502  045503  046120
5741   030076  047101      105
5742   030101     015  020012  051050          .ASCII  <15><12>@ (REFER TO THE APPLICABLE MAINTENANCE MANUAL FOR THE DRIVE SETU
5743   030106  043105  051105  052040
5744   030114  020117  044124  020105
5745   030122  050101  046120  041511
5746   030130  041101  042514  046440
5747   030136  044501  052116  047105
5748   030144  047101  042503  046440
5749   030152  047101  040525  020114
5750   030160  047506  020122  044124
5751   030166  020105  051104  053111
5752   030174  020105  042523  052524
5753   030202  027120      051
5754   030205     015  005012  047524          .ASCIZ  <15><12><12>@TOGGLE SWITCH 7 FOR HEAD SELECTION@<15><12>
5755   030212  043507  042514  051440
5756   030220  044527  041524  020110
5757   030226  020067  047506  020122
5758   030234  042510  042101  051440
5759   030242  046105  041505  044524
5760   030250  047117  005015      000
5761   030255     015  042412  052116  MSG16B: .ASCIZ  <15><12>@ENTER HEAD ADDRESS (IN OCTAL): @
5762   030262  051105  044040  040505
5763   030270  020104  042101  051104
5764   030276  051505  020123  044450
5765   030304  020116  041517  040524
5766   030312  024514  020072      000
5767
5768   030317     015  005012  050122  MSG17A: .ASCIZ  <15><12><12>@RP11E CONTROL PANEL SWITCH TEST@
5769   030324  030461  020105  047503
5770   030332  052116  047522  020114
5771   030340  040520  042516  020114
5772   030346  053523  052111  044103
5773   030354  052040  051505  000124
5774   030362  005015  054524  042520  MSG17B: .ASCIZ  <15><12>@TYPE A 'CR' TO START TEST AFTER EACH SETUP@<15><12>
```

```
5775  030370  040440  023440  051103
5776  030376  020047  047524  051440
5777  030404  040524  052122  052040
5778  030412  051505  020124  043101
5779  030420  042524  020122  040505
5780  030426  044103  051440  052105
5781  030434  050125  005015  000
5782  030441  015  042012  051511  MSG17C: .ASCIZ  <15><12>@DISABLE DRIVE # @
5783  030446  041101  042514  042040
5784  030454  044522  042526  021440
5785  030462  000040
5786  030464  005015  047105  041101  MSG17D: .ASCIZ  <15><12>@ENABLE DRIVE # @
5787  030472  042514  042040  044522
5788  030500  042526  021440  000040
5789  030506  005015  042523  020124  MSG17E: .ASCIZ  <15><12>@SET 'READ ONLY' ON DRIVE @
5790  030514  051047  040505  020104
5791  030522  047117  054514  020047
5792  030530  047117  042040  044522
5793  030536  042526  000040
5794  030542  005015  042523  020124  MSG17F: .ASCIZ  <15><12>@SET 'READ-WRITE' ON DRIVE @
5795  030550  051047  040505  026504
5796  030556  051127  052111  023505
5797  030564  047440  020116  051104
5798  030572  053111  020105  000
5799  030577  015  051412  052105  MSG17G: .ASCII  <15><12>@SET 'WRITE LOCKOUT' AND CLEAR THE LOA SWITCHES@
5800  030604  023440  051127  052111
5801  030612  020105  047514  045503
5802  030620  052517  023524  040440
5803  030626  042116  041440  042514
5804  030634  051101  052040  042510
5805  030642  046040  040517  051440
5806  030650  044527  041524  042510
5807  030656  123
5808  030657  015  047412  020116  .ASCIZ  <15><12>@ON THE RP11E CONTROL PANEL@
5809  030664  044124  020105  050122
5810  030672  030461  020105  047503
5811  030700  052116  047522  020114
5812  030706  040520  042516  000114
5813  030714  005015  042523  020124  MSG17H: .ASCIZ  <15><12>@SET THE FOLLOWING OCTAL VALUES IN THE CYLINDER LOA SWITCHES: @
5814  030722  044124  020105  047506
5815  030730  046114  053517  047111
5816  030736  020107  041517  040524
5817  030744  020114  040526  052514
5818  030752  051505  044440  020116
5819  030760  044124  020105  054503
5820  030766  044514  042116  051105
5821  030774  046040  040517  051440
5822  031002  044527  041524  042510
5823  031010  035123  000040
5824  031014  005015  042523  020124  MSG17I: .ASCIZ  <15><12>@SET SWITCHES TO @
5825  031022  053523  052111  044103
5826  031030  051505  052040  020117
5827  031036  000
5828  031037  015  051012  051505  MSG17J: .ASCII  <15><12>@RESET THE CYLINDER LOA SWITCHES@
5829  031044  052105  052040  042510
5830  031052  041440  046131  047111
```

```
 5831  031060  042504  020122  047514
 5832  031066  020101  053523  052111
 5833  031074  044103  051505
 5834  031100  005015  051412  052105          .ASCIZ  <15><12><12>@SET THE FOLLOWING OCTAL VALUES IN THE DRIVE LOA SWITCHES: @
 5835  031106  052040  042510  043040
 5836  031114  046117  047514  044527
 5837  031122  043516  047440  052103
 5838  031130  046101  053040  046101
 5839  031136  042525  020123  047111
 5840  031144  052040  042510  042040
 5841  031152  044522  042526  046040
 5842  031160  040517  051440  044527
 5843  031166  041524  042510  035123
 5844  031174  000040
 5845  031176  005015  042412  042116  MSG17K: .ASCII  <15><12><12>@END OF CONROL PANEL SWITCH TEST@
 5846  031204  047440  020106  047503
 5847  031212  051116  046117  050040
 5848  031220  047101  046105  051440
 5849  031226  044527  041524  020110
 5850  031234  042524  052123
 5851  031240  005015  042522  052524          .ASCIZ  <15><12>@RETURN RP11E CONTROL PANEL SWITCHES TO NORMAL@<15><12>
 5852  031246  047122  051040  030520
 5853  031254  042461  041440  047117
 5854  031262  051124  046117  050040
 5855  031270  047101  046105  051440
 5856  031276  044527  041524  042510
 5857  031304  020123  047524  047040
 5858  031312  051117  040515  006514
 5859  031320  000012
 5860
 5861  031322  000057                  SLASH:  .ASCIZ  @/@
 5862  031324  005015  047105  042524  ENTNM:  .ASCIZ  <15><12>@ENTER TEST NUMBER: @
 5863  031332  020122  042524  052123
 5864  031340  047040  046525  042502
 5865  031346  035122  000040
 5866  031352  047105  042524  020122  DRVNM:  .ASCIZ  @ENTER DRIVE NUMBER: @
 5867  031360  051104  053111  020105
 5868  031366  052516  041115  051105
 5869  031374  020072     000
 5870  031377     104  044522  042526  DRIVE:  .ASCIZ  @DRIVE @
 5871  031404  000040
 5872  031406  044103  047101  042507  CHNGPM: .ASCIZ  @CHANGE PARAMETERS FOR TEST 14 ? @
 5873  031414  050040  051101  046501
 5874  031422  052105  051105  020123
 5875  031430  047506  020122  042524
 5876  031436  052123  030440  020064
 5877  031444  020077     000
 5878  031447     115  054101  046511  MXWRDS: .ASCIZ  @MAXIMUM WORD COUNT FOR TEST 14 (IN OCTAL): @
 5879  031454  046525  053440  051117
 5880  031462  020104  047503  047125
 5881  031470  020124  047506  020122
 5882  031476  042524  052123  030440
 5883  031504  020064  044450  020116
 5884  031512  041517  040524  024514
 5885  031520  020072     000
 5886  031523     105  052116  051105  ENTWC:  .ASCIZ  @ENTER NEW WORD COUNT (IN OCTAL): @
```

```
 5887  031530  047040  053505  053440
 5888  031536  051117  020104  047503
 5889  031544  047125  020124  044450
 5890  031552  020116  041517  040524
 5891  031560  024514  020072     000
 5892  031565     105  052116  051105  CHNGPT: .ASCIZ  @ENTER PATTERN SELECTION CODE FOR TEST 14: @
 5893  031572  050040  052101  042524
 5894  031600  047122  051440  046105
 5895  031606  041505  044524  047117
 5896  031614  041440  042117  020105
 5897  031622  047506  020122  042524
 5898  031630  052123  030440  035064
 5899  031636  000040
 5900  031640  051525  020105  020101  CHNGAD: .ASCIZ  @USE A SPECIFIC DISK ADDRESS FOR TEST 14 ? @
 5901  031646  050123  041505  043111
 5902  031654  041511  042040  051511
 5903  031662  020113  042101  051104
 5904  031670  051505  020123  047506
 5905  031676  020122  042524  052123
 5906  031704  030440  020064  020077
 5907  031712     000
 5908  031713     040  040450  051440  ONEPAT: .ASCII  @ (A SINGLE DATA PATTERN MUST HAVE BEEN SPECIFIED@<15><12>
 5909  031720  047111  046107  020105
 5910  031726  040504  040524  050040
 5911  031734  052101  042524  047122
 5912  031742  046440  051525  020124
 5913  031750  040510  042526  041040
 5914  031756  042505  020116  050123
 5915  031764  041505  043111  042511
 5916  031772  006504     012
 5917  031775     040  043111  047040          .ASCIZ  @ IF NOT, PATTERN 0 (ALL ZEROS) WILL BE USED)@<15><12>
 5918  032002  052117  020054  040520
 5919  032010  052124  051105  020116
 5920  032016  020060  040450  046114
 5921  032024  055040  051105  051517
 5922  032032  020051  044527  046114
 5923  032040  041040  020105  051525
 5924  032046  042105  006451  000012
 5925  032054  047105  042524  020122  ENTCYL: .ASCIZ  @ENTER CYLINDER ADDRESS: @
 5926  032062  054503  044514  042116
 5927  032070  051105  040440  042104
 5928  032076  042522  051523  020072
 5929  032104     000
 5930  032105     105  052116  051105  ENTRK:  .ASCIZ  @ENTER TRACK ADDRESS: @
 5931  032112  052040  040522  045503
 5932  032120  040440  042104  042522
 5933  032126  051523  020072     000
 5934  032133     105  052116  051105  ENTSEC: .ASCIZ  @ENTER SECTOR ADDRESS: @
 5935  032140  051440  041505  047524
 5936  032146  020122  042101  051104
 5937  032154  051505  035123  000040
 5938
 5939  032162  050122  042101  020122  MRPADR: .ASCIZ  @RPADR = @
 5940  032170  020075     000
 5941
 5942                                   .SBTTL  ERROR MESSAGES
```

```
5943
5944  032173    122  030520  020061  EM1:    .ASCIZ  @RP11 DIDN'T RESPOND TO ADDRESSING@
5945  032200 044504  047104  052047
5946  032206 051040  051505  047520
5947  032214 042116  052040  020117
5948  032222 042101  051104  051505
5949  032230 044523  043516     000
5950  032235    103  047101  052047  EM2:    .ASCIZ  @CAN'T START READ COMMAND@
5951  032242 051440  040524  052122
5952  032250 051040  040505  020104
5953  032256 047503  046515  047101
5954  032264 000104
5955  032266 041447  042514  051101  EM3:    .ASCIZ  @'CLEAR' COMMAND DIDN'T TERMINATE DATA TRANSFER OPERATION@
5956  032274 020047  047503  046515
5957  032302 047101  020104  044504
5958  032310 047104  052047  052040
5959  032316 051105  044515  040516
5960  032324 042524  042040  052101
5961  032332 020101  051124  047101
5962  032340 043123  051105  047440
5963  032346 042520  040522  044524
5964  032354 047117     000
5965  032357    105  051122  051117  EM4:    .ASCIZ  @ERROR WRITING TEST SECTOR(S)@
5966  032364 053440  044522  044524
5967  032372 043516  052040  051505
5968  032400 020124  042523  052103
5969  032406 051117  051450  000051
5970  032414 051127  052111  020105  EM5:    .ASCIZ  @WRITE CHECK ERROR CHECKING TEST SECTOR(S)@
5971  032422 044103  041505  020113
5972  032430 051105  047522  020122
5973  032436 044103  041505  044513
5974  032444 043516  052040  051505
5975  032452 020124  042523  052103
5976  032460 051117  051450  000051
5977  032466 054105  042520  052103  EM6:    .ASCIZ  @EXPECTED 'WCE' DIDN'T OCCUR@
5978  032474 042105  023440  041527
5979  032502 023505  042040  042111
5980  032510 023516  020124  041517
5981  032516 052503  000122
5982  032522 051105  047522  020122  EM7:    .ASCIZ  @ERROR WRITING LESS THAN A FULL SECTOR@
5983  032530 051127  052111  047111
5984  032536 020107  042514  051523
5985  032544 052040  040510  020116
5986  032552 020101  052506  046114
5987  032560 051440  041505  047524
5988  032566 000122
5989  032570 047117  020105  043117  EM11:   .ASCIZ  @ONE OF 2 WORDS WRITTEN IN THE PARTIAL WRITE DIDN'T COMPARE@
5990  032576 031040  053440  051117
5991  032604 051504  053440  044522
5992  032612 052124  047105  044440
5993  032620 020116  044124  020105
5994  032626 040520  052122  040511
5995  032634 020114  051127  052111
5996  032642 020105  044504  047104
5997  032650 052047  041440  046517
5998  032656 040520  042522     000
```

```
5999  032663    116  047117  055055  EM12:   .ASCII  @NON-ZERO DATA IN PART OF SECTOR WHICH SHOULD HAVE@<15><12>
6000  032670 051105  020117  040504
6001  032676 040524  044440  020116
6002  032704 040520  052122  047440
6003  032712 020106  042523  052103
6004  032720 051117  053440  044510
6005  032726 044103  051440  047510
6006  032734 046125  020104  040510
6007  032742 042526  005015
6008  032746 042502  047105  055040          .ASCIZ  @BEEN ZERO FILLED DURING PARTIAL WRITE@
6009  032754 051105  020117  044506
6010  032762 046114  042105  042040
6011  032770 051125  047111  020107
6012  032776 040520  052122  040511
6013  033004 020114  051127  052111
6014  033012 000105
6015  033014 042447  050117  020047  EM13:   .ASCII  @'EOP' DIDN'T SET DURING A 2 SECTOR WRITE BEGINNING@
6016  033022 044504  047104  052047
6017  033030 051440  052105  042040
6018  033036 051125  047111  020107
6019  033044 020101  020062  042523
6020  033052 052103  051117  053440
6021  033060 044522  042524  041040
6022  033066 043505  047111  044516
6023  033074 043516
6024  033076 005015  052101  052040          .ASCIZ  <15><12>@AT THE LAST SECTOR OF THE PACK@
6025  033104 042510  046040  051501
6026  033112 020124  042523  052103
6027  033120 051117  047440  020106
6028  033126 044124  020105  040520
6029  033134 045503     000
6030  033137    047  051105  023522  EM14:   .ASCIZ  @'ERR' DIDN'T SET WITH 'EOP'@
6031  033144 042040  042111  023516
6032  033152 020124  042523  020124
6033  033160 044527  044124  023440
6034  033166 047505  023520     000
6035  033173    047  047505  023520  EM15:   .ASCIZ  @'EOP' DIDN'T CLEAR RPCA@
6036  033200 042040  042111  023516
6037  033206 020124  046103  040505
6038  033214 020122  050122  040503
6039  033222    000
6040  033223    123  041525  020101  EM16:   .ASCIZ  @SUCA NOT CORRECT AFTER 'EOP'@
6041  033230 047516  020124  047503
6042  033236 051122  041505  020124
6043  033244 043101  042524  020122
6044  033252 042447  050117  000047
6045  033260 051105  047522  020122  EM17:   .ASCIZ  @ERROR READING TEST SECTOR@
6046  033266 042522  042101  047111
6047  033274 020107  042524  052123
6048  033302 051440  041505  047524
6049  033310 000122
6050  033312 047503  052116  047105  EM20:   .ASCII  @CONTENTS OF FIRST SECTOR ON PACK CHANGED AFTER@<15><12>
6051  033320 051524  047440  020106
6052  033326 044506  051522  020124
6053  033334 042523  052103  051117
6054  033342 047440  020116  040520
```

```
 6055  033350  045503  041440  040510
 6056  033356  043516  042105  040440
 6057  033364  052106  051105  005015
 6058  033372  047506  041522  047111          .ASCIZ  @FORCING 'EOP' DURING WRITE@
 6059  033400  020107  042447  050117
 6060  033406  020047  052504  044522
 6061  033414  043516  053440  044522
 6062  033422  042524     000
 6063  033425     047  051120  043517  EM21:   .ASCII  @'PROG' ERROR DIDN'T SET WHEN READ COMMAND ISSUED@
 6064  033432  020047  051105  047522
 6065  033440  020122  044504  047104
 6066  033446  052047  051440  052105
 6067  033454  053440  042510  020116
 6068  033462  042522  042101  041440
 6069  033470  046517  040515  042116
 6070  033476  044440  051523  042525
 6071  033504     104
 6072  033505     015  053412  044510          .ASCIZ  <15><12>@WHILE CONTROLLER BUSY@
 6073  033512  042514  041440  047117
 6074  033520  051124  046117  042514
 6075  033526  020122  052502  054523
 6076  033534     000
 6077  033535     047  051105  023522  EM22:   .ASCIZ  @'ERR' DIDN'T SET WITH 'PROG'@
 6078  033542  042040  042111  023516
 6079  033550  020124  042523  020124
 6080  033556  044527  044124  023440
 6081  033564  051120  043517  000047
 6082  033572  051105  047522  020122  EM23:   .ASCIZ  @ERROR ATTEMPTING TO MISFORMAT SECTOR 0@
 6083  033600  052101  042524  050115
 6084  033606  044524  043516  052040
 6085  033614  020117  044515  043123
 6086  033622  051117  040515  020124
 6087  033630  042523  052103  051117
 6088  033636  030040     000
 6089  033641     105  051122  051117  EM24:   .ASCIZ  @ERROR VERIFYING THE MISFORMATTED TEST HEADER@
 6090  033646  053040  051105  043111
 6091  033654  044531  043516  052040
 6092  033662  042510  046440  051511
 6093  033670  047506  046522  052101
 6094  033676  042524  020104  042524
 6095  033704  052123  044040  040505
 6096  033712  042504  000122
 6097  033716  044515  043123  051117  EM25:   .ASCIZ  @MISFORMATTED TEST HEADER IS NOT CORRECT@
 6098  033724  040515  052124  042105
 6099  033732  052040  051505  020124
 6100  033740  042510  042101  051105
 6101  033746  044440  020123  047516
 6102  033754  020124  047503  051122
 6103  033762  041505  000124
 6104  033766  044047  043116  020047  EM26:   .ASCIZ  @'HNF' DIDN'T SET WHEN SEARCHING FOR MISFORMATTED SECTOR@
 6105  033774  044504  047104  052047
 6106  034002  051440  052105  053440
 6107  034010  042510  020116  042523
 6108  034016  051101  044103  047111
 6109  034024  020107  047506  020122
 6110  034032  044515  043123  051117
```

```
6111   034040  040515  052124  042105
6112   034046  051440  041505  047524
6113   034054  000122
6114   034056  042047  045523  020047  EM27:   .ASCIZ  @'DSK' NOT SET WITH 'HNF'@
6115   034064  047516  020124  042523
6116   034072  020124  044527  044124
6117   034100  023440  047110  023506
6118   034106     000
6119   034107     047  042510  020047  EM30:   .ASCIZ  @'HE' DIDN'T SET WITH 'HNF'@
6120   034114  044504  047104  052047
6121   034122  051440  052105  053440
6122   034130  052111  020110  044047
6123   034136  043116  000047
6124   034142  042447  051122  020047  EM31:   .ASCIZ  @'ERR' NOT SET WITH 'HNF'@
6125   034150  047516  020124  042523
6126   034156  020124  044527  044124
6127   034164  023440  047110  023506
6128   034172     000
6129   034173     105  051122  051117  EM32:   .ASCIZ  @ERROR WHILE RESTORING MISFORMATTED TEST HEADER@
6130   034200  053440  044510  042514
6131   034206  051040  051505  047524
6132   034214  044522  043516  046440
6133   034222  051511  047506  046522
6134   034230  052101  042524  020104
6135   034236  042524  052123  044040
6136   034244  040505  042504  000122
6137   034252  047503  052116  047522  EM33:   .ASCII  @CONTROLLER DIDN'T BECOME BUSY WHEN READ COMMAND@
6138   034260  046114  051105  042040
6139   034266  042111  023516  020124
6140   034274  042502  047503  042515
6141   034302  041040  051525  020131
6142   034310  044127  047105  051040
6143   034316  040505  020104  047503
6144   034324  046515  047101     104
6145   034331     015  044412  051523          .ASCIZ  <15><12>@ISSUED TO SEEKING DRIVE@
6146   034336  042525  020104  047524
6147   034344  051440  042505  044513
6148   034352  043516  042040  044522
6149   034360  042526     000
6150   034363     105  051122  051117  EM34:   .ASCIZ  @ERROR DURING READ COMMAND WHICH WAS ISSUED TO SEEKING DRIVE@
6151   034370  042040  051125  047111
6152   034376  020107  042522  042101
6153   034404  041440  046517  040515
6154   034412  042116  053440  044510
6155   034420  044103  053440  051501
6156   034426  044440  051523  042525
6157   034434  020104  047524  051440
6158   034442  042505  044513  043516
6159   034450  042040  044522  042526
6160   034456     000
6161   034457     104  052101  020101  EM35:   .ASCIZ  @DATA INCORRECT FROM HEADER READ COMMAND ISSUED TO A SEEKING DRIVE@
6162   034464  047111  047503  051122
6163   034472  041505  020124  051106
6164   034500  046517  044040  040505
6165   034506  042504  020122  042522
6166   034514  042101  041440  046517
```

```
6167  034522  040515  042116  044440
6168  034530  051523  042525  020104
6169  034536  047524  040440  051440
6170  034544  042505  044513  043516
6171  034552  042040  044522  042526
6172  034560     000
6173  034561     047  054116  042515  EM36:   .ASCIZ  @'NXME' DIDN'T SET WHEN LOCATION 760000 ADDRESSED@
6174  034566  020047  044504  047104
6175  034574  052047  051440  052105
6176  034602  053440  042510  020116
6177  034610  047514  040503  044524
6178  034616  047117  033440  030066
6179  034624  030060  020060  042101
6180  034632  051104  051505  042523
6181  034640  000104
6182  034642  044047  023505  042040  EM37:   .ASCIZ  @'HE' DIDN'T SET WITH 'NXME'@
6183  034650  042111  023516  020124
6184  034656  042523  020124  044527
6185  034664  044124  023440  054116
6186  034672  042515  000047
6187  034676  042447  051122  020047  EM40:   .ASCIZ  @'ERR' DIDN7T SET WITH 'NXME'@
6188  034704  044504  047104  052067
6189  034712  051440  052105  053440
6190  034720  052111  020110  047047
6191  034726  046530  023505     000
6192  034733     105  051122  051117  EM41:   .ASCIZ  @ERROR FORMATTING TRACK 0@
6193  034740  043040  051117  040515
6194  034746  052124  047111  020107
6195  034754  051124  041501  020113
6196  034762  000060
6197  034764  051105  047522  020122  EM42:   .ASCIZ  @ERROR READING THE HEADER FROM THE TEST SECTOR@
6198  034772  042522  042101  047111
6199  035000  020107  044124  020105
6200  035006  042510  042101  051105
6201  035014  043040  047522  020115
6202  035022  044124  020105  042524
6203  035030  052123  051440  041505
6204  035036  047524  000122
6205  035042  042523  052103  051117  EM43:   .ASCIZ  @SECTOR FIELD FROM HEADER IS NOT CORRECT@
6206  035050  043040  042511  042114
6207  035056  043040  047522  020115
6208  035064  042510  042101  051105
6209  035072  044440  020123  047516
6210  035100  020124  047503  051122
6211  035106  041505  000124
6212  035112  051105  047522  020122  EM44:   .ASCIZ  @ERROR WRITING SECTOR ADDRESS IN DATA FIELD@
6213  035120  051127  052111  047111
6214  035126  020107  042523  052103
6215  035134  051117  040440  042104
6216  035142  042522  051523  044440
6217  035150  020116  040504  040524
6218  035156  043040  042511  042114
6219  035164     000
6220  035165     047  047523  023524  EM45:   .ASCIZ  @'SOT' OR SECTOR ADDRESS REIGSTER IS NOT CORRECT@
6221  035172  047440  020122  042523
6222  035200  052103  051117  040440
```

```
6223 035206 042104 042522 051523
6224 035214 051040 044505 051507
6225 035222 042524 020122 051511
6226 035230 047040 052117 041440
6227 035236 051117 042522 052103
6228 035244    000
6229 035245    104 052101 020101   EM46:   .ASCIZ  @DATA READ FROM THE SECTOR IS NOT CORRECT@
6230 035252 042522 042101 043040
6231 035260 047522 020115 044124
6232 035266 020105 042523 052103
6233 035274 051117 044440 020123
6234 035302 047516 020124 047503
6235 035310 051122 041505 000124
6236 035316 042523 052103 051117   EM47:   .ASCII  @SECTOR CONTENTS WRONG, DATA SHOULD BE THE@<15><12>
6237 035324 041440 047117 042524
6238 035332 052116 020123 051127
6239 035340 047117 026107 042040
6240 035346 052101 020101 044123
6241 035354 052517 042114 041040
6242 035362 020105 044124 006505
6243 035370    012
6244 035371    124 040522 045503           .ASCIZ  @TRACK NUMBER OF THE CURRENTLY SELECTED HEAD@
6245 035376 047040 046525 042502
6246 035404 020122 043117 052040
6247 035412 042510 041440 051125
6248 035420 042522 052116 054514
6249 035426 051440 046105 041505
6250 035434 042524 020104 042510
6251 035442 042101    000
6252 035445    105 051122 051117   EM50:   .ASCIZ  @ERROR AFTER 2 WORD READ STARTING AT LOC 177776@
6253 035452 040440 052106 051105
6254 035460 031040 053440 051117
6255 035466 020104 042522 042101
6256 035474 051440 040524 052122
6257 035502 047111 020107 052101
6258 035510 046040 041517 030440
6259 035516 033467 033467 000066
6260 035524 046447 054105 023460   EM51:   .ASCII  @'MEX0' DIDN'T SET AFTER 2 WORD READ STARTING@<15><12>
6261 035532 042040 042111 023516
6262 035540 020124 042523 020124
6263 035546 043101 042524 020122
6264 035554 020062 047527 042122
6265 035562 051040 040505 020104
6266 035570 052123 051101 044524
6267 035576 043516 005015
6268 035602 052101 046040 041517           .ASCIZ  @AT LOC 177776@
6269 035610 030440 033467 033467
6270 035616 000066
6271 035620 046447 054105 023461   EM52:   .ASCIZ  @'MEX1' SET AFTER 2 WORD READ STARTING AT LOC 177776@
6272 035626 051440 052105 040440
6273 035634 052106 051105 031040
6274 035642 053440 051117 020104
6275 035650 042522 042101 051440
6276 035656 040524 052122 047111
6277 035664 020107 052101 046040
6278 035672 041517 030440 033467
```

```
 6279  035700  033467  000066
 6280  035704  047503  052116  047105  EM53:   .ASCII  @CONTENTS OF LOC 200000 WRONG AFTER 2 WORD READ@<15><12>
 6281  035712  051524  047440  020106
 6282  035720  047514  020103  030062
 6283  035726  030060  030060  053440
 6284  035734  047522  043516  040440
 6285  035742  052106  051105  031040
 6286  035750  053440  051117  020104
 6287  035756  042522  042101  005015
 6288  035764  052123  051101  044524          .ASCIZ  @STARTING AT LOC 177776@
 6289  035772  043516  040440  020124
 6290  036000  047514  020103  033461
 6291  036006  033467  033067     000
 6292  036013     105  051122  051117  EM54:   .ASCIZ  @ERROR AFTER 2 WORD READ STARTING AT LOC 377776@
 6293  036020  040440  052106  051105
 6294  036026  031040  053440  051117
 6295  036034  020104  042522  042101
 6296  036042  051440  040524  052122
 6297  036050  047111  020107  052101
 6298  036056  046040  041517  031440
 6299  036064  033467  033467  000066
 6300  036072  046447  054105  023460  EM55:   .ASCIZ  @'MEX0' DIDN'T CLEAR AFTER 2 WORD STARTING AT LOC 377776@
 6301  036100  042040  042111  023516
 6302  036106  020124  046103  040505
 6303  036114  020122  043101  042524
 6304  036122  020122  020062  047527
 6305  036130  042122  051440  040524
 6306  036136  052122  047111  020107
 6307  036144  052101  046040  041517
 6308  036152  031440  033467  033467
 6309  036160  000066
 6310  036162  046447  054105  023461  EM56:   .ASCIZ  @'MEX1' DIDN'T SET AFTER 2 WORD READ STARTING AT LOC 377776@
 6311  036170  042040  042111  023516
 6312  036176  020124  042523  020124
 6313  036204  043101  042524  020122
 6314  036212  020062  047527  042122
 6315  036220  051040  040505  020104
 6316  036226  052123  051101  044524
 6317  036234  043516  040440  020124
 6318  036242  047514  020103  033463
 6319  036250  033467  033067     000
 6320  036255     103  047117  042524  EM57:   .ASCII  @CONTENTS OF LOC 400000 WRONG AFTER 2 WORD READ@<15><12>
 6321  036262  052116  020123  043117
 6322  036270  046040  041517  032040
 6323  036276  030060  030060  020060
 6324  036304  051127  047117  020107
 6325  036312  043101  042524  020122
 6326  036320  020062  047527  042122
 6327  036326  051040  040505  006504
 6328  036334     012
 6329  036335     123  040524  052122          .ASCIZ  @STARTING AT LOCATION 377776@
 6330  036342  047111  020107  052101
 6331  036350  046040  041517  052101
 6332  036356  047511  020116  033463
 6333  036364  033467  033067     000
 6334  036371     105  051122  051117  EM60:   .ASCII  @ERROR RETURNING SECTOR TO PDP-11 MODE USING A@<15><12>
```

```
036376  051040  052105  051125
036404  044516  043516  051440
036412  041505  047524  020122
036420  047524  050040  050104
036426  030455  020061  047515
036434  042504  052440  044523
036442  043516  040440  005015
036450  020062  047527  042122          .ASCIZ  @2 WORD WRITE@
036456  053440  044522  042524
036464     000
036465     105  051122  051117  EM61:   .ASCIZ  @ERROR WRITING TEST SECTOR(S)@
036472  053440  044522  044524
036500  043516  052040  051505
036506  020124  042523  052103
036514  051117  051450  000051
036522  051105  047522  020122  EM62:   .ASCIZ  @ERROR READING TEST SECTOR(S)@
036530  042522  042101  047111
036536  020107  042524  052123
036544  051440  041505  047524
036552  024122  024523     000
036557     104  044522  042526  EM63:   .ASCII  @DRIVE DIDN'T RETURN TO CYL 0 FROM THE MAXIMUM@<15><12>
036564  042040  042111  023516
036572  020124  042522  052524
036600  047122  052040  020117
036606  054503  020114  020060
036614  051106  046517  052040
036622  042510  046440  054101
036630  046511  046525  005015
036636  054503  020114  047117          .ASCIZ  @CYL ON CONTROLLER POWER FAIL@
036644  041440  047117  051124
036652  046117  042514  020122
036660  047520  042527  020122
036666  040506  046111     000
036673     103  047117  042524  EM64:   .ASCII  @CONTENTS OF MEMORY CHANGED DURING POWER@<15><12>
036700  052116  020123  043117
036706  046440  046505  051117
036714  020131  044103  047101
036722  042507  020104  052504
036730  044522  043516  050040
036736  053517  051105  005015
036744  040506  046111  053440          .ASCIZ  @FAIL WHILE READING THE DISK@
036752  044510  042514  051040
036760  040505  044504  043516
036766  052040  042510  042040
036774  051511  000113
037000  051447  047525  023514  EM65:   .ASCIZ  @'SUOL' SET WITH DRIVE DISABLED@
037006  051440  052105  053440
037014  052111  020110  051104
037022  053111  020105  044504
037030  040523  046102  042105
037036     000
037037     047  052523  046117  EM66:   .ASCIZ  @'SUOL' NOT SET WITH DRIVE ENABLED@
037044  020047  047516  020124
037052  042523  020124  044527
037060  044124  042040  044522
037066  042526  042440  040516
```

```
6391  037074  046102  042105  000
6392  037101  047  052523  050127  EM67:   .ASCIZ  @'SUWP' NOT SET WITH DRIVE SET TO 'READ ONLY'@
6393  037106  020047  047516  020124
6394  037114  042523  020124  044527
6395  037122  044124  042040  044522
6396  037130  042526  051440  052105
6397  037136  052040  020117  051047
6398  037144  040505  020104  047117
6399  037152  054514  000047
6400  037156  051447  053525  023520  EM70:   .ASCIZ  @'SUWP' SET WITH DRIVE SET TO 'READ/WRITE'@
6401  037164  051440  052105  053440
6402  037172  052111  020110  051104
6403  037200  053111  020105  042523
6404  037206  020124  047524  023440
6405  037214  042522  042101  053457
6406  037222  044522  042524  000047
6407  037230  051447  053525  023520  EM71:   .ASCII  @'SUWP' SET WITH RP11 'WRITE LOCKOUT' SET, CYLINDER@<15><12>
6408  037236  051440  052105  053440
6409  037244  052111  020110  050122
6410  037252  030461  023440  051127
6411  037260  052111  020105  047514
6412  037266  045503  052517  023524
6413  037274  051440  052105  020054
6414  037302  054503  044514  042116
6415  037310  051105  005015
6416  037314  047514  023501  020123          .ASCIZ  @LOA'S = 0, AND RPCA = 0@
6417  037322  020075  026060  040440
6418  037330  042116  051040  041520
6419  037336  020101  020075  000060
6420  037344  051447  053525  023520  EM72:   .ASCIZ  @'SUWP' SET WITH RPCA = 2 & CYL LOA'S = 0@
6421  037352  051440  052105  053440
6422  037360  052111  020110  050122
6423  037366  040503  036440  031040
6424  037374  023040  041440  046131
6425  037402  046040  040517  051447
6426  037410  036440  030040  000
6427  037415  047  052523  050127  EM73:   .ASCIZ  @'SUWP' NOT SET; RPCA = VALUE IN CYL LOA'S@
6428  037422  020047  047516  020124
6429  037430  042523  035524  051040
6430  037436  041520  020101  020075
6431  037444  040526  052514  020105
6432  037452  047111  041440  046131
6433  037460  046040  040517  051447
6434  037466  000
6435  037467  047  052523  050127  EM74:   .ASCIZ  @'SUWP' SET WITH RPCA ONE GREATER THAN CYL LOA VALUE@
6436  037474  020047  042523  020124
6437  037502  044527  044124  051040
6438  037510  041520  020101  047117
6439  037516  020105  051107  040505
6440  037524  042524  020122  044124
6441  037532  047101  041440  046131
6442  037540  046040  040517  053040
6443  037546  046101  042525  000
6444  037553  047  052523  050127  EM75:   .ASCIZ  @'SUWP' SET WITH DRIVE LOA SWITCHES EQUAL TO SELECTED DRIVE@
6445  037560  020047  042523  020124
6446  037566  044527  044124  042040
```

```
6447  037574  044522  042526  046040
6448  037602  040517  051440  044527
6449  037610  041524  042510  020123
6450  037616  050505  040525  020114
6451  037624  047524  051440  046105
6452  037632  041505  042524  020104
6453  037640  051104  053111  000105
6454  037646  051104  053111  020105  EM76:   .ASCIZ  @DRIVE HAS GONE OFFLINE@
6455  037654  040510  020123  047507
6456  037662  042516  047440  043106
6457  037670  044514  042516     000
6458  037675     104  044522  042526  EM77:   .ASCIZ  @DRIVE IS UNSAFE@
6459  037703  044440  020123  047125
6460  037710  040523  042506     000
6461  037715     104  044522  042526  EM100:  .ASCIZ  @DRIVE TIMED OUT@
6462  037722  052040  046511  042105
6463  037730  047440  052124     000
6464  037735     103  047117  051124  EM101:  .ASCIZ  @CONTROLLER TIMED OUT@
6465  037742  046117  042514  020122
6466  037750  044524  042515  020104
6467  037756  052517  000124
6468  037762  040504  040524  041440  EM102:  .ASCIZ  @DATA COMPARE ERROR@
6469  037770  046517  040520  042522
6470  037776  042440  051122  051117
6471  040004     000
6472  040005     104  052101  020101  EM103:  .ASCIZ  @DATA COMPARE ERROR (MEMORY MANAGEMENT ENABLED)@
6473  040012  047503  050115  051101
6474  040020  020105  051105  047522
6475  040026  020122  046450  046505
6476  040034  051117  020131  040515
6477  040042  040516  042507  042515
6478  040050  052116  042440  040516
6479  040056  046102  042105  000051
6480  040064  042447  051122  020047  EM106:  .ASCIZ  @'ERR' DIDN'T SET WITH 'WCE'@
6481  040072  044504  047104  052047
6482  040100  051440  052105  053440
6483  040106  052111  020110  053447
6484  040114  042503  000047
6485  040120  044047  023505  042040  EM107:  .ASCIZ  @'HE' DIDN'T SET WITH 'PROG'@
6486  040126  042111  023516  020124
6487  040134  042523  020124  044527
6488  040142  044124  023440  051120
6489  040150  043517  000047
6490  040154  051104  053111  020105  EM112:  .ASCIZ  @DRIVE UNSAFE AFTER POWER FAIL@
6491  040162  047125  040523  042506
6492  040170  040440  052106  051105
6493  040176  050040  053517  051105
6494  040204  043040  044501  000114
6495  040212  051447  053525  023520  EM113:  .ASCIZ  @'SUWP' NOT SET WITH DRIVE LOA SWITCHES GREATER THAN SELECTED DRIVE@
6496  040220  047040  052117  051440
6497  040226  052105  053440  052111
6498  040234  020110  051104  053111
6499  040242  020105  047514  020101
6500  040250  053523  052111  044103
6501  040256  051505  043440  042522
6502  040264  052101  051105  052040
```

```
 6503  040272  040510  020116  042523
 6504  040300  042514  052103  042105
 6505  040306  042040  044522  042526
 6506  040314     000
 6507
 6508                              ;'DH' MESSAGE LINES
 6509
 6510  040315     122  040520  051104  DH1:    .ASCIZ  @RPADR@
 6511  040322     000
 6512  040323     124  051505  020124  DH2:    .ASCIZ  @TEST #  ERR PC  DRV #  RPDS  RPER  RPCS  RPWC@
 6513  040330  020043  042440  051122
 6514  040336  050040  020103  042040
 6515  040344  053122  021440  020040
 6516  040352  051040  042120  020123
 6517  040360  020040  051040  042520
 6518  040366  020122  020040  051040
 6519  040374  041520  020123  020040
 6520  040402  051040  053520  000103
 6521  040410  050122  040502  020040  DH2A:   .ASCIZ  @RPBA  RPCA  RPDA  RPM1  SUCA  SILO@
 6522  040416  020040  050122  040503
 6523  040424  020040  020040  050122
 6524  040432  040504  020040  020040
 6525  040440  050122  030515  020040
 6526  040446  020040  052523  040503
 6527  040454  020040  020040  044523
 6528  040462  047514     000
 6529  040465     124  051505  020124  DH5:    .ASCIZ  @TEST #  ERR PC  DRV #  PATRN #  RPDS  RPER  RPCS@
 6530  040472  020043  042440  051122
 6531  040500  050040  020103  042040
 6532  040506  053122  021440  020040
 6533  040514  050040  052101  047122
 6534  040522  021440  051040  042120
 6535  040530  020123  020040  051040
 6536  040536  042520  020122  020040
 6537  040544  051040  041520  000123
 6538  040552  050122  041527  020040  DH5A:   .ASCIZ  @RPWC  RPBA  RPCA  RPDA  RPM1  SUCA  SILO@
 6539  040560  020040  050122  040502
 6540  040566  020040  020040  050122
 6541  040574  040503  020040  020040
 6542  040602  050122  040504  020040
 6543  040610  020040  050122  030515
 6544  040616  020040  020040  052523
 6545  040624  040503  020040  020040
 6546  040632  044523  047514     000
 6547  040637     124  051505  020124  DH11:   .ASCIZ  @TEST #  ERR PC  DRV #  EXPT'D 1ST WORD 2ND WORD@
 6548  040644  020043  042440  051122
 6549  040652  050040  020103  042040
 6550  040660  053122  021440  020040
 6551  040666  042440  050130  023524
 6552  040674  020104  051461  020124
 6553  040702  047527  042122  031040
 6554  040710  042116  053440  051117
 6555  040716  000104
 6556  040720  042524  052123  021440  DH12:   .ASCIZ  @TEST #  ERR PC  DRV #  EXPT'D  RECV'D  LOCN@
 6557  040726  020040  051105  020122
 6558  040734  041520  020040  051104
```

```
 6559  040742  020126  020043  020040
 6560  040750  054105  052120  042047
 6561  040756  020040  042522  053103
 6562  040764  042047  020040  047514
 6563  040772  047103     000
 6564  040775     124  051505  020124  DH14:   .ASCIZ  @TEST #  ERR PC  DRV #   RPDS    RPER    RPCS@
 6565  041002  020043  042440  051122
 6566  041010  050040  020103  042040
 6567  041016  053122  021440  020040
 6568  041024  051040  042120  020123
 6569  041032  020040  051040  042520
 6570  041040  020122  020040  051040
 6571  041046  041520  000123
 6572  041052  020040  020040  020040  DH16:   .ASCII  @                          EXPT'D  ACTUAL@<15><12>
 6573  041060  020040  020040  020040
 6574  041066  020040  020040  020040
 6575  041074  020040  020040  020040
 6576  041102  054105  052120  042047
 6577  041110  020040  041501  052524
 6578  041116  046101  005015
 6579  041122  042524  052123  021440          .ASCIZ  @TEST #  ERR PC  DRV #   SUCA    SUCA@
 6580  041130  020040  051105  020122
 6581  041136  041520  020040  051104
 6582  041144  020126  020043  020040
 6583  041152  052523  040503  020040
 6584  041160  020040  052523  040503
 6585  041166     000
 6586  041167     124  051505  020124  DH25:   .ASCIZ  @TEST #  ERR PC  DRV #   EXPT'D  RECV'D@
 6587  041174  020043  042440  051122
 6588  041202  050040  020103  042040
 6589  041210  053122  021440  020040
 6590  041216  042440  050130  023524
 6591  041224  020104  051040  041505
 6592  041232  023526  000104
 6593  041236  042510  042101  051105  DH25A:  .ASCIZ  @HEADER CONTENTS@
 6594  041244  041440  047117  042524
 6595  041252  052116  000123
 6596  041256  020040  020040  020040  DH43:   .ASCII  @                          EXPT'D  ACTUAL@<15><12>
 6597  041264  020040  020040  020040
 6598  041272  020040  020040  020040
 6599  041300  020040  020040  020040
 6600  041306  054105  052120  042047
 6601  041314  020040  041501  052524
 6602  041322  046101  005015
 6603  041326  042524  052123  021440          .ASCIZ  @TEST #  ERR PC  DRV #   SECTOR  SECTOR@
 6604  041334  020040  051105  020122
 6605  041342  041520  020040  051104
 6606  041350  020126  020043  020040
 6607  041356  042523  052103  051117
 6608  041364  020040  042523  052103
 6609  041372  051117     000
 6610  041375     122  042120  020123  DH43A:  .ASCIZ  @RPDS    RPER    RPCS    RPWC    RPBA    RPCA    RPDA@
 6611  041402  020040  051040  042520
 6612  041410  020122  020040  051040
 6613  041416  041520  020123  020040
 6614  041424  051040  053520  020103
```

```
 6615  041432  020040  051040  041120
 6616  041440  020101  020040  051040
 6617  041446  041520  020101  020040
 6618  041454  051040  042120  000101
 6619  041462  042510  042101  051105  DH43B:  .ASCIZ  @HEADER CONTENTS@
 6620  041470  041440  047117  042524
 6621  041476  052116  000123
 6622  041502  020040  020040  020040  DH45:   .ASCII  @                              EXPT'D  'SOT'   SECTOR@<15><12>
 6623  041510  020040  020040  020040
 6624  041516  020040  020040  020040
 6625  041524  020040  020040  020040
 6626  041532  054105  052120  042047
 6627  041540  020040  051447  052117
 6628  041546  020047  020040  042523
 6629  041554  052103  051117  005015
 6630  041562  042524  052123  021440          .ASCIZ  @TEST #  ERR PC  DRV #   VALUE  COUNTER  ADDR@
 6631  041570  020040  051105  020122
 6632  041576  041520  020040  051104
 6633  041604  020126  020043  020040
 6634  041612  040526  052514  020105
 6635  041620  041440  052517  052116
 6636  041626  051105  020040  042101
 6637  041634  051104     000
 6638  041637     122  042120  020123  DH45A:  .ASCIZ  @RPDS    RPER    RPCS    RPDA@
 6639  041644  020040  051040  042520
 6640  041652  020122  020040  051040
 6641  041660  041520  020123  020040
 6642  041666  051040  042120  000101
 6643  041674  020040  020040  020040  DH46:   .ASCII  @                                      EXPT'D  RECV'D@<15><12>
 6644  041702  020040  020040  020040
 6645  041710  020040  020040  020040
 6646  041716  020040  020040  020040
 6647  041724  020040  020040  020040
 6648  041732  020040  054105  052120
 6649  041740  042047  020040  042522
 6650  041746  053103  042047  005015
 6651  041754  042524  052123  021440          .ASCIZ  @TEST #  ERR PC  DRV #   SECTOR  DATA    DATA@
 6652  041762  020040  051105  020122
 6653  041770  041520  020040  051104
 6654  041776  020126  020043  020040
 6655  042004  042523  052103  051117
 6656  042012  020040  040504  040524
 6657  042020  020040  020040  040504
 6658  042026  040524     000
 6659  042031     124  051505  020124  DH47:   .ASCIZ  @TEST #  ERR PC  DRV #   EXPT'D  RECV'D@
 6660  042036  020043  042440  051122
 6661  042044  050040  020103  042040
 6662  042052  053122  021440  020040
 6663  042060  042440  050130  023524
 6664  042066  020104  051040  041505
 6665  042074  023526  000104
 6666  042100  050122  051504  020040  DH47A:  .ASCIZ  @RPDS    RPER    RPCS    RPCA    RPDA@
 6667  042106  020040  050122  051105
 6668  042114  020040  020040  050122
 6669  042122  051503  020040  020040
 6670  042130  050122  040503  020040
```

```
6671  042136  020040  050122  040504
6672  042144     000
6673  042145     124  051505  020124  DH51:   .ASCIZ  @TEST #  ERR PC  DRV #  RPDS    RPER    RPCS    RPWC    RPBA@
6674  042152  020043  042440  051122
6675  042160  050040  020103  042040
6676  042166  053122  021440  020040
6677  042174  051040  042120  020123
6678  042202  020040  051040  042520
6679  042210  020122  020040  051040
6680  042216  041520  020123  020040
6681  042224  051040  053520  020103
6682  042232  020040  051040  041120
6683  042240  000101
6684  042242  042524  052123  021440  DH63:   .ASCIZ  @TEST #  ERR PC  DRV #  RPDS    RPER    RPCS    SUCA@
6685  042250  020040  051105  020122
6686  042256  041520  020040  051104
6687  042264  020126  020043  020040
6688  042272  050122  051504  020040
6689  042300  020040  050122  051105
6690  042306  020040  020040  050122
6691  042314  051503  020040  020040
6692  042322  052523  040503     000
6693  042327     124  051505  020124  DH71:   .ASCIZ  @TEST #  ERR PC  DRV #  RPDS    RPER    RPCS    RPCA@
6694  042334  020043  042440  051122
6695  042342  050040  020103  042040
6696  042350  053122  021440  020040
6697  042356  051040  042120  020123
6698  042364  020040  051040  042520
6699  042372  020122  020040  051040
6700  042400  041520  020123  020040
6701  042406  051040  041520  000101
6702  042414  042524  052123  021440  DH102:  .ASCIZ  @TEST #  ERR PC  DRV #  SEC STRT CYL     TRK     SEC     PATTERN@
6703  042422  020040  051105  020122
6704  042430  041520  020040  051104
6705  042436  020126  020043  051440
6706  042444  041505  051440  051124
6707  042452  020124  054503  020114
6708  042460  020040  020040  051124
6709  042466  020113  020040  020040
6710  042474  042523  020103  020040
6711  042502  020040  040520  052124
6712  042510  051105  000116
6713  042514  020040  020040  020040  DH102A: .ASCII  @         GOOD    BAD@<15><12>
6714  042522  020040  047507  042117
6715  042530  020040  020040  040502
6716  042536  006504     012
6717  042541     101  042104  020122          .ASCIZ  @ADDR    DATA    DATA@
6718  042546  020040  042040  052101
6719  042554  020101  020040  042040
6720  042562  052101  000101
6721  042566  044513  040520  031522  DH103:  .ASCIZ  @KIPAR3  KIPAR4  KIPAR5  KIPDR3  KIPDR4  KIPDR5@
6722  042574  020040  044513  040520
6723  042602  032122  020040  044513
6724  042610  040520  032522  020040
6725  042616  044513  042120  031522
6726  042624  020040  044513  042120
```

```
 6727  042632  032122  020040  044513
 6728  042640  042120  032522     000
 6729  042645     126  051111  052524  DH103A: .ASCII  @VIRTUAL GOOD    BAD@<15><12>
 6730  042652  046101  043440  047517
 6731  042660  020104  020040  041040
 6732  042666  042101  005015
 6733  042672  042101  051104  020040          .ASCIZ  @ADDR    DATA    DATA@
 6734  042700  020040  040504  040524
 6735  042706  020040  020040  040504
 6736  042714  040524     000
 6737  042717     124  052117  046101  DH105:  .ASCIZ  @TOTAL COMPARE ERRORS:@
 6738  042724  041440  046517  040520
 6739  042732  042522  042440  051122
 6740  042740  051117  035123     000
 6741  042745     125  051516  042503  DH110:  .ASCIZ  @UNSCESSFUL AFTER 3 RETRIES@
 6742  042752  051523  052506  020114
 6743  042760  043101  042524  020122
 6744  042766  020063  042522  051124
 6745  042774  042511  000123
 6746  043000  042522  051124  020131  DH111:  .ASCII  @RETRY SUCESSFUL@<15><12>
 6747  043006  052523  042503  051523
 6748  043014  052506  006514     012
 6749  043021     122  052105  054522          .ASCIZ  @RETRY COUNT:@
 6750  043026  041440  052517  052116
 6751  043034  000072
 6752
 6753                                  .EVEN
 6754
 6755                                  ;'DT' WORDS
 6756
 6757  043036  001324                  DT1:    .WORD   RPADR
 6758  043040  001160  001116  001176  DT2:    .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$RPWC
 6759  043046  001334  001336  001340
 6760  043054  001342
 6761  043056  001344  001346  001350          .WORD   $RPBA,$RPCA,$RPDA,$RPM1,$SUCA,$SILO
 6762  043064  001352  001354  001356
 6763  043072  001160  001116  001176  DT5:    .WORD   $TMPO,$ERRPC,CHKDRV,PATNUM,$RPDS,$RPER,$RPCS
 6764  043100  001204  001334  001336
 6765  043106  001340
 6766  043110  001342  001344  001346          .WORD   $RPWC,$RPBA,$RPCA,$RPDA,$RPM1,$SUCA,$SILO
 6767  043116  001350  001352  001354
 6768  043124  001356
 6769  043126  001160  001116  001176  DT11:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,BUFFER,BUFFER+2
 6770  043134  001124  043656  043660
 6771  043142  001160  001116  001176  DT12:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,$BDDAT,$BDADR
 6772  043150  001124  001126  001122
 6773  043156  001160  001116  001176  DT14:   .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS
 6774  043164  001334  001336  001340
 6775  043172  001160  001116  001176  DT16:   .WORD   $TMPO,$ERRPC,CHKDRV,MAXCYL,$SUCA
 6776  043200  001210  001354
 6777  043204  001160  001116  001176  DT25:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,$BDDAT
 6778  043212  001124  001126
 6779  043216  043656  043660  043662          .WORD   BUFFER,BUFFER+2,BUFFER+4
 6780  043224  001160  001116  001176  DT43:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,BUFFER+4
 6781  043232  001124  043662
 6782  043236  001334  001336  001340          .WORD   $RPDS,$RPER,$RPCS,$RPWC,$RPBA,$RPCA,$RPDA
```

```
 6783  043244  001342  001344  001346
 6784  043252  001350
 6785  043254  043656  043660  043662                    .WORD   BUFFER,BUFFER+2,BUFFER+4
 6786  043262  001160  001116  001176  DT45:             .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,$BDDAT,$BDADR
 6787  043270  001124  001126  001122
 6788  043276  001334  001336  001340                    .WORD   $RPDS,$RPER,$RPCS,$RPDA
 6789  043304  001350
 6790  043306  001160  001116  001176  DT46:             .WORD   $TMPO,$ERRPC,CHKDRV,$GDADR,$GDDAT,$BDDAT
 6791  043314  001120  001124  001126
 6792  043322  001160  001116  001176  DT47:             .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,BUFFER
 6793  043330  001124  043656
 6794  043334  001334  001336  001340                    .WORD   $RPDS,$RPER,$RPCS,$RPCA,$RPDA
 6795  043342  001346  001350
 6796  043346  001160  001116  001176  DT51:             .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$RPWC,$RPBA
 6797  043354  001334  001336  001340
 6798  043362  001342  001344
 6799  043366  001160  001116  001176  DT63:             .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$SUCA
 6800  043374  001334  001336  001340
 6801  043402  001354
 6802  043404  001160  001116  001176  DT71:             .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$RPCA
 6803  043412  001334  001336  001340
 6804  043420  001346
 6805  043422  001160  001116  001176  DT102:            .WORD   $TMPO,$ERRPC,CHKDRV,CMBUF,CMCYL,CMTRK,CMSEC,PATNUM
 6806  043430  025430  025422  025426
 6807  043436  025424  001204
 6808  043442  001122  001124  001126                    .WORD   $BDADR,$GDDAT,$BDDAT
 6809  043450  001160  001116  001176  DT103:            .WORD   $TMPO,$ERRPC,CHKDRV,CMBUF,CMCYL,CMTRK,CMSEC,PATNUM
 6810  043456  025430  025422  025426
 6811  043464  025424  001204
 6812  043470  172346  172350  172352                    .WORD   KIPAR3,KIPAR4,KIPAR5,KIPDR3,KIPDR4,KIPDR5
 6813  043476  172306  172310  172312
 6814  043504  001122  001124  001126  DT104:            .WORD   $BDADR,$GDDAT,$BDDAT
 6815  043512  025414            DT105:            .WORD   ERCTR
 6816  043514  001314            DT111:            .WORD   RETRY
 6817
 6818                            ;'DF' BLOCKS
 6819
 6820  043516  000001            DF1:              .WORD   1
 6821  043520     001                                .BYTE   1
 6822  043521     000                                .BYTE   0
 6823
 6824  043522  000002            DF2:              .WORD   2
 6825  043524     007                                .BYTE   7
 6826  043525     000                                .BYTE   0
 6827  043526  040410                               .WORD   DH2A
 6828  043530     006                                .BYTE   6
 6829  043531     000                                .BYTE   0
 6830
 6831  043532  000002            DF5:              .WORD   2
 6832  043534     007                                .BYTE   7
 6833  043535     010                                .BYTE   10
 6834  043536  040552                               .WORD   DH5A
 6835  043540     007                                .BYTE   7
 6836  043541     000                                .BYTE   0
 6837
 6838  043542  000001            DF11:             .WORD   1
```

```
                                        .BYTE   6
                                        .BYTE   0

DF25:   .WORD   2
                .BYTE   5
                .BYTE   0
                .WORD   DH25A
                .BYTE   3
                .BYTE   0

DF43:   .WORD   3
                .BYTE   6
                .BYTE   0
                .WORD   DH43A
                .BYTE   5
                .BYTE   0
                .WORD   DH43B
                .BYTE   3
                .BYTE   0

DF45:   .WORD   2
                .BYTE   6
                .BYTE   0
                .WORD   DH45A
                .BYTE   4
                .BYTE   0

DF47:   .WORD   2
                .BYTE   5
                .BYTE   0
                .WORD   DH47A
                .BYTE   5
                .BYTE   0

DF51:   .WORD   1
                .BYTE   8.
                .BYTE   0

DF53:   .WORD   1
                .BYTE   5
                .BYTE   0

DF63:   .WORD   1
                .BYTE   7
                .BYTE   0

DF102:  .WORD   2
                .BYTE   8.
                .BYTE   200
                .WORD   DH102A
                .BYTE   0
                .BYTE   0

DF103:  .WORD   3
                .BYTE   8.
                .BYTE   200
```

```
 6895  043642  042566                        .WORD   DH103
 6896  043644     006                        .BYTE   6
 6897  043645     000                        .BYTE   0
 6898  043646  042645                        .WORD   DH103A
 6899  043650     000                        .BYTE   0
 6900  043651     000                        .BYTE   0
 6901
 6902  043652  000001              DF104:    .WORD   1
 6903  043654     003                        .BYTE   3
 6904  043655     000                        .BYTE   0
 6905
 6906
 6907                                .EVEN
 6908
 6909          043656                BUFFER=.                ;BUFFER STARTS HERE
 6910
 6911  043656  005015  046412  044501  TITLE:  .ASCII  <15><12><12>@MAINDEC-11-DZRPY-C@
 6912  043664  042116  041505  030455
 6913  043672  026461  055104  050122
 6914  043700  026531     103
 6915  043703     015  051012  030520          .ASCIZ  <15><12>@RP11-E FUNCTIONAL LOGIC & READ/WRITE TEST@<15><12><12>
 6916  043710  026461  020105  052506
 6917  043716  041516  044524  047117
 6918  043724  046101  046040  043517
 6919  043732  041511  023040  051040
 6920  043740  040505  027504  051127
 6921  043746  052111  020105  042524
 6922  043754  052123  005015  000012
 6923          000001                        .END
```

```
ACTMEM  001224
AIE     = 020000
ATABIT  001360
BIT5    001400
BIT0    = 000001
BIT00   = 000001
BIT01   = 000002
BIT02   = 000004
BIT03   = 000010
BIT04   = 000020
BIT05   = 000040
BIT06   = 000100
BIT07   = 000200
BIT08   = 000400
BIT09   = 001000
BIT1    = 000002
BIT10   = 002000
BIT11   = 004000
BIT12   = 010000
BIT13   = 020000
BIT14   = 040000
BIT15   = 100000
BIT2    = 000004
BIT3    = 000010
BIT4    = 000020
BIT5    = 000040
BIT6    = 000100
BIT7    = 000200
BIT8    = 000400
BIT9    = 001000
BLKS14  001254
BLNKS2  027430
BPTVEC= 000014
BUFFER= 043656
BUSADR  001216
BYPASS  001200
CHKDRV  001176
CHNGAD  031640
CHNGPM  031406
CHNGPT  031565
CIR     023120
CKSWR   = 104407
CK.CHR  027122
CK.NUM  027136
CK.OCT  027072
CLEAR   = 000001
CLRP    025654
CMBUF   025430
CMCNT   025420
CMCYL   025422
CMDAT   025250
CMPAR   025130
CMPRT   025432
CMPRT1  025536
CMPRX   025374
CMP.CT  001320
CMSEC   025434
CMSTA   025166
CMTRK   025426
CNTRLC  001206
COMMA   027377
CONRDY  023664
CR      = 000015
CRLF    = 000200
DATCMP  024614
DDISP   = 177570
DF1     023516
DF10    023636
DF11    023656
DF12    023652
DF2     023542
DF25    023546
DF45    023556
DF46    023572
DF47    023602
DF5     023532
DF51    023612
DF53    023616
DF63    023622
DH1     040315
DH102   042414
DH102A  042514
DH103   042566
DH103A  042646
DH105   042717
DH11    040637
DH110   042745
DH111   043000
DH12    040720
DH14    040775
DH16    041052
DH2     040323
DH2A    040410
DH25    041167
DH25A   041236
DH43    041256
DH43A   041375
DH43B   041462
DH45    041502
DH45A   041637
DH46    041674
DH47    042031
DH47A   042100
DH5     040465
DH5A    040552
DH51    042145
DH63    042244
DH71    042327
DISPLA  001142
DISPRE  001174
DP8     002500
DRIVE   031377
DRSTAT  027236
DRVBAD  001310
DRVINT  022666
DRVMSK  001306
DRVNM   031352
DRVOK   005041
DRVRDY  023546
DRVSEL  001316
DRVSTA  001370
DRVTST  027345
DRVTYP  001304
DSWR    = 177570
DT1     043036
DT102   043226
DT103   043450
DT104   043504
DT105   043512
DT11    043126
DT111   043514
DT12    043142
DT15    043156
DT16    043172
DT2     043040
DT25    043204
DT45    043220
DT46    043306
DT47    043322
DT5     043072
DT51    043346
DT63    043366
DT71    043404
EMTVEC= 000030
EM1     032173
EM100   037715
EM101   037735
EM102   037762
EM103   040005
EM106   040064
EM107   040120
EM11    032570
EM112   040154
EM113   040212
EM12    032663
EM13    033014
EM14    033137
EM15    033173
EM16    033223
EM17    033260
EM2     033305
EM20    033312
EM21    033342
EM22    033355
EM23    033372
EM24    033416
EM25    033716
EM26    033766
EM27    034056
EM3     034066
EM30    034107
EM31    034142
EM32    034172
EM33    034252
EM34    034367
EM35    034457
EM36    034561
EM37    034642
EM4     032357
EM40    034676
EM41    034733
EM42    034764
EM43    035042
EM44    035111
EM45    035216
EM46    035245
EM47    035316
EM5     032414
EM50    035445
EM51    035524
EM52    035620
EM53    035704
EM54    036013
EM55    036072
EM56    036162
EM57    036255
EM6     032466
EM60    036371
EM61    036465
EM62    036522
EM63    036557
EM64    036673
EM65    037000
EM66    037037
EM67    037101
EM7     032522
EM70    037156
EM71    037230
EM72    037344
EM73    037415
EM74    037467
EM75    037553
EM76    037646
EM77    037675
ENDCMP  025630
ENTCYL  032054
ENTNM   031324
ENTPRM  026314
ENTPRX  027070
ENTPR1  026374
ENTPR2  026572
ENTPR3  026670
ENTPR4  026716
ENTRK   032106
ENTSEC  032133
ENTWC   031523
ERCTR   025414
ERR     = 100000
ERRVEC= 000004
EXIT0   005306
EXIT1   005722
EXIT10  011530
EXIT11  012060
EXIT12  012712
EXIT13  013572
EXIT14  014616
EXIT15  015302
EXIT16  015620
EXIT17  016604
EXIT2   006400
EXIT3   007056
EXIT4   007306
EXIT5   010032
EXIT6   010366
EXIT7   010610
FILSEC  024336
FILTRK  024376
FLAG    001202
FRSTER  025412
GETADR  026150
GETBUF  024502
GTSWR   = 104406
HDR     = 004000
HE      = 040000
HIMEM   001234
HOMSEK= 000015
HT      = 000011
ICNT    001214
IDE     = 000100
INCADR  024114
INCSEC  001250
INCTRK  001252
IOTVEC= 000020
```

```
KIPAR0= 172340          OFFLIN  027312          RPCS   = 000004          SUSI   = 004000          TSTING  027401
KIPAR1= 172342          ONEPAT  031713          RPDA   = 000014          SUWP   = 000400          TSTNMS  001276
KIPAR2= 172344          ONLINE  027262          RPDS   = 000000          SWR      001140          TST0    005064
KIPAR3= 172346          OPRSEL  001260          RPER   = 000002          SWREG    000176          TST1    005310
KIPAR4= 172350          PAKSIZ  001244          RPINIT   022566          SW0    = 000001          TST10   010612
KIPAR5= 172352          PATNUM  001204          RPM1   = 000010          SW00   = 000001          TST11   011532
KIPAR6= 172354          PATSEL  024050          RPM2   = 000020          SW01   = 000002          TST12   012062
KIPAR7= 172356          PATTRN  001300          RPM3   = 000022          SW02   = 000004          TST12B  012534
KIPDR0= 172300          PAT.PT  001440          RPPRIO   001332          SW03   = 000010          TST13   012720
KIPDR1= 172302          PAT0    001500          RPVEC    001324          SW04   = 000020          TST13A  013474
KIPDR2= 172304          PAT1    001540          RPWC   = 000006          SW05   = 000040          TST14   013574
KIPDR3= 172306          PAT10   002200          RPO2     027327          SW06   = 000100          TST14A  014342
KIPDR4= 172310          PAT11   002240          RPO3     027336          SW07   = 000200          TST15   014630
KIPDR5= 172312          PAT12   002300          RP11     022770          SW08   = 000400          TST15A  015112
KIPDR6= 172314          PAT13   002340          RSTRT1   004740          SW09   = 001000          TST15B  015314
KIPDR7= 172316          PAT14   002400          RSTRT2   004752          SW1    = 000002          TST15C  015324
LDRFLG  001322          PAT15   002440          RSTRT3   004760          SW10   = 002000          TST16   015372
LF     = 000012          PAT2    001600          R6       =%000006          SW11   = 004000          TST17   015624
LIMIT   025416          PAT3    001640          R7       =%000007          SW12   = 010000          TST2    005724
LMTBK   001262          PAT4    001700          SAVREG=  104412          SW13   = 020000          TST3    006402
LODSEC  024304          PAT5    001740          SAVRP    023240          SW14   = 040000          TST4    007060
LOMEM   001240          PAT6    002000          SCYL     001266          SW15   = 100000          TST5    007310
LRETRY  001316          PAT7    002040          SECNBR   024002          SW2    = 000004          TST5A   007634
MAXCYL  001210          PAT8    002100          SEEK   = 000011          SW3    = 000010          TST5B   007742
MAXMEM  001230          PAT9    002140          SETBUF   024432          SW4    = 000020          TST6    010034
MAXWC   001264          PIRQ   = 177772          SILO   = 000026          SW5    = 000040          TST7    010370
MEMSIZ  001302          PIRQVE= 000240          SIZMEM   025700          SW6    = 000100          TYPOS  = 104405
MEX0   = 000020          PR0    = 000000          SLASH    031352          SW7    = 000200          TYPE   = 104401
MEX1   = 000040          PR1    = 000040          SR0    = 177572          SW8    = 000400          TYPERR  017254
MMACTV  001222          PR2    = 000100          SR1    = 177574          SW9    = 001000          TYPOC  = 104402
MMVEC  = 000250          PR3    = 000140          SR2    = 177576          TBITVE= 000014          TYPON  = 104404
MODE   = 010000          PR4    = 000200          SR3    = 172516          TESTO   005160          TYPOS  = 104403
MOVLDR  023346          PR5    = 000240          SSEC     001272          TEST1   005404          UNSAFE  027276
MRPADR  032162          PR6    = 000300          STACK  = 001100          TEST10  010706          WC14    001256
MSG15A  027433          PR7    = 000340          STALL    023202          TEST11  011652          WRITE  = 000013
MSG15B  027456          PS     = 177776          STALLT   001220          TEST12  012202          WRSW    016206
MSG15C  027617          PSW    = 177776          START    003710          TEST13  013014          WRSW1   016270
MSG15D  027652          PWRVEC= 000024          START1   003654          TEST14  014066          WRSW2   016412
MSG16A  027743          RDCHR  = 104410          START2   003676          TEST15  014736          WRTCHK= 000007
MSG16B  030255          RDERR    001274          START3   003644          TEST16  015460          WRTSEK= 000003
MSG17A  030317          RDLIN  = 104411          START4   003666          TEST17  015716          $AUTOB  001134
MSG17B  030362          RDSEK  = 000005          START5   004146          TEST2   006020          $BDADR  001122
MSG17C  030441          RDSW     016056          START6   004360          TEST3   006476          $BDDAT  001126
MSG17D  030464          RDY    = 000200          START7   004544          TEST4   007154          $BELL   001166
MSG17E  030506          READ   = 000017          STKLMT= 177774          TEST5   007404          $BUF    002506
MSG17F  030542          RESLDR   023452          STRK     001270          TEST6   010130          $CHARC  017720
MSG17G  030577          RESREG=  104413          STRT1A   003660          TEST7   010464          $CKSWR  020704
MSG17H  030714          RESTAR   004762          STRT2A   003702          TIMOUT  001212          $CMTAG  001100
MSG17I  031014          RESVEC=  000010          SUCA   = 000024          TITLE   043656          $CM3   = 000000
MSG17J  031037          RETRY    001314          SUFU   = 001000          TKVEC  = 000060          $CM4   = 000001
MSG17K  031176          RPADR    001324          SUOL   = 040000          TPVEC  = 000064          $CNTLC  021604
MXWRDS  031447          RPBA   = 000010          SURDY  = 100000          TRAPVE= 000034          $CNTLG  021616
NONE    027372          RPCA   = 000012          SURP03= 020000          TRTVEC= 000014          $CNTLU  021611
```

```
$CORE   022402        $GDDAT  001124        $OVER   022062        $SIZEX  022436        $TRAP   022502
$CRLF   001173        $GET42  017040        $PASS   001100        $STUP = 177777        $TRAP2  022524
$CROUT  022432        $GTSWR  020774        $QUES   001172        $SUCA   001354        $TRK    002513
$CYL    002510        $HD   = 000000        $RDCHR  021246        $SVLAD  022034        $TRP  = 000014
$DBLK   020366        $ICNT   001104        $RDLIN  021336        $SVPC = 000200        $TRPAD  022536
$DOAGN  017060        $INTAG  001135        $RDSZ = 000012        $SWR  = 167000        $TSTNM  001102
$DTBL   020356        $ITEMB  001114        $RESRE  022136        $SWRMK= 000000        $TTYIN  021572
$ENDAD  017050        $KTNEX  022374        $RPBA   001344        $TIMES  001162        $TYPDS  020152
$ENDCT  017004        $KTOUT  022364        $RPCA   001346        $TKB    001146        $TYPE   017504
$ENULL  017064        $KTI1   022232        $RPCS   001340        $TKCNT  020376        $TYPEC  017654
$EOP    016616        $LF     001174        $RPDA   001350        $TKINT  020416        $TYPEX  017722
$EOPCT  016776        $LPADR  001106        $RPOS   001334        $TKQEN= 020416        $TYPOC  017750
$ERFLG  001103        $LPERR  001110        $RPER   001336        $TKQIN  020400        $TYPON  017764
$ERMAX  001115        $LSTAD  022476        $RPM1   001352        $TKQOU  020402        $TYPOS  017724
$ERROR  017070        $LSTBK  022500        $RPWC   001342        $TKQSR  020404        $WC     002504
$ERRPC  001116        $MNEW   021634        $RTNAD  017062        $TKS    001144        $XTSTR  021660
$ERRTB  002514        $MSWR   021623        $SAVRE  022100        $TKSRV  020466        $$GET4= 000000
$ERTTL  001112        $MXCNT  022076        $SCOPE  021646        $TMPO   001160        $OFILL  020147
$ESCAP  001164        $NULL   001154        $SEC    002512        $TN   = 000020        .     = 043762
$FILLC  001156        $NWTST= 000001        $SETUP= 000127        $TPB    001152
$FILLS  001155        $OCNT   020146        $SILO   001356        $TPFLG  001157
$GDADR  001120        $OMODE  020150        $SIZE   022174        $TPS    001150

. ABS.   043762      000

 ERRORS DETECTED:  0

 DSKW:DZRPYC,DSKW:DZRPYC/SOL=DSKW:SYSMAC.SML,DSKM:DZRPYC.P11
 RUN-TIME:  16 24  .6 SECONDS
 RUN-TIME RATIO: 371/41=8.9
 CORE USED:  37K  (73 PAGES)
```