RP11E

FUNCTIONAL LOGIC & READ/
CZRPYD0
WRITE TEST

AH-9303D-MC

FICHE 1 OF 1

JUL 1978
digital
MADE IN USA

.REM %

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

PRODUCT CODE: AC-9302D-MC

PRODUCT NAME: CZRPYDO RP11E FUNCTIONAL LOGIC & READ/WRITE TEST

DATE: 1-MAY-78

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 PPOGRAM 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. PPESS 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 RP11F 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 TEST 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
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  CZRPY-D, RP11-E FUNCTIONAL LOGIC & R/W TEST
                                ;*COPYRIGHT (C) 1975,1978
                                ;*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
```

```
 1035            002000                  SW10=   2000
 1036            001000                  SW09=   1000
 1037            000400                  SW08=   400
 1038            000200                  SW07=   200
 1039            000100                  SW06=   100
 1040            000040                  SW05=   40
 1041            000020                  SW04=   20
 1042            000010                  SW03=   10
 1043            000004                  SW02=   4
 1044            000002                  SW01=   2
 1045            000001                  SW00=   1
 1046                                    .EQUIV  SW09,SW9
 1047                                    .EQUIV  SW08,SW8
 1048                                    .EQUIV  SW07,SW7
 1049                                    .EQUIV  SW06,SW6
 1050                                    .EQUIV  SW05,SW5
 1051                                    .EQUIV  SW04,SW4
 1052                                    .EQUIV  SW03,SW3
 1053                                    .EQUIV  SW02,SW2
 1054                                    .EQUIV  SW01,SW1
 1055                                    .EQUIV  SW00,SW0
 1056
 1057                                    ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
 1058            100000                  BIT15=  100000
 1059            040000                  BIT14=  40000
 1060            020000                  BIT13=  20000
 1061            010000                  BIT12=  10000
 1062            004000                  BIT11=  4000
 1063            002000                  BIT10=  2000
 1064            001000                  BIT09=  1000
 1065            000400                  BIT08=  400
 1066            000200                  BIT07=  200
 1067            000100                  BIT06=  100
 1068            000040                  BIT05=  40
 1069            000020                  BIT04=  20
 1070            000010                  BIT03=  10
 1071            000004                  BIT02=  4
 1072            000002                  BIT01=  2
 1073            000001                  BIT00=  1
 1074                                    .EQUIV  BIT09,BIT9
 1075                                    .EQUIV  BIT08,BIT8
 1076                                    .EQUIV  BIT07,BIT7
 1077                                    .EQUIV  BIT06,BIT6
 1078                                    .EQUIV  BIT05,BIT5
 1079                                    .EQUIV  BIT04,BIT4
 1080                                    .EQUIV  BIT03,BIT3
 1081                                    .EQUIV  BIT02,BIT2
 1082                                    .EQUIV  BIT01,BIT1
 1083                                    .EQUIV  BIT00,BIT0
 1084
 1085                                    ;*BASIC "CPU" TRAP VECTOR ADDRESSES
 1086            000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
 1087            000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
 1088            000014                  TBITVEC=14              ;;"T" BIT
 1089            000014                  TRTVEC= 14              ;;TRACE TRAP
 1090            000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
```

```
1091    000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
1092    000024          PWRVEC= 24              ;;POWER FAIL
1093    000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
1094    000034          TRAPVEC=34              ;;"TRAP" TRAP
1095    000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
1096    000064          TPVEC=  64              ;;TTY PRINTER VECTOR
1097    000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
1098                    .SBTTL  MEMORY MANAGEMENT DEFINITIONS
1099
1100                    ;*KT11 VECTOR ADDRESS
1101
1102    000250          MMVEC=  250
1103
1104                    ;*KT11 STATUS REGISTER ADDRESSES
1105
1106    177572          SR0=    177572
1107    177574          SR1=    177574
1108    177576          SR2=    177576
1109    172516          SR3=    172516
1110
1111                    ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
1112
1113    172300          KIPDR0= 172300
1114    172302          KIPDR1= 172302
1115    172304          KIPDR2= 172304
1116    172306          KIPDR3= 172306
1117    172310          KIPDR4= 172310
1118    172312          KIPDR5= 172312
1119    172314          KIPDR6= 172314
1120    172316          KIPDR7= 172316
1121
1122                    ;*KERNEL "I" PAGE ADDRESS REGISTERS
1123
1124    172340          KIPAR0= 172340
1125    172342          KIPAR1= 172342
1126    172344          KIPAR2= 172344
1127    172346          KIPAR3= 172346
1128    172350          KIPAR4= 172350
1129    172352          KIPAR5= 172352
1130    172354          KIPAR6= 172354
1131    172356          KIPAR7= 172356
1132
1133
1134                    ;RP11E REGISTER INDEX EQUATES
1135
1136    000000          RPDS=00
1137    000002          RPER=02
1138    000004          RPCS=04
1139    000006          RPWC=06
1140    000010          RPBA=10
1141    000012          RPCA=12
1142    000014          RPDA=14
1143    000016          RPM1=16
1144    000020          RPM2=20
1145    000022          RPM3=22
1146    000024          SUCA=24
```

```
1147          000026                  SILO=26
1148
1149                                  ;RP11 OP CODE DEFINITIONS
1150
1151          000001                  CLEAR=1                         ;CLEAR THE CONTROLLER
1152          000003                  WRTSEK=3                        ;WRITE WITH IMPLIED SEEK AND HEAD SELECTION
1153          000005                  RDSEK=5                         ;READ WITH IMPLIED SEEK AND HEAD SELECT
1154          000007                  WRTCHK=7                        ;WRITE CHECK WITH IMPLIIED SFEK AND HEAD SELECT
1155          000011                  SEEK=11                         ;SEEK
1156          000013                  WRITE=13                        ;WRITE (NO IMPLIED SEEK OR HEAD SELECT)
1157          000015                  HOMSEK=15                       ;HOME SEEK
1158          000017                  READ=17                         ;READ (NO IMPLIED SEEK OR HEAD SELECT)
1159
1160
1161                                  ;RPDS REGISTER BIT DEFINITIONS
1162
1163          000400                  SUWP=400                        ;SELECTED UNIT WRITE PROTECTED
1164          001000                  SUFU=1000                       ;SELECTED UNIT FILE UNSAFE
1165          004000                  SUSI=4000                       ;SELECTED UNIT SEEK INCOMPLETE
1166          020000                  SURP03=20000                    ;SELECTED UNIT IS AN RP03
1167          040000                  SUOL=40000                      ;SELECTED UNIT IS ONLINE
1168          100000                  SURDY=100000                    ;SELECTED UNIT IS READY
1169
1170                                  ;RPCS REGISTER BIT DEFINITIONS
1171
1172          000020                  MEX0=20                         ;EXTENDED MEMORY BIT #1
1173          000040                  MEX1=40                         ;EXTENDED MEMORY BIT #2
1174          000100                  IDE=100                         ;INTERRUPT ON DONE ENABLE BIT
1175          000200                  RDY=200                         ;RP11 CONTROLLER READY BIT
1176          004000                  HDR=4000                        ;HEADER OPERATION BIT
1177          010000                  MODE=10000                      ;MODE BIT
1178          020000                  AIE=20000                       ;ATTENTION INTERRUPT ENABLE BIT
1179          040000                  HE=40000                        ;HARD ERROR BIT
1180          100000                  ERR=100000                      ;ERROR BIT
1181
1182
1183
```

```
                                .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          $TMP0:  .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
```

```
1240  001224  000000  000000          ACTMEM: .WORD   0,0             ;PHYSICAL ADDRESS OF NEXT READ BUFFER
1241  001230  000000  000000          MAXMEM: .WORD   0,0             ;ADDRESS OF LAST MEMORY BANK (FILLED BY $SIZE)
1242  001234  000000  000000          HIMEM:  .WORD   0,0             ;PHYSICAL ADDRESS OF LAST MEMORY BANK (FOR TEST 14)
1243  001240  000000  000000          LOMEM:  .WORD   0,0             ;PHYSICAL ADDRESS OF FIRST BUFFER LOCATION
1244  001244  000000  000000          PAKSIZ: .WORD   0,0             ;SECTORS ON THE PACK ON THE DRIVE BEING TESTED
1245  001250  000000                  INCSEC: .WORD   0               ;SECTOR INCREMENT VALUE
1246  001252  000000                  INCTRK: .WORD   0               ;TRACK INCREMENT VALUE
1247  001254  000000                  BLKS14: .WORD   0               ;NUMBER SECTORS IN TRANSFER IN TEST 14
1248  001256  000000                  WC14:   .WORD   0               ;WORD COUNT FOR TEST 14
1249  001260  000000                  OPRSEL: .WORD   0               ;OPERATOR SELECTED ADDRESS FLAG
1250  001262  020000                  LMT8K:  .WORD   8192.           ;LARGEST TRANSFER PERMITTED
1251  001264  000000                  MAXWC:  .WORD   0               ;MAXMUM WORD COUNT ALLOWED ON THIS SYSTEM
1252  001266  000000                  SCYL:   .WORD   0               ;OPERATOR SELECTED CYLINDER
1253  001270  000000                  STRK:   .WORD   0               ;OPERATOR SELECTED TRACK
1254  001272  000000                  SSEC:   .WORD   0               ;OPERATOR SELECTED SECTOR
1255  001274  000000                  RDERR:  .WORD   0               ;READ RETRY COUNTER
1256  001276  000000                  TSTNMS: .WORD   0               ;TEST SELECTION WORD.  BIT 0 = TEST 0, ETC.
1257  001300  000000                  PATTRN: .WORD   0               ;PATTERN SELECTION WORD FOR TEST 14
1258  001302  000000                  MEMSIZ: .WORD   0               ;ADDRESS OF HIGHEST NON-MEMORY MANAGEMENT
1259                                                                  ;MEMORY LOCATION
1260  001304  000000                  DRVTYP: .WORD   0               ;CONTAINS A BIT FOR EACH RP03 DRIVE
1261                                                                  ;ON THE SYSTEM.  BIT00 = DRIVE 0, ETC.
1262  001306  000000                  DRVMSK: .WORD   0               ;DRIVE TEST SELECTION MASK
1263  001310  000000                  DRVBAD: .WORD   0               ;CONTAINS A BIT FOR THE DRIVE IF THE DRIVE
1264                                                                  ;GOES OFFLINE OR BECOMES UNSAFE DURING TESTING.
1265                                                                  ;BIT00 = DRIVE 0, ETC.
1266  001312  000000                  DRVSEL: .WORD   0               ;CONTAINS A BIT FOR EACH DRIVE TO BE TESTED
1267  001314  000000                  RETRY:  .WORD   0               ;RETRY COUNTER FOR TEST 14
1268  001316  000003                  LRETRY: .WORD   3               ;RETRY LIMIT FOR TEST 14
1269  001320  000003                  CMP.CT: .WORD   3               ;NUMBER OF COMPARE ERROR TYPEOUTS
1270  001322  000000                  LDRFLG: .WORD   0               ;WHEN =0 LOADERS ARE AT TOP OF MEMORY
1271
1272                                  ;RP11 ADDRESS VALUES
1273
1274  001324  176710                  RPADR   .WORD   176710          ;RP11 BUS ADDRESS
1275  001326  000254  000256          RPVEC:  .WORD   254,256         ;RP11 VECTOR ADDRESS
1276  001332  000240                  RPPRIO: .WORD   <5*32.>         ;RP11 PRIORITY
1277
1278                                  ;SAVE THE RP11E REGISTERS HERE
1279
1280  001334  000000                  $RPDS:  .WORD   0               ;DRIVE STATUS REGISTER
1281  001336  000000                  $RPER:  .WORD   0               ;ERROR REGISTER
1282  001340  000000                  $RPCS:  .WORD   0               ;COMMAND & STATUS REGISTER
1283  001342  000000                  $RPWC:  .WORD   0               ;WORD COUNT REGISTER
1284  001344  000000                  $RPBA:  .WORD   0               ;BUFFER ADDRESS REGISTER
1285  001346  000000                  $RPCA:  .WORD   0               ;CURRENT CYLINDER ADDRESS REGISTER
1286  001350  000000                  $RPDA:  .WORD   0               ;TRACK-SECTOR ADDRESS REGISTER
1287  001352  000000                  $RPM1:  .WORD   0               ;MAINTENANCE REGISTER #1
1288  001354  000000                  $SUCA:  .WORD   0               ;SELECTED UNIT CYLINDER ADDRESS REGISTER
1289  001356  000000                  $SILO:  .WORD   0               ;SILO REGISTER
1290
1291                                  ;ATTENTION BITS
1292
1293  001340     001                  ATABIT: .BYTE   1               ;DRIVE 0
1294  001361     002                          .BYTE   2               ;DRIVE 1
1295  001362     004                          .BYTE   4               ;DRIVE 2
```

```
1296  001363  010                     .BYTE   10              ;DRIVE 3
1297  001364  020                     .BYTE   20              ;DRIVE 4
1298  001365  040                     .BYTE   40              ;DRIVE 5
1299  001366  100                     .BYTE   100             ;DRIVE 6
1300  001367  200                     .BYTE   200             ;DRIVE 7
1301
1302
1303                          ;DRIVE STATUS INDICATORS (DRVSTA=8 BYTES)
1304                                  ;DRVSTA=0 DRIVE NONEXISTENT OR OFFLINE
1305                                  ;DRVSTA>0 DRIVE IS ONLINE
1306                                  ;DRVSTA<0 DRIVE IS UNSAFE
1307
1308  001370  000                     DRVSTA: .BYTE   0               ;DRIVE 0
1309  001371  000                             .BYTE   0               ;DRIVE 1
1310  001372  000                             .BYTE   0               ;DRIVE 2
1311  001373  000                             .BYTE   0               ;DRIVE 3
1312  001374  000                             .BYTE   0               ;DRIVE 4
1313  001375  000                             .BYTE   0               ;DRIVE 5
1314  001376  000                             .BYTE   0               ;DRIVE 6
1315  001377  000                             .BYTE   0               ;DRIVE 7
1316
1317                          ;BIT TABLE
1318
1319  001400  000001          BITS:   .WORD   BIT00
1320  001402  000002                  .WORD   BIT01
1321  001404  000004                  .WORD   BIT02
1322  001406  000010                  .WORD   BIT03
1323  001410  000020                  .WORD   BIT04
1324  001412  000040                  .WORD   BIT05
1325  001414  000100                  .WORD   BIT06
1326  001416  000200                  .WORD   BIT07
1327  001420  000400                  .WORD   BIT08
1328  001422  001000                  .WORD   BIT09
1329  001424  002000                  .WORD   BIT10
1330  001426  004000                  .WORD   BIT11
1331  001430  010000                  .WORD   BIT12
1332  001432  020000                  .WORD   BIT13
1333  001434  040000                  .WORD   BIT14
1334  001436  100000                  .WORD   BIT15
1335
1336  001440  001500          PAT.PT: .WORD   PAT0            ;TABLE OF POINTERS WHICH POINT TO THE
1337  001442  001540                  .WORD   PAT1            ;PATTERNS USED BY THE DATA TEST
1338  001444  001600                  .WORD   PAT2
1339  001446  001640                  .WORD   PAT3
1340  001450  001700                  .WORD   PAT4
1341  001452  001740                  .WORD   PAT5
1342  001454  002000                  .WORD   PAT6
1343  001456  002040                  .WORD   PAT7
1344  001460  002100                  .WORD   PAT8
1345  001462  002140                  .WORD   PAT9
1346  001464  002200                  .WORD   PAT10
1347  001466  002240                  .WORD   PAT11
1348  001470  002300                  .WORD   PAT12
1349  001472  002340                  .WORD   PAT13
1350  001474  002400                  .WORD   PAT14
1351  001476  002440                  .WORD   PAT15
```

```
1352
1353                                    ;PATTERNS 0 THRU 15
1354
1355  001500  000000                    PAT0:   .WORD   000000          ;PATTERN 0
1356  001502  000000                            .WORD   000000
1357  001504  000000                            .WORD   000000
1358  001506  000000                            .WORD   000000
1359  001510  000000                            .WORD   000000
1360  001512  000000                            .WORD   000000
1361  001514  000000                            .WORD   000000
1362  001516  000000                            .WORD   000000
1363  001520  000000                            .WORD   000000
1364  001522  000000                            .WORD   000000
1365  001524  000000                            .WORD   000000
1366  001526  000000                            .WORD   000000
1367  001530  000000                            .WORD   000000
1368  001532  000000                            .WORD   000000
1369  001534  000000                            .WORD   000000
1370  001536  000000                            .WORD   000000
1371
1372  001540  000001                    PAT1:   .WORD   000001          ;PATTERN 1
1373  001542  000003                            .WORD   000003
1374  001544  000007                            .WORD   000007
1375  001546  000017                            .WORD   000017
1376  001550  000037                            .WORD   000037
1377  001552  000077                            .WORD   000077
1378  001554  000177                            .WORD   000177
1379  001556  000377                            .WORD   000377
1380  001560  000777                            .WORD   000777
1381  001562  001777                            .WORD   001777
1382  001564  003777                            .WORD   003777
1383  001566  007777                            .WORD   007777
1384  001570  017777                            .WORD   017777
1385  001572  037777                            .WORD   037777
1386  001574  077777                            .WORD   077777
1387  001576  177777                            .WORD   177777
1388
1389  001600  177776                    PAT2:   .WORD   177776          ;PATTERN 2
1390  001602  177774                            .WORD   177774
1391  001604  177770                            .WORD   177770
1392  001606  177760                            .WORD   177760
1393  001610  177740                            .WORD   177740
1394  001612  177700                            .WORD   177700
1395  001614  177600                            .WORD   177600
1396  001616  177400                            .WORD   177400
1397  001620  177000                            .WORD   177000
1398  001622  176000                            .WORD   176000
1399  001624  174000                            .WORD   174000
1400  001626  170000                            .WORD   170000
1401  001630  160000                            .WORD   160000
1402  001632  140000                            .WORD   140000
1403  001634  100000                            .WORD   100000
1404  001636  000000                            .WORD   000000
1405
1406  001640  000000                    PAT3:   .WORD   000000          ;PATTERN 3
1407  001642  000000                            .WORD   000000
```

```
 1408  001644  000000                    .WORD   000000
 1409  001646  177777                    .WORD   177777
 1410  001650  177777                    .WORD   177777
 1411  001652  177777                    .WORD   177777
 1412  001654  000000                    .WORD   000000
 1413  001656  000000                    .WORD   000000
 1414  001660  177777                    .WORD   177777
 1415  001662  177777                    .WORD   177777
 1416  001664  000000                    .WORD   000000
 1417  001666  177777                    .WORD   177777
 1418  001670  000000                    .WORD   000000
 1419  001672  177777                    .WORD   177777
 1420  001674  177777                    .WORD   177777
 1421  001676  000000                    .WORD   000000
 1422
 1423  001700  000000            PAT4:   .WORD   000000          ;PATTERN 4
 1424  001702  010421                    .WORD   010421
 1425  001704  021042                    .WORD   021042
 1426  001706  031463                    .WORD   031463
 1427  001710  042104                    .WORD   042104
 1428  001712  052525                    .WORD   052525
 1429  001714  063146                    .WORD   063146
 1430  001716  073567                    .WORD   073567
 1431  001720  104210                    .WORD   104210
 1432  001722  114631                    .WORD   114631
 1433  001724  125252                    .WORD   125252
 1434  001726  135673                    .WORD   135673
 1435  001730  146314                    .WORD   146314
 1436  001732  156735                    .WORD   156735
 1437  001734  167356                    .WORD   167356
 1438  001736  177777                    .WORD   177777
 1439
 1440  001740  052525            PAT5:   .WORD   052525          ;PATTERN 5
 1441  001742  052525                    .WORD   052525
 1442  001744  052525                    .WORD   052525
 1443  001746  125252                    .WORD   125252
 1444  001750  125252                    .WORD   125252
 1445  001752  125252                    .WORD   125252
 1446  001754  052525                    .WORD   052525
 1447  001756  052525                    .WORD   052525
 1448  001760  125252                    .WORD   125252
 1449  001762  125252                    .WORD   125252
 1450  001764  052525                    .WORD   052525
 1451  001766  125252                    .WORD   125252
 1452  001770  052525                    .WORD   052525
 1453  001772  125252                    .WORD   125252
 1454  001774  125252                    .WORD   125252
 1455  001776  052525                    .WORD   052525
 1456
 1457  002000  007417            PAT6:   .WORD   007417          ;PATTERN 6
 1458  002002  007417                    .WORD   007417
 1459  002004  007417                    .WORD   007417
 1460  002006  170360                    .WORD   170360
 1461  002010  170360                    .WORD   170360
 1462  002012  170360                    .WORD   170360
 1463  002014  007417                    .WORD   007417
```

```
 1464  002016  007417                    .WORD   007417
 1465  002020  170360                    .WORD   170360
 1466  002022  170360                    .WORD   170360
 1467  002024  007417                    .WORD   007417
 1468  002026  170360                    .WORD   170360
 1469  002030  007417                    .WORD   007417
 1470  002032  170360                    .WORD   170360
 1471  002034  170360                    .WORD   170360
 1472  002036  007417                    .WORD   007417
 1473
 1474  002040  026455            PAT7:   .WORD   026455          ;PATTERN 7
 1475  002042  026455                    .WORD   026455
 1476  002044  026455                    .WORD   026455
 1477  002046  151322                    .WORD   151322
 1478  002050  151322                    .WORD   151322
 1479  002052  151322                    .WORD   151322
 1480  002054  026455                    .WORD   026455
 1481  002056  026455                    .WORD   026455
 1482  002060  151322                    .WORD   151322
 1483  002062  151322                    .WORD   151322
 1484  002064  026455                    .WORD   026455
 1485  002066  151322                    .WORD   151322
 1486  002070  026455                    .WORD   026455
 1487  002072  151322                    .WORD   151322
 1488  002074  151322                    .WORD   151322
 1489  002076  026455                    .WORD   026455
 1490
 1491  002100  077577            PAT8:   .WORD   077577          ;PATTERN 8 (WORST CASE PATTERN)
 1492  002102  077577                    .WORD   077577
 1493  002104  077577                    .WORD   077577
 1494  002106  077577                    .WORD   077577
 1495  002110  077577                    .WORD   077577
 1496  002112  077577                    .WORD   077577
 1497  002114  077577                    .WORD   077577
 1498  002116  077577                    .WORD   077577
 1499  002120  077577                    .WORD   077577
 1500  002122  077577                    .WORD   077577
 1501  002124  077577                    .WORD   077577
 1502  002126  077577                    .WORD   077577
 1503  002130  077577                    .WORD   077577
 1504  002132  077577                    .WORD   077577
 1505  002134  077577                    .WORD   077577
 1506  002136  077577                    .WORD   077577
 1507
 1508  002140  000001            PAT9:   .WORD   000001          ;PATTERN 9
 1509  002142  000002                    .WORD   000002
 1510  002144  000004                    .WORD   000004
 1511  002146  000010                    .WORD   000010
 1512  002150  000020                    .WORD   000020
 1513  002152  000040                    .WORD   000040
 1514  002154  000100                    .WORD   000100
 1515  002156  000200                    .WORD   000200
 1516  002160  000400                    .WORD   000400
 1517  002162  001000                    .WORD   001000
 1518  002164  002000                    .WORD   002000
 1519  002166  004000                    .WORD   004000
```

```
 1520  002170  010000                        .WORD   010000
 1521  002172  020000                        .WORD   020000
 1522  002174  040000                        .WORD   040000
 1523  002176  100000                        .WORD   100000
 1524
 1525  002200  177776              PAT10:    .WORD   177776          ;PATTERN 10
 1526  002202  177775                        .WORD   177775
 1527  002204  177773                        .WORD   177773
 1528  002206  177767                        .WORD   177767
 1529  002210  177757                        .WORD   177757
 1530  002212  177737                        .WORD   177737
 1531  002214  177677                        .WORD   177677
 1532  002216  177577                        .WORD   177577
 1533  002220  177377                        .WORD   177377
 1534  002222  176777                        .WORD   176777
 1535  002224  175777                        .WORD   175777
 1536  002226  173777                        .WORD   173777
 1537  002230  167777                        .WORD   167777
 1538  002232  157777                        .WORD   157777
 1539  002234  137777                        .WORD   137777
 1540  002236  077777                        .WORD   077777
 1541
 1542  002240  172666              PAT11:    .WORD   172666          ;PATTERN 11
 1543  002242  155555                        .WORD   155555
 1544  002244  172666                        .WORD   172666
 1545  002246  155555                        .WORD   155555
 1546  002250  172666                        .WORD   172666
 1547  002252  155555                        .WORD   155555
 1548  002254  172666                        .WORD   172666
 1549  002256  155555                        .WORD   155555
 1550  002260  172666                        .WORD   172666
 1551  002262  155555                        .WORD   155555
 1552  002264  172666                        .WORD   172666
 1553  002266  155555                        .WORD   155555
 1554  002270  172666                        .WORD   172666
 1555  002272  155555                        .WORD   155555
 1556  002274  172666                        .WORD   172666
 1557  002276  155555                        .WORD   155555
 1558
 1559  002300  077777              PAT12:    .WORD   077777          ;PATTERN 12
 1560  002302  137777                        .WORD   137777
 1561  002304  157777                        .WORD   157777
 1562  002306  167777                        .WORD   167777
 1563  002310  173777                        .WORD   173777
 1564  002312  175777                        .WORD   175777
 1565  002314  176777                        .WORD   176777
 1566  002316  177377                        .WORD   177377
 1567  002320  177577                        .WORD   177577
 1568  002322  177677                        .WORD   177677
 1569  002324  177737                        .WORD   177737
 1570  002326  177757                        .WORD   177757
 1571  002330  177767                        .WORD   177767
 1572  002332  177773                        .WORD   177773
 1573  002334  177775                        .WORD   177775
 1574  002336  177776                        .WORD   177776
 1575
```

```
1576  002340  177777          PAT13:  .WORD   177777          ;PATTERN 13
1577  002342  177777                  .WORD   177777
1578  002344  177777                  .WORD   177777
1579  002346  177777                  .WORD   177777
1580  002350  177777                  .WORD   177777
1581  002352  177777                  .WORD   177777
1582  002354  177777                  .WORD   177777
1583  002356  177777                  .WORD   177777
1584  002360  177777                  .WORD   177777
1585  002362  177777                  .WORD   177777
1586  002364  177777                  .WORD   177777
1587  002366  177777                  .WORD   177777
1588  002370  177777                  .WORD   177777
1589  002372  177777                  .WORD   177777
1590  002374  177777                  .WORD   177777
1591  002376  177777                  .WORD   177777
1592
1593  002400  000000          PAT14:  .WORD   000000          ;PATTERN 14
1594  002402  000000                  .WORD   000000
1595  002404  177777                  .WORD   177777
1596  002406  177777                  .WORD   177777
1597  002410  177777                  .WORD   177777
1598  002412  177777                  .WORD   177777
1599  002414  177777                  .WORD   177777
1600  002416  177777                  .WORD   177777
1601  002420  177777                  .WORD   177777
1602  002422  177777                  .WORD   177777
1603  002424  177777                  .WORD   177777
1604  002426  177777                  .WORD   177777
1605  002430  177777                  .WORD   177777
1606  002432  177777                  .WORD   177777
1607  002434  177777                  .WORD   177777
1608  002436  177777                  .WORD   177777
1609
1610  002440  177777          PAT15:  .WORD   177777          ;PATTERN 15
1611  002442  177777                  .WORD   177777
1612  002444  000000                  .WORD   000000
1613  002446  000000                  .WORD   000000
1614  002450  000000                  .WORD   000000
1615  002452  000000                  .WORD   000000
1616  002454  000000                  .WORD   000000
1617  002456  000000                  .WORD   000000
1618  002460  000000                  .WORD   000000
1619  002462  000000                  .WORD   000000
1620  002464  000000                  .WORD   000000
1621  002466  000000                  .WORD   000000
1622  002470  000000                  .WORD   000000
1623  002472  000000                  .WORD   000000
1624  002474  000000                  .WORD   000000
1625  002476  000000                  .WORD   000000
1626
1627                          ;DRIVE OPERATION CONTROL BLOCK
1628
1629  002500      000         DPB:    .BYTE   0               ;DRIVE NUMBER
1630  002501      000                 .BYTE   0               ;'MEX' BITS
1631  002502      000                 .BYTE   0               ;OPERATION CODE
```

```
 1632  002503     000                    .BYTE   0               ;MODE AND HDR BITS
 1633  002504  000000            $WC:    .WORD   0               ;WORD COUNT
 1634  002506  000000            $BUF:   .WORD   0               ;BUFFER ADDRESS
 1635  002510  000000            $CYL:   .WORD   0               ;CYLINDER ADDRESS
 1636  002512     000            $SEC:   .BYTE   0               ;SECTOR ADDRESS
 1637  002513     000            $TRK:   .BYTE   0               ;TRACK ADDRESS
 1638
```

```
                                .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
```

```
 1751  002670  043172                    DT16
 1752  002672  043616                    DF53
 1753
 1754                                ;ERROR 17
 1755
 1756  002674  033260                    EM17          ;ERROR READING TEST SECTOR
 1757  002676  040323                    DH2
 1758  002700  043040                    DT2
 1759  002702  043522                    DF2
 1760
 1761                                ;ERROR 20
 1762
 1763  002704  033312                    EM20          ;CONTENTS OF FIRST SECTOR OF PACK CHANGED AFTER
 1764                                                  ;FORCING 'EOP' DURING WRITE
 1765  002706  040637                    DH11
 1766  002710  043126                    DT11
 1767  002712  043542                    DF11
 1768
 1769                                ;ERROR 21
 1770
 1771  002714  033425                    EM21          ;'PROG' ERROR DIDN'T SET WHEN READ COMMAND ISSUED
 1772                                                  ;WHILE CONTROLLER BUSY
 1773  002716  040775                    DH14
 1774  002720  043156                    DT14
 1775  002722  043542                    DF11
 1776
 1777                                ;ERROR 22
 1778
 1779  002724  033535                    EM22          ;'ERR' DIDN'T SET WITH 'PROG'
 1780  002726  040775                    DH14
 1781  002730  043156                    DT14
 1782  002732  043542                    DF11
 1783
 1784                                ;ERROR 23
 1785
 1786  002734  033572                    EM23          ;ERROR ATTEMPTING TO MISFORMAT SECTOR 0
 1787  002736  040323                    DH2
 1788  002740  043040                    DT2
 1789  002742  043522                    DF2
 1790
 1791                                ;ERROR 24
 1792
 1793  002744  033641                    EM24          ;ERROR VERIFYING THE MISFORMATTED TEST HEADER
 1794  002746  040775                    DH14
 1795  002750  043156                    DT14
 1796  002752  043542                    DF11
 1797
 1798                                ;ERROR 25
 1799
 1800  002754  033716                    EM25          ;MISFORMATTED TEST HEADER IS NOT CORRECT
 1801  002756  041167                    DH25
 1802  002760  043204                    DT25
 1803  002762  043546                    DF25
 1804
 1805                                ;ERROR 26
 1806
```

```
 1807  002764  033766                    EM26          ;'HNF' DIDN'T SET WHEN SEARCHING FOR MISFORMATTED SECTOR
 1808  002766  040323                    DH2
 1809  002770  043040                    DT2
 1810  002772  043522                    DF2
 1811
 1812                                ;ERROR 27
 1813
 1814  002774  034056                    EM27          ;'DSK' NOT SET WITH 'HNF'
 1815  002776  040775                    DH14
 1816  003000  043156                    DT14
 1817  003002  043542                    DF11
 1818
 1819                                ;ERROR 30
 1820
 1821  003004  034107                    EM30          ;'HE' DIDN'T SET WITH 'HNF'
 1822  003006  040775                    DH14
 1823  003010  043156                    DT14
 1824  003012  043542                    DF11
 1825
 1826                                ;ERROR 31
 1827
 1828  003014  034142                    EM31          ;'ERR' NOT SET WITH 'HNF'
 1829  003016  040775                    DH14
 1830  003020  043156                    DT14
 1831  003022  043542                    DF11
 1832
 1833                                ;ERROR 32
 1834
 1835  003024  034173                    EM32          ;ERROR WHILE RESTORING MISFORMATTED TEST HEADER
 1836  003026  040323                    DH2
 1837  003030  043040                    DT2
 1838  003032  043522                    DF2
 1839
 1840                                ;ERROR 33
 1841
 1842  003034  034252                    EM33          ;CONTROLLER DIDN'T BECOME BUSY WHEN READ COMMAND
 1843                                                  ;ISSUED TO SEEKING DRIVE
 1844  003036  040323                    DH2
 1845  003040  043040                    DT2
 1846  003042  043522                    DF2
 1847
 1848                                ;ERROR 34
 1849
 1850  003044  034363                    EM34          ;ERROR DURING READ COMMAND WHICH WAS ISSUED TO SEEKING DRIVE
 1851  003046  040323                    DH2
 1852  003050  043040                    DT2
 1853  003052  043522                    DF2
 1854
 1855                                ;ERROR 35
 1856
 1857  003054  034457                    EM35          ;DATA INCORRECT FROM HEADER READ COMMAND ISSUED
 1858                                                  ;TO SEEKING DRIVE
 1859  003056  041167                    DH25
 1860  003060  043204                    DT25
 1861  003062  043546                    DF25
 1862
```

```
 1863                                   ;ERROR 36
 1864
 1865  003064  034561                          EM36            ;'NXME' DIDN'T SET WHEN LOCATION 760000 ADDRESSED
 1866  003066  040323                          DH2
 1867  003070  043040                          DT2
 1868  003072  043522                          DF2
 1869
 1870                                   ;ERROR 37
 1871
 1872  003074  034642                          EM37            ;'HE' DIDN'T SET WITH 'NXME'
 1873  003076  040775                          DH14
 1874  003100  043156                          DT14
 1875  003102  043542                          DF11
 1876
 1877                                   ;ERROR 40
 1878
 1879  003104  034676                          EM40            ;'ERR' DIDN7T SET WITH 'NXME'
 1880  003106  040775                          DH14
 1881  003110  043156                          DT14
 1882  003112  043542                          DF11
 1883
 1884                                   ;ERROR 41
 1885
 1886  003114  034733                          EM41            ;ERROR FORMATTING TRACK 0
 1887  003116  040323                          DH2
 1888  003120  043040                          DT2
 1889  003122  043522                          DF2
 1890
 1891                                   ;ERROR 42
 1892
 1893  003124  034764                          EM42            ;ERROR READING THE HEADER FROM THE TEST SECTOR
 1894  003126  040323                          DH2
 1895  003130  043040                          DT2
 1896  003132  043522                          DF2
 1897
 1898                                   ;ERROR 43
 1899
 1900  003134  035042                          EM43            ;SECTOR FIELD FROM HEADER IS NOT CORRECT
 1901  003136  041256                          DH43
 1902  003140  043224                          DT43
 1903  003142  043556                          DF43
 1904
 1905                                   ;ERROR 44
 1906
 1907  003144  035112                          EM44            ;ERROR WRITING SECTOR ADDRESS IN DATA FIELD
 1908  003146  040323                          DH2
 1909  003150  043040                          DT2
 1910  003152  043522                          DF2
 1911
 1912                                   ;ERROR 45
 1913
 1914  003154  035165                          EM45            ;'SOT' OR SECTOR ADDRESS REGISTER IS NOT CORRECT
 1915  003156  041502                          DH45
 1916  003160  043262                          DT45
 1917  003162  043572                          DF45
 1918
```

```
 1919                                   ;ERROR 46
 1920
 1921  003164  035245                          EM46            ;DATA NOT CORRECT FOR SECTOR READ
 1922  003166  041674                          DH46
 1923  003170  043306                          DT46
 1924  003172  043542                          DF11
 1925
 1926                                   ;ERROR 47
 1927
 1928  003174  035316                          EM47            ;SECTOR CONTENTS WRONG, DATA SHOULD BE THE
 1929                                                          ;TRACK NUMBER OF THE CURRENTLY SELECTED HEAD
 1930  003176  042031                          DH47
 1931  003200  043322                          DT47
 1932  003202  043602                          DF47
 1933
 1934                                   ;ERROR 50
 1935
 1936  003204  035445                          EM50            ;ERROR AFTER 2 WORD READ STARTING AT LOC 177776
 1937  003206  040323                          DH2
 1938  003210  043040                          DT2
 1939  003212  043522                          DF2
 1940
 1941                                   ;ERROR 51
 1942
 1943  003214  035524                          EM51            ;'MEX0' DIDN'T SET AFTER 2 WORD READ STARTING
 1944                                                          ;AT LOC 177776
 1945  003216  042145                          DH51
 1946  003220  043346                          DT51
 1947  003222  043612                          DF51
 1948
 1949                                   ;ERROR 52
 1950
 1951  003224  035620                          EM52            ;'MEX1' SET AFTER 2 WORD READ STARTING AT LOC 177776
 1952  003226  042145                          DH51
 1953  003230  043346                          DT51
 1954  003232  043612                          DF51
 1955
 1956                                   ;ERROR 53
 1957
 1958  003234  035704                          EM53            ;CONTENTS OF LOC 200000 WRONG AFTER 2 WORD READ
 1959                                                          ;STARTING AT LOC 177776
 1960  003236  041167                          DH25
 1961  003240  043204                          DT25
 1962  003242  043616                          DF53
 1963
 1964                                   ;ERROR 54
 1965
 1966  003244  036013                          EM54            ;ERROR AFTER 2 WORD READ STARTING AT LOC 377776
 1967  003246  040323                          DH2
 1968  003250  043040                          DT2
 1969  003252  043522                          DF2
 1970
 1971                                   ;ERROR 55
 1972
 1973
 1974  003254  036072                          EM55            ;'MEX0' DIDN'T CLEAR AFTER 2 WORD STARTING AT LOC 377776
```

```
 1975  003256  042145                    DH51
 1976  003260  043346                    DT51
 1977  003262  043612                    DF51
 1978
 1979                            ;ERROR 56
 1980
 1981  003264  036162                    EM56        ;'MEX1' DIDN'T SET AFTER 2 WORD READ STARTING AT LOC 377776
 1982  003266  042145                    DH51
 1983  003270  043346                    DT51
 1984  003272  043612                    DF51
 1985
 1986                            ;ERROR 57
 1987
 1988  003274  036255                    EM57        ;CONTENTS OF LOC 400000 WRONG AFTER 2 WORD READ
 1989                                                ;STARTING AT LOCATION 377776
 1990  003276  041167                    DH25
 1991  003300  043204                    DT25
 1992  003302  043616                    DF53
 1993
 1994                            ;ERROR 60
 1995
 1996  003304  036371                    EM60        ;ERROR RETURNING SECTOR TO PDP-11 MODE USING A
 1997                                                ;2 WORD WRITE
 1998  003306  040323                    DH2
 1999  003310  043040                    DT2
 2000  003312  043522                    DF2
 2001
 2002                            ;ERROR 61
 2003
 2004  003314  036465                    EM61        ;ERROR WRITING TEST SECTOR(S)
 2005  003316  040465                    DH5
 2006  003320  043072                    DT5
 2007  003322  043532                    DF5
 2008
 2009                            ;ERROR 62
 2010
 2011  003324  036522                    EM62        ;ERROR READING TEST SECTOR(S)
 2012  003326  040465                    DH5
 2013  003330  043072                    DT5
 2014  003332  043532                    DF5
 2015
 2016                            ;ERROR 63
 2017
 2018  003334  036557                    EM63        ;DRIVE DIDN'T RETURN TO CYL 0 FROM THE MAXIMUM
 2019                                                ;CYL ON CONTROLLER POWER FAIL
 2020  003336  042242                    DH63
 2021  003340  043366                    DT63
 2022  003342  043622                    DF63
 2023
 2024                            ;ERROR 64
 2025
 2026  003344  036673                    EM64        ;CONTENTS OF MEMORY CHANGED DURING POWER
 2027                                                ;FAIL WHILE READING THE DISK
 2028  003346  040720                    DH12
 2029  003350  043142                    DT12
 2030  003352  043542                    DF11
```

```
 2031
 2032                                   ;ERROR 65
 2033
 2034  003354  037000                          EM65            ;'SUOL' SET WITH DRIVE DISABLED
 2035  003356  040775                          DH14
 2036  003360  043156                          DT14
 2037  003362  043542                          DF11
 2038
 2039                                   ;ERROR 66
 2040
 2041  003364  037037                          EM66            ;'SUOL' NOT SET WITH DRIVE ENABLED
 2042  003366  040775                          DH14
 2043  003370  043156                          DT14
 2044  003372  043542                          DF11
 2045
 2046                                   ;ERROR 67
 2047
 2048  003374  037101                          EM67            ;'SUWP' NOT SET WITH DRIVE SET TO 'READ ONLY'
 2049  003376  040775                          DH14
 2050  003400  043156                          DT14
 2051  003402  043542                          DF11
 2052
 2053                                   ;ERROR 70
 2054
 2055  003404  037156                          EM70            ;'SUWP' SET WITH DRIVE SET TO 'READ/WRITE'
 2056  003406  040775                          DH14
 2057  003410  043156                          DT14
 2058  003412  043542                          DF11
 2059
 2060                                   ;ERROR 71
 2061
 2062  003414  037230                          EM71            ;'SUWP' SET WITH RP11 'WRITE LOCKOUT' SET, CYLINDER
 2063                                                          ;LOA'S = 0, AND RPCA = 0
 2064  003416  042327                          DH71
 2065  003420  043404                          DT71
 2066  003422  043622                          DF63
 2067
 2068                                   ;ERROR 72
 2069
 2070  003424  037344                          EM72            ;'SUWP' SET WITH RPCA = 2 & CYL LOA'S = 0
 2071  003426  042327                          DH71
 2072  003430  043404                          DT71
 2073  003432  043622                          DF63
 2074
 2075                                   ;ERROR 73
 2076
 2077  003434  037415                          EM73            ;'SUWP' NOT SET; RPCA = VALUE IN CYL LOA'S
 2078  003436  042327                          DH71
 2079  003440  043404                          DT71
 2080  003442  043622                          DF63
 2081
 2082                                   ;ERROR 74
 2083
 2084  003444  037467                          EM74            ;'SUWP' SET WITH RPCA ONE GREATER THAN CYL LOA VALUE
 2085  003446  042327                          DH71
 2086  003450  043404                          DT71
```

```
 2087  003452  043622                    DF63
 2088
 2089                              ;ERROR 75
 2090
 2091  003454  037553                    EM75          ;'SUWP' SET WITH DRIVE LOA SWITCHES EQUAL SELECTED DRIVE
 2092  003456  040775                    DH14
 2093  003460  043156                    DT14
 2094  003462  043542                    DF11
 2095
 2096                              ;ERROR 76
 2097
 2098  003464  037646                    EM76          ;DRIVE HAS GONE OFFLINE
 2099  003466  040323                    DH2
 2100  003470  043040                    DT2
 2101  003472  043522                    DF2
 2102
 2103                              ;ERROR 77
 2104
 2105  003474  037675                    EM77          ;DRIVE IS UNSAFE
 2106  003476  040323                    DH2
 2107  003500  043040                    DT2
 2108  003502  043522                    DF2
 2109
 2110                              ;ERROR 100
 2111
 2112  003504  037715                    EM100         ;DRIVE TIMED OUT
 2113  003506  040323                    DH2
 2114  003510  043040                    DT2
 2115  003512  043522                    DF2
 2116
 2117                              ;ERROR 101
 2118
 2119  003514  037735                    EM101         ;CONTROLLER TIMED OUT
 2120  003516  040323                    DH2
 2121  003520  043040                    DT2
 2122  003522  043522                    DF2
 2123
 2124                              ;ERROR 102
 2125
 2126  003524  037762                    EM102         ;DATA COMPARE ERROR
 2127  003526  042414                    DH102
 2128  003530  043422                    DT102
 2129  003532  043626                    DF102
 2130
 2131                              ;ERROR 103
 2132
 2133  003534  040005                    EM103         ;DATA COMPARE ERROR (MEMORY MANAGEMENT ENABLED)
 2134  003536  042414                    DH102
 2135  003540  043450                    DT103
 2136  003542  043636                    DF103
 2137
 2138                              ;ERROR 104
 2139
 2140  003544  000000                    0             ;DATA COMPARE ERROR DETAIL LINE
 2141  003546  000000                    0
 2142  003550  043504                    DT104
```

```
 2143  003552  043652                    DF104
 2144
 2145                              ;ERROR 105
 2146
 2147  003554  000000                    0                ;DATA COMPARE ERROR SUMMARY LINE
 2148  003556  042717                    DH105
 2149  003560  043512                    DT105
 2150  003562  043516                    DF1
 2151
 2152                              ;ERROR 106
 2153
```

```
 2154  003564  040064                    EM106          ;'ERR' DIDN'T SET WITH 'WCE'
 2155  003566  040775                    DH14
 2156  003570  043156                    DT14
 2157  003572  043542                    DF11
 2158
 2159                                ;ERROR 107
 2160
```

```
 2161  003574  040120                  EM107           ;'HE' DIDN'T SET WITH 'PROG'
 2162  003576  040775                  DH14
 2163  003600  043156                  DT14
 2164  003602  043542                  DF11
 2165
 2166                            ;ERROR 110
 2167
 2168  003604  000000                  0               ;JNSUCESSFUL AFTER 3 RETRIES
 2169  003606  042745                  DH110
 2170  003610  000000                  0
 2171  003612  000000                  0
 2172
 2173                            ;ERROR 111
 2174
 2175  003614  000000                  0               ;SUCESSFUL AFTER 'N' RETRIES
 2176  003616  043000                  DH111
 2177  003620  043514                  DT111
 2178  003622  043516                  DF1
 2179
 2180                            ;ERROR 112
 2181
 2182  003624  040154                  EM112           ;DRIVE UNSAFE AFTER POWER FAIL
 2183  003626  040323                  DH2
 2184  003630  043040                  DT2
 2185  003632  043522                  DF2
 2186
 2187                            ;ERROR 113
 2188
 2189  003634  040212                  EM113           ;'SUWP' NOT SET WITH DRIVE LOA SWITCHES GREATER THAN SELECTED DR
 2190  003636  040775                  DH14
 2191  003640  043156                  DT14
 2192  003642  043542                  DF11
 2193
```

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

                                        .SBTTL  START OF PROGRAM

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

003644  012737  177777  001216  START3: MOV     #-1,BUSADR      ;SET BUSADR FLAG
003652  000402                          BR      STRT1A
003654  005037  001216          START1: CLR     @#BUSADR        ;CLR BUSADR FLAG
003660  005037  001206          STRT1A: CLR     @#CNTRLC        ;NO CONVERSATION MODE
003664  000411                          BR      START           ;
003666  012737  177777  001216  START4: MOV     #-1,BUSADR      ;SET BUSADR FLAG
003674  000402                          BR      STRT2A
003676  005037  001216          START2: CLR     @#BUSADR        ;CLR BUSADR FLAG
003702  012737  177777  001206  STRT2A: 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                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
003720  022706  001140                  CMP     #SWR,R6 ;;DONE?
003724  001374                          BNE     .-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?
```

```
2250  004132  001405                          BEQ     START5          ;YES, SKIP TITLE PRINTOUT
2251  004134  005227  177777                  INC     #-1             ;FIRST START ?
2252  004140  001002                          BNE     START5          ;BR IF NOT
2253  004142  104401  043656                  TYPE    ,TITLE          ;TYPE THE PROGRAM'S MAINDEC NUMBER AND NAME
2254  004146  004737  020416          START5: JSR     PC,$TKINT       ;SETUP THE TTY KEYBOARD
2255                                  .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
2256  004152  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
2257  004156  001006                          BNE     64$             ;;BRANCH IF YES
2258  004160  023727  001140  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
2259  004166  001005                          BNE     65$             ;;BRANCH IF NO
2260  004170  104406                          GTSWR                   ;;GET SOFT-SWR SETTINGS
2261  004172  000403                          BR      65$
2262  004174  112737  000001  001134  64$:     MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
2263  004202                          65$:
2264  004202  004737  023452                  JSR     PC,RESLDR       ;RESTORE THE LOADER
2265  004206  004737  025700                  JSR     PC,SIZMEM       ;SET MEMORY SIZE VARIABLES
2266  004212  005037  001112                  CLR     $ERTTL          ;CLEAR TOTAL ERROR COUNT
2267  004216  005037  001260                  CLR     OPRSEL          ;CLEAR OPERATOR SELECTED ADDRESS FLAG
2268  004222  005037  001266                  CLR     SCYL            ;OPERATOR SPECIFIED CYLINDER
2269  004226  005037  001270                  CLR     STRK            ;OPERATOR SPECIFIED TRACK
2270  004232  005037  001272                  CLR     SSEC            ;OPERATOR SPECIFIED SECTOR
2271  004236  013737  001264  001256          MOV     MAXWC,WC14      ;ASSUME THAT MAXIMUM WORD COUNT LESS THAN 8K
2272  004244  023737  001264  001262          CMP     MAXWC,LMT8K     ;MAXIMUM WORD COUNT GREATER THAN LIMIT
2273  004252  101403                          BLOS    1$              ;BR IF NOT
2274  004254  013737  001262  001256          MOV     LMT8K,WC14      ;USE AN 8K MAXIMUM
2275  004262  012737  177777  001300  1$:     MOV     #177777,PATTRN  ;ENABLE ALL PATTERNS
2276  004270  012737  017777  001276          MOV     #17777,TSTNMS   ;SELECT TESTS 0 - 14
2277  004276  005037  177776                  CLR     PSW             ;ENSURE THE PRIORITY = 0
2278  004302  005037  001310                  CLR     DRVBAD          ;CLEAR OFFLINE/UNSAFE DRIVE BITS
2279  004306  005037  001162                  CLR     $TIMES          ;INITIALIZE NUMBER OF ITERATIONS
2280  004312  005037  001164                  CLR     $ESCAPE         ;CLEAR THE ESCAPE ON ERROR ADDRESS
2281  004316  012737  016616  001200          MOV     #$EOP,BYPASS    ;INITIAL BYPASS ADDRESS
2282  004324  012737  000001  001104          MOV     #1,$ICNT        ;PRESET ITERATION COUNT TO 1
2283  004332  112737  000001  001115          MOVB    #1,$ERMAX       ;ALLOW ONE ERROR PER TEST
2284  004340  012737  004340  001106          MOV     #.,$LPADR       ;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2285  004346  012737  004346  001110          MOV     #.,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
2286  004354  005037  001222                  CLR     MMACTV          ;CLEAR MEMORY MANAGEMENT FLAG
2287  004360  005037  177776          START6: CLR     PS              ;CLEAR THE PROCESSOR PRIORITY
2288  004364  004737  026150                  JSR     PC,GETADR       ;CHECK THE RP11 ADDRESS
2289  004370  004737  022566                  JSR     PC,RPINIT       ;FIND OUT WHICH DRIVES ARE ON SYSTEM
2290  004374  012737  000312  001210          MOV     #202.,MAXCYL    ;ASSUME RP02'S
2291  004402  005737  001304                  TST     DRVTYP          ;WHICH DRIVES ?
2292  004406  001403                          BEQ     2$              ;BR IF THEY REALLY WERE RP02'S
2293  004410  012737  000625  001210          MOV     #405.,MAXCYL    ;SET MAX CYLINDER FOR RP03'S
2294  004416  005227  177777          2$:     INC     #-1             ;FIRST START ?
2295  004422  001404                          BEQ     9$              ;BR IF IT IS
2296  004424  032777  000200  174506          BIT     #SW07,@SWR      ;BYPASS DRIVE STATUS TYPEOUT ?
2297  004432  001044                          BNE     START7          ;BR IF YES
2298  004434  104401  027236          9$:     TYPE    ,DRSTAT         ;'DRIVE STATUS'
2299  004440  005001                          CLR     R1              ;CLEAR TABLE POINTER
2300  004442  012702  000010                  MOV     #8.,R2          ;COUNTER
2301  004446                          3$:
2302  004446  010146                          MOV     R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
2303                                                                  ;;TYPE DRIVE NUMBER
2304  004450  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
2305  004452     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
```

```
2306  004453    000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
2307  004454  105761  001370                 TSTB    DRVSTA(R1)      ;CHECK DRIVE'S STATUS
2308  004460  001420                         BEQ     7$              ;DRIVE IS OFFLINE OR NON-EXISTENT
2309  004462  100403                         BMI     4$              ;DRIVE IS UNSAFE
2310  004464  104401  027262                 TYPE    ,ONLINE         ;'ONLINE'
2311  004470  000402                         BR      5$              ;
2312  004472  104401  027276         4$:     TYPE    ,UNSAFE         ;'UNSAFE'
2313  004476  136137  001360  001304  5$:     BITB    ATABIT(R1),DRVTYP  ;SEE WHICH DRIVE TYPE
2314  004504  001003                         BNE     6$              ;BR IF RP03
2315  004506  104401  027327                 TYPE    ,RP02           ;'RP02'
2316  004512  000405                         BR      8$              ;
2317  004514  104401  027336         6$:     TYPE    ,RP03           ;'RP03'
2318  004520  000402                         BR      8$              ;
2319  004522  104401  027312         7$:     TYPE    ,OFFLIN         ;'OFFLINE'
2320  004526  104401  001173         8$:     TYPE    ,$CRLF          ;CR-LF
2321  004532  005201                         INC     R1              ;INCREMENT TABLE POINTER
2322  004534  005302                         DEC     R2              ;DECREMENT THE COUNTER
2323  004536  001343                         BNE     3$              ;CONTINUE
2324  004540  104401  001173                 TYPE    ,$CRLF          ;CR-LF
2325  004544  005737  001206         START7: TST     CNTRLC          ;CONVERSATION MODE ?
2326  004550  001403                         BEQ     1$              ;NO--BRANCH
2327  004552  004737  026314                 JSR     PC,ENTPRM       ;YES--GET PARAMETERS
2328  004556  000416                         BR      5$              ;GO TYPE DRIVES TO BE TESTED
2329  004560  005037  001312         1$:     CLR     DRVSEL          ;NO DRIVES SELECTED
2330  004564  005000                         CLR     R0              ;DETERMINE THE DRIVES THAT
2331  004566  012701  000001                 MOV     #1,R1           ;ARE AVAILABLE FOR TESTING
2332  004572  105760  001370         2$:     TSTB    DRVSTA(R0)      ;DRIVE ONLINE ?
2333  004576  003403                         BLE     3$              ;BR IF NOT
2334  004600  156037  001360  001312         BISB    ATABIT(R0),DRVSEL  ;SET SELECTION BIT FOR DRIVE
2335  004606  005200                 3$:     INC     R0              ;INCREMENT DRIVE ADDRESS
2336  004610  106301                         ASLB    R1              ;COUNT
2337  004612  001367                         BNE     2$              ;BR IF NOT ALL DRIVES CHECKED
2338  004614  104401  027345         5$:     TYPE    ,DRVTST         ;'DRIVES TO BE TESTED'
2339  004620  005037  017004                 CLR     $ENDCT          ;DETERMINE PASSES TO MAKE AND
2340  004624  005000                         CLR     R0              ;THE DRIVES TO BE TESTED
2341  004626  013701  001312                 MOV     DRVSEL,R1       ;ANY DRIVES SELECTED?
2342  004632  001004                         BNE     6$              ;YES--BRANCH
2343  004634  104401  027372                 TYPE    ,NONE           ;'NONE'
2344  004640  000137  016616                 JMP     @#$EOP          ;GO TO END OF PROGRAM
2345  004644  005737  001206         6$:     TST     CNTRLC          ;CONVERSATION MODE START ?
2346  004650  001011                         BNE     7$              ;BR IF NOT
2347  004652  005737  000042                 TST     42              ;UNDER MONITOR CONTROL ?
2348  004656  001406                         BEQ     7$              ;BR IF NOT
2349  004660  122737  000003  000041         CMPB    #3,41           ;LOADED BY RP02/RP03 ?
2350  004666  001002                         BNE     7$              ;BR IF NOT
2351  004670  006201                         ASR     R1              ;EXCLUDE DRIVE 0 FROM TESTING
2352  004672  000413                         BR      8$              ;CHECK ON THE OTHER DRIVES
2353  004674  006201                 7$:     ASR     R1              ;REPORT THE DRIVES TO BE TESTED
2354  004676  103011                         BCC     8$              ;BR IF DRIVE IS NOT SELECTED
2355  004700  005237  017004                 INC     $ENDCT          ;GIVE THIS DRIVE A PASS
2356  004704  010046                         MOV     R0,-(SP)        ;;SAVE R0 FOR TYPEOUT
2357                                                                 ;;TYPE THE DRIVE NUMBER
2358  004706  104403                         TYPOS                   ;;GO TYPE--OCTAL ASCII
2359  004710    001                         .BYTE   1               ;;TYPE 1 DIGIT(S)
2360  004711    000                         .BYTE   0               ;;SUPPRESS LEADING ZEROS
2361  004712  005701                         TST     R1              ;MORE DRIVES?
```

```
 2362  004714  001404                          BEQ     9$              ;NO--BRANCH
 2363  004716  104401  027377                  TYPE    ,COMMA          ;','
 2364  004722  005200                  8$:     INC     R0              ;INCREMENT DRIVE NUMBER
 2365  004724  000763                          BR      7$              ;CONTINUE
 2366  004726  013737  017004  016776  9$:     MOV     $ENDCT,$EOPCT   ;SETUP TEST COUNT
 2367  004734  005037  001206                  CLR     CNTRLC          ;CLEAR CONVERSATION MODE FLAG
 2368  004740  005037  001176          RSTRT1: CLR     CHKDRV          ;INITIALIZE THE CHECK DRIVE KEY
 2369  004744  012737  000001  001306          MOV     #1,DRVMSK       ;START TO CHECK DESIRED DRIVES
 2370  004752  033737  001306  001312  RSTRT2: BIT     DRVMSK,DRVSEL   ;IS THIS DRIVE SELECTED?
 2371  004760  001011                  RSTRT3: BNE     DRVOK           ;YES--GO CHECK IF DRIVE IS READY FOR TESTING
 2372  004762  005237  001176          RESTART:INC     CHKDRV          ;MOVE  TO NEXT DRIVE NUMBER
 2373  004766  106337  001306                  ASLB    DRVMSK          ;POSITION THE MASK
 2374  004772  103367                          BCC     RSTRT2          ;BR IF MORE DRIVES
 2375  004774  043737  001310  001312          BIC     DRVBAD,DRVSEL   ;CLEAR SELECTION BITS FOR ANY OFFLINE/UNSAFE
 2376                                                                  ;DRIVES
 2377  005002  000756                          BR      RSTRT1          ;CONTINUE WITH CYCLE
 2378  005004  113737  001176  002500  DRVOK:  MOVB    CHKDRV,DPB      ;PICKUP THE DRIVE NUMBER
 2379  005012  104401  027401                  TYPE    ,TSTING         ;'TESTING WITH DRIVE '
 2380  005016  013746  001176                  MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
 2381                                                                  ;;TYPE THE DRIVE NUMBER
 2382  005022  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 2383  005024     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
 2384  005025     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 2385  005026  104401  001173                  TYPE    ,$CRLF          ;CR-LF
 2386  005032  013704  001324                  MOV     RPADR,R4        ;RP11 ADDRESS
 2387  005036  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
 2388  005042  112737  000015  002502          MOVB    #HOMSEK,DPB+2   ;HOME SEEK COMMAND
 2389  005050  004737  022770                  JSR     PC,RP11         ;START THE COMMAND
 2390  005054  004737  023546                  JSR     PC,DRVRDY       ;WAIT FOR THE DRIVE TO FINISH
 2391  005060  000137  005064                  JMP     TST0            ;START THE PROGRAM
 2392
 2393                                  .SBTTL  #### TESTS ####
 2394
 2395
 2396                                  ;;**********************************************************************
 2397                                  ;*TEST 0        TEST 'CLEAR' TERMINATION
 2398
 2399                                  ;*VERIFY THAT 'CLEAR' TERMINATES AN OPERATION AND CAUSES CONTROLLER
 2400                                  ;*READY TO SET.
 2401
 2402                                  ;;**********************************************************************
 2403  005064                  TST0:
 2404  005064  033737  001400  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
 2405  005072  001002                          BNE     .+6             ;BR IF YES
 2406  005074  000137  005310                  JMP     TST1            ;GO TO THE NEXT TEST
 2407  005100  012737  005160  001110          MOV     #TEST0,$LPERR   ;SETUP THE ERROR LOOP ADDRESS
 2408  005106  012737  005064  001106          MOV     #TST0,$LPADR    ;SETUP THE SCOPE LOOP ADDRESS
 2409  005114  012737  000000  001102          MOV     #0,$TSTNM       ;SETUP TEST NUMBER AND
 2410                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
 2411  005122  013777  001102  174012          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
 2412  005130  012737  005306  001200          MOV     #EXIT0,BYPASS  ;SETUP BYPASS ADDRESS
 2413  005136  013704  001324                  MOV     RPADR,R4        ;RP11E BUS ADDRESS
 2414  005142  105037  002501                  CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
 2415  005146  105037  002503                  CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
 2416  005152  012737  000144  001162          MOV     #100.,$TIMES    ;;DO 100. ITERATIONS
 2417  005160  012706  001100          TEST0:  MOV     #STACK,SP       ;SETUP THE STACK POINTER
```

```
2418  005164  004737  025654                  JSR     PC,CLRP          ;CLEAR THE RP11
2419  005170  012737  043656  002506          MOV     #BUFFER,$BUF     ;LOAD BUFFER ADDRESS
2420  005176  012737  000001  002504          MOV     #1,$WC           ;SETUP WORD COUNT FOR 1 WORD
2421  005204  005037  002510                  CLR     $CYL             ;CYLINDER 0
2422  005210  005037  002512                  CLR     $SEC             ;TRACK & SECTOR 0
2423  005214  112737  000005  002502          MOVB    #RDSEK,DPB+2     ;READ COMMAND
2424  005222  004737  022770                  JSR     PC,RP11          ;ISSUE THE COMMAND AND RETURN
2425  005226  032764  000200  000004          BIT     #RDY,RPCS(R4)    ;IS THE CONTROLLER READY ?
2426  005234  001404                          BEQ     1$               ;BR IF NOT
2427  005236  004737  023240                  JSR     PC,SAVRP         ;STORE THE REGISTERS
2428  005242  104002                          ERROR   2                ;CAN'T START READ COMMAND
2429  005244  000420                          BR      EXIT0            ;EXIT FROM TEST
2430  005246  012764  000001  000004  1$:     MOV     #CLEAR,RPCS(R4)  ;ISSUE THE CONTROLLER CLEAR
2431  005254  012764  000001  000004          MOV     #CLEAR,RPCS(R4)   ;CLEAR THE CONTROLLER AGAIN
2432  005262  004737  023240                  JSR     PC,SAVRP         ;STORE THE REGISTERS
2433  005266  032737  000200  001340          BIT     #RDY,$RPCS       ;IS THE CONTROLLER READY ?
2434  005274  001004                          BNE     EXIT0            ;BR IF IT IS
2435  005276  104003                          ERROR   3                ;'READY' DIDN'T SET AFTER ISSUING 'CLEAR'
2436                                                                   ;DURING READ COMMAND
2437  005300  000005                          RESET                    ;FORCE THE CONTROLLER READY
2438  005302  004737  020416                  JSR     PC,$TKINT        ;SETUP THE TTY KEYBOARD
2439  005306  000004                  EXIT0:  SCOPE                    ;LOOP ?
2440
2441
2442                                   ;;*****************************************************************
2443                                   ;*TEST 1        WRITE CHECK TEST
2444
2445                                   ;*THIS TEST VERIFIES THE WRITE CHECK LOGIC AND TO VERIFY THAT A WRITE
2446                                   ;*CHECK ERROR CAN BE DETECTED.  'FLOATING ONES' AND 'FLOATING ZEROS'
2447                                   ;*PATTERNS ARE USED TO TEST THE WRITE CHECK COMPARE LOGIC.
2448
2449                                   ;;*****************************************************************
2450  005310                           TST1:
2451  005310  033737  001402  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
2452  005316  001002                          BNE     .+6              ;BR IF YES
2453  005320  000137  005724                  JMP     TST2             ;GO TO THE NEXT TEST
2454  005324  012737  005404  001110          MOV     #TEST1,$LPERR    ;SETUP THE ERROR LOOP ADDRESS
2455  005332  012737  005310  001106          MOV     #TST1,$LPADR     ;SETUP THE SCOPE LOOP ADDRESS
2456  005340  012737  000001  001102          MOV     #1,$TSTNM        ;SETUP TEST NUMBER AND
2457                                                                   ;CLEAR THE ERROR FLAG ($ERFLG)
2458  005346  013777  001102  173566          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2459  005354  012737  005722  001200          MOV     #EXIT1,BYPASS  ;SETUP BYPASS ADDRESS
2460  005362  013704  001324                  MOV     RPADR,R4         ;RP11E BUS ADDRESS
2461  005366  105037  002501                  CLRB    DPB+1            ;CLEAR ANY EXTENDED MEMORY BITS
2462  005372  105037  002503                  CLRB    DPB+3            ;CLEAR 'MODE' & 'HDR' BITS
2463  005376  012737  000144  001162          MOV     #100.,$TIMES     ;;DO 100. ITERATIONS
2464  005404  012706  001100          TEST1:  MOV     #STACK,SP        ;SETUP THE STACK POINTER
2465  005410  005037  002512                  CLR     $SEC             ;CLEAR TRACK & SECTOR ADDRESS
2466  005414  005037  002510                  CLR     $CYL             ;CLEAR CYLINDER ADDRESS
2467  005420  012737  000400  002504          MOV     #256.,$WC        ;SET WORD COUNT TO 1 SECTOR
2468  005426  012737  000011  001204          MOV     #9.,PATNUM       ;STARTING PATTERN NUMBER
2469  005434  004737  024432                  JSR     PC,SETBUF        ;FILL THE BUFFER WITH THE PATTERN
2470  005440  012737  043656  002506          MOV     #BUFFER,$BUF     ;SETUP OUTPUT BUFFER
2471  005446  012737  005446  001110  1$:     MOV     #1$,$LPERR       ;SETUP LOOP ON ERROR ADDRESS
2472  005454  004737  025654                  JSR     PC,CLRP          ;CLEAR THE RP11
2473  005460  012737  000003  002502          MOV     #WRTSEK,DPB+2    ;LOAD COMMAND
```

```
2474  005466  004737  022770                    JSR     PC,RP11          ;WRITE THE SECTOR WITH THE TEST PATTERN
2475  005472  004737  023664                    JSR     PC,CONRDY        ;WAIT FOR THE OPERATION TO COMPLETE
2476  005476  004737  023240                    JSR     PC,SAVRP         ;SAVE THE REGISTERS
2477  005502  032737  100000  001340            BIT     #ERR,$RPCS       ;DID AN ERROR OCCUR ?
2478  005510  001401                            BEQ     3$               ;BR IF NOT
2479  005512  104004                            ERROR   4                ;ERROR OCCURED WRITING WRITE CHECK TEST PATTERN
2480  005514  012737  005514  001110  3$:       MOV     #3$,$LPERR       ;CHANGE LOOP ON ERROR ADDRESS
2481  005522  004737  025654                    JSR     PC,CLRP          ;CLEAR THE RP11
2482  005526  012737  000007  002502            MOV     #WRTCHK,DPB+2    ;LOAD WRITE CHECK COMMAND
2483  005534  004737  022770                    JSR     PC,RP11          ;CHECK THE SECTOR JUST WRITTEN
2484  005540  004737  023664                    JSR     PC,CONRDY        ;WAIT FOR WRITE CHECK TO COMPLETE
2485  005544  004737  023240                    JSR     PC,SAVRP         ;STORE THE REGISTERS
2486  005550  032737  100000  001340            BIT     #ERR,$RPCS       ;CHECK OK ?
2487  005556  001403                            BEQ     4$               ;BR IF OK
2488  005560  104005                            ERROR   5                ;ERROR ATTEMPTING TO CHECK TEST PATTERN
2489  005562  004737  025654                    JSR     PC,CLRP          ;CLEAR THE RP11
2490  005566  005001                  4$:       CLR     R1               ;SETUP TO CLEAR BUFFER
2491  005570  005002                            CLR     R2               ;BUFFER INDEX
2492  005572  023727  001204  000011            CMP     PATNUM,#9.       ;'FLOATING 1' PATTERN ?
2493  005600  001002                            BNE     5$               ;BRANCH IF NOT
2494  005602  012701  177777                    MOV     #-1,R1           ;FILL BUFFER WITH ONES
2495  005606  010162  043656          5$:       MOV     R1,BUFFER(R2)    ;FILL BUFFER WITH PATTERN IN R1
2496  005612  022702  000400                    CMP     #256.,R2         ;SEE IF FINISHED
2497  005616  001403                            BEQ     6$               ;BR IF FINISHED
2498  005620  062702  000002                    ADD     #2,R2            ;INCREMENT THE INDEX
2499  005624  000770                            BR      5$               ;CONTINUE
2500  005626  012737  005626  001110  6$:       MOV     #6$,$LPERR       ;CHANGE THE LOOP ON ERROR ADDRESS
2501  005634  004737  025654                    JSR     PC,CLRP          ;CLEAR THE RP11
2502  005640  004737  022770                    JSR     PC,RP11          ;CHECK THE SECTOR AGAIN, ERROR SHOULD OCCUR
2503  005644  004737  023664                    JSR     PC,CONRDY        ;WAIT FOR THE WRITE CHECK TO COMPLETE
2504  005650  004737  023240                    JSR     PC,SAVRP         ;STORE THE REGISTERS
2505  005654  032737  000010  001336            BIT     #BIT03,$RPER     ;DID WRITE CHECK ERROR SET?
2506  005662  001002                            BNE     7$               ;BRANCH IF YES
2507  005664  104006                            ERROR   6                ;WRITE CHECK ERROR DID NOT SET
2508  005666  000405                            BR      8$               ;BYPASS REST OF TEST
2509  005670  032737  100000  001340  7$:       BIT     #ERR,$RPCS       ;DID 'ERR' SET ?
2510  005676  001001                            BNE     8$               ;BR IF IT SET
2511  005700  104106                            ERROR   106              ;'ERR' DIDN'T SET WITH 'WCE'
2512  005702  023727  001204  000012  8$:       CMP     PATNUM,#10.      ;FLOATING A ZERO ?
2513  005710  001404                            BEQ     EXIT1            ;BR IF ZERO
2514  005712  012737  000012  001204            MOV     #10.,PATNUM      ;'FLOATING ZERO' PATTERN
2515  005720  000652                            BR      1$               ;DO THE TEST AGAIN
2516  005722  000004                  EXIT1:    SCOPE                    ;LOOP ?
2517
2518
2519                                            ;;*****************************************************************
2520                                            ;*TEST 2        PARTIAL SECTOR WRITE TEST
2521
2522                                            ;*CHECK THE ABILITY OF THE RP11E TO CLEAR THE REMAINDER OF A SECTOR
2523                                            ;*ON A PARTIAL WRITE OPERATION. A SECTOR OF ALL ONES IS WRITTEN AND
2524                                            ;*THEN A TWO WORD WRITE OPERATION IS PERFORMED. THE SECTOR IS THEN
2525                                            ;*READ BACK AND VERIFIED. THE FIRST TWO WORDS SHOULD BE ONES AND
2526                                            ;*THE REST SHOULD BE ZEROS.
2527
2528                                            ;;*****************************************************************
2529  005724                                    TST2:
```

```
2530  005724  033737  001404  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
2531  005732  001002                          BNE     .+6                  ;BR IF YES
2532  005734  000137  006402                  JMP     TST3                 ;GO TO THE NEXT TEST
2533  005740  012737  006020  001110          MOV     #TEST2,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
2534  005746  012737  005724  001106          MOV     #TST2,$LPADR         ;SETUP THE SCOPE LOOP ADDRESS
2535  005754  012737  000002  001102          MOV     #2,$TSTNM            ;SETUP TEST NUMBER AND
2536                                                                       ;CLEAR THE ERROR FLAG ($ERFLG)
2537  005762  013777  001102  173152          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2538  005770  012737  006400  001200          MOV     #EXIT2,BYPASS  ;SETUP BYPASS ADDRESS
2539  005776  013704  001324                  MOV     RPADR,R4             ;RP11E BUS ADDRESS
2540  006002  105037  002501                  CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
2541  006006  105037  002503                  CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
2542  006012  012737  000144  001162          MOV     #100.,$TIMES         ;;DO 100. ITERATIONS
2543  006020  012706  001100          TEST2:  MOV     #STACK,SP            ;SETUP THE STACK POINTER
2544  006024  012737  006020  001110          MOV     #TEST2,$LPERR        ;SETUP INITIAL LOOP ON ERROR ADDRESS
2545  006032  004737  025654                  JSR     PC,CLRP              ;CLEAR THE RP11
2546  006036  012737  043656  002506          MOV     #BUFFER,$BUF         ;LOAD BUFFER ADDRESS
2547  006044  012737  000400  002504          MOV     #256.,$WC            ;WORD COUNT = 1 SECTOR
2548  006052  005037  002512                  CLR     $SEC                 ;CLEAR SECTOR/TRACK ADDRESS
2549  006056  005037  002510                  CLR     $CYL                 ;CLEAR CYLINDER ADDRESS
2550  006062  012737  000015  001204          MOV     #13.,PATNUM          ;ONE'S PATTERN NUMBER
2551  006070  004737  024432                  JSR     PC,SETBUF            ;LOAD THE PATTERN INTO THE BUFFER
2552  006074  012737  000003  002502          MOV     #WRTSEK,DPB+2        ;WRITE COMMAND
2553  006102  004737  022770                  JSR     PC,RP11              ;WRITE THE SECTOR
2554  006106  004737  023664                  JSR     PC,CONRDY            ;WAIT FOR THE WRITE TO COMPLETE
2555  006112  004737  023240                  JSR     PC,SAVRP             ;STORE THE REGISTERS
2556  006116  032737  100000  001340          BIT     #ERR,$RPCS           ;DID AN ERROR OCCUR ?
2557  006124  001402                          BEQ     1$                   ;BR IF NOT
2558  006126  104004                          ERROR   4                    ;ERROR OCCURED DURING WRITE
2559  006130  000523                          BR      EXIT2                ;BYPASS REST OF TEST
2560  006132  012737  006132  001110  1$:     MOV     #1$,$LPERR           ;CHANGE LOOP ON ERROR ADDRESS
2561  006140  004737  025654                  JSR     PC,CLRP              ;CLEAR THE RP11
2562  006144  004737  024432                  JSR     PC,SETBUF            ;LOAD ONE'S PATTERN INTO THE BUFFER
2563  006150  012737  000002  002504          MOV     #2,$WC               ;2 WORD TRANSFER
2564  006156  112737  000003  002502          MOVB    #WRTSEK,DPB+2        ;CHANGE COMMAND TO WRITE
2565  006164  004737  022770                  JSR     PC,RP11              ;WRITE
2566  006170  004737  023664                  JSR     PC,CONRDY            ;WAIT FOR CONTROLLER READY
2567  006174  004737  023240                  JSR     PC,SAVRP             ;STORE THE REGISTERS
2568  006200  032737  100000  001340          BIT     #ERR,$RPCS           ;ERROR ?
2569  006206  001402                          BEQ     2$                   ;BR IF NOT
2570  006210  104007                          ERROR   7                    ;ERROR WRITING THE PARTIAL SECTOR
2571  006212  000472                          BR      EXIT2                ;BYPASS THE REST OF THE TEST
2572  006214  012737  006214  001110  2$:     MOV     #2$,$LPERR           ;CHANGE THE ERROR LOOP ADDRESS
2573  006222  004737  025654                  JSR     PC,CLRP              ;CLEAR THE RP11
2574  006226  012737  000400  002504          MOV     #256.,$WC            ;CHANGE THE WORD COUNT
2575  006234  012737  000005  002502          MOV     #RDSEK,DPB+2         ;READ COMMAND
2576  006242  004737  022770                  JSR     PC,RP11              ;READ THE SECTOR
2577  006246  004737  023664                  JSR     PC,CONRDY            ;WAIT FOR CONTROLLER READY
2578  006252  004737  023240                  JSR     PC,SAVRP             ;STORE THE REGISTERS
2579  006256  032737  100000  001340          BIT     #ERR,$RPCS           ;DID AN ERROR OCCUR ?
2580  006264  001402                          BEQ     3$                   ;BR IF NOT
2581  006266  104017                          ERROR   17                   ;ERROR OCCURED READING SECTOR
2582  006270  000443                          BR      EXIT2                ;BYPASS REST OF TEST
2583  006272  022737  177777  043656  3$:     CMP     #177777,BUFFER   ;COMPARE FIRST WORD SHOULD BE ONES
2584  006300  001004                          BNE     4$                   ;BR IF NOT
2585  006302  022737  177777  043660          CMP     #177777,BUFFER+2  ;CHECK THE SECOND WORD
```

```
2586  006310  001407                          BEQ     5$                ;BR IF OK
2587  006312  012737  177777  001124  4$:     MOV     #177777,$GDDAT    ;EXPECTED DATA
2588  006320  013737  043656  001126          MOV     BUFFER,$BDDAT     ;ACTUAL DATA
2589  006326  104011                          ERROR   11                ;DATA COMPARE ERROR IN FIRST 2 WORDS
2590  006330  012700  043662          5$:     MOV     #BUFFER+4,R0      ;STARTING ADDRESS OF LAST PART OF THE BUFFER
2591  006334  012701  000376                  MOV     #254.,R1          ;WORD COUNT
2592  006340  005720                  6$:     TST     (R0)+             ;REMAINDER OF SECTOR SHOULD BE ZEROS
2593  006342  001003                          BNE     7$                ;BRANCH IF NOT
2594  006344  005301                          DEC     R1                ;DECREMENT THE CUNT
2595  006346  001374                          BNE     6$                ;BR IF NOT FINISHED
2596  006350  000413                          BR      EXIT2             ;EXIT
2597  006352  016037  177776  001126  7$:     MOV     -2(R0),$BDDAT     ;INCORRECT DATA
2598  006360  005037  001124                  CLR     $GDDAT            ;EXPECTED DATA (ZEROS)
2599  006364  010037  001122                  MOV     R0,$BDADR         ;ADDRESS OF INCORRECT CHARACTER
2600  006370  162737  000002  001122          SUB     #2,$BDADR         ;DECREMENT THE ADDRESS
2601  006376  104012                          ERROR   12                ;DATA FOUND IN AREA OF SECTOR
2602                                                                    ;WHICH SHOULD HAVE BEEN CLEARED
2603                                                                    ;BY A ONE WORD WRITE
2604  006400  000004                  EXIT2:  SCOPE
2605
2606
2607                                  ;;*****************************************************************
2608                                  ;*TEST 3        'EOP' TEST
2609
2610                                  ;*TEST THAT 'EOP' SETS WHEN THE CONTROLLER TRIES TO WRITE BEYOND THE
2611                                  ;*LIMITS OF THE DRIVE.  THE FIRST SECTOR OF THE PACK IS WRITTEN WITH
2612                                  ;*ZEROS; THEN A TWO SECTOR WRITE OF ALL ONE'S IS ISSUED FOR THE MAXIMUM
2613                                  ;*CYLINDER, HEAD 19, SECTOR 9.  'EOP' AND THE ERROR BITS IN 'RPCS' SHOULD
2614                                  ;*BE SET.  THE FIRST SECTOR OF THE PACK IS CHECKED TO MAKE SURE THAT IS
2615                                  ;*STILL CONTAINS ZEROS.
2616
2617                                  ;;*****************************************************************
2618  006402                          TST3:
2619  006402  033737  001406  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
2620  006410  001002                          BNE     .+6               ;BR IF YES
2621  006412  000137  007060                  JMP     TST4              ;GO TO THE NEXT TEST
2622  006416  012737  006476  001110          MOV     #TEST3,$LPERR     ;SETUP THE ERROR LOOP ADDRESS
2623  006424  012737  006402  001106          MOV     #TST3,$LPADR      ;SETUP THE SCOPE LOOP ADDRESS
2624  006432  012737  000003  001102          MOV     #3,$TSTNM         ;SETUP TEST NUMBER AND
2625                                                                    ;CLEAR THE ERROR FLAG ($ERFLG)
2626  006440  013777  001102  172474          MOV     $TSTNM,@DISPLAY   ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2627  006446  012737  007056  001200          MOV     #EXIT3,BYPASS  ;SETUP BYPASS ADDRESS
2628  006454  013704  001324                  MOV     RPADR,R4          ;RP11E BUS ADDRESS
2629  006460  105037  002501                  CLRB    DPB+1             ;CLEAR ANY EXTENDED MEMORY BITS
2630  006464  105037  002503                  CLRB    DPB+3             ;CLEAR 'MODE' & 'HDR' BITS
2631  006470  012737  000031  001162          MOV     #25.,$TIMES       ;;DO 25. ITERATIONS
2632  006476  012706  001100          TEST3:  MOV     #STACK,SP         ;SETUP THE STACK POINTER
2633  006502  012737  006476  001110          MOV     #TEST3,$LPERR     ;INITIAL LOOP ON ERRORR ADDRESS
2634  006510  004737  025654                  JSR     PC,CLRP           ;CLEAR THE RP11
2635  006514  012737  000400  002504          MOV     #256.,$WC         ;SET WORD COUNT TO 1 SECTOR
2636  006522  012737  043656  002506          MOV     #BUFFER,$BUF      ;SETUP OUTPUT BUFFER
2637  006530  005037  001204                  CLR     PATNUM            ;SETUP FOR PATTERN ZERO
2638  006534  004737  024432                  JSR     PC,SETBUF         ;LOAD ZEROS INTO THE BUFFER
2639  006540  005037  002512                  CLR     $SEC              ;SET SECTOR & TRACK TO ZERO
2640  006544  005037  002510                  CLR     $CYL              ;SET CYLINDER TO ZERO
2641  006550  112737  000003  002502          MOVB    #WRTSEK,DPB+2     ;WRITE COMMAND
```

```
 2642  006556  004737  022770                    JSR     PC,RP11         ;WRITE ZEROS IN C0, T0, S0
 2643  006562  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
 2644  006566  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
 2645  006572  032737  100000  001340            BIT     #ERR,$RPCS      ;DID AN ERROR OCCUR ?
 2646  006600  001402                            BEQ     1$              ;BR IF NOT
 2647  006602  104004                            ERROR   4               ;ERROR OCCURED DURING SETUP
 2648  006604  000524                            BR      EXIT3           ;BYPASS REST OF TEST
 2649  006606  012737  006606  001110  1$:       MOV     #1$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
 2650  006614  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
 2651  006620  012737  000015  001204            MOV     #13.,PATNUM     ;ONE'S PATTERN INDEX
 2652  006626  004737  024432                    JSR     PC,SETBUF       ;LOAD THE PATTERN INTO THE BUFFER
 2653  006632  012737  001000  002504            MOV     #512.,$WC       ;CHANGE WORD COUNT TO 2 SECTORS
 2654  006640  013737  001210  002510            MOV     MAXCYL,$CYL     ;SELECT THE DRIVE'S MAXIMUM CYLINDER
 2655  006646  012737  000011  002512            MOV     #9.,$SEC        ;SELECT SECTOR 9
 2656  006654  112737  000023  002513            MOVB    #19.,$TRK       ;SELECT HEAD 19
 2657  006662  004737  022770                    JSR     PC,RP11         ;START THE READ
 2658  006666  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
 2659  006672  004737  023240                    JSR     PC,SAVRP        ;STORE THE RP11 REGISTERS
 2660  006676  032737  000002  001336            BIT     #BIT01,$RPER    ;DID EOP ERROR SET?
 2661  006704  001002                            BNE     2$              ;BRANCH IF SET
 2662  006706  104013                            ERROR   13              ;'EOP' DIDN'T SET
 2663  006710  000405                            BR      3$              ;CHECK RPCA
 2664  006712  032737  100000  001340  2$:       BIT     #ERR,$RPCS      ;DID THE ERROR FLAG SET?
 2665  006720  001001                            BNE     3$              ;BRANCH IF SET
 2666  006722  104014                            ERROR   14              ;ERROR DID NOT SET AFTER GENERATING 'EOP'
 2667  006724  005737  001346            3$:     TST     $RPCA           ;SEE IF THE CYLINDER ADDRESS REGISTER WAS CLEARED
 2668  006730  001401                            BEQ     4$              ;BR IF CLEARED
 2669  006732  104015                            ERROR   15              ;'RPCA' NOT CLEARED BY 'EOP'
 2670  006734  023737  001354  001210  4$:       CMP     $SUCA,MAXCYL    ;SUCA STILL EQUAL TO THE MAXIMUM CLINDER
 2671  006742  001401                            BEQ     5$              ;BR IF IT IS
 2672  006744  104016                            ERROR   16              ;SUCA NOT CORRECT AFTER EOP ERROR
 2673  006746  012737  006746  001110  5$:       MOV     #5$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
 2674  006754  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
 2675  006760  012737  000002  002504            MOV     #2,$WC          ;WORD COUNT
 2676  006766  005037  002512                    CLR     $SEC            ;CLEAR SECTOR/TRACK ADDRESS
 2677  006772  005037  002510                    CLR     $CYL            ;CLEAR THE CYLINDER ADDRESS
 2678  006776  012737  000005  002502            MOV     #RDSEK,DPB+2    ;CHANGE COMMAND TO READ
 2679  007004  004737  022770                    JSR     PC,RP11         ;EXECUTE THE COMMAND
 2680  007010  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
 2681  007014  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
 2682  007020  032737  100000  001340            BIT     #ERR,$RPCS      ;WERE THERE ANY ERRORS?
 2683  007026  001402                            BEQ     6$              ;BRANCH IF NOT
 2684  007030  104017                            ERROR   17              ;ERROR ENCOUNTERED ON 2 WORD READ
 2685  007032  000411                            BR      EXIT3           ;OF FIRST SECTOR ON THE PACK
 2686  007034  013737  043656  001126  6$:       MOV     BUFFER,$BDDAT   ;GET FIRST WORD OF BUFFER
 2687  007042  005737  001126                    TST     $BDDAT          ;DOES 1ST SECTOR STILL CONTAIN ZEROS?
 2688  007046  001403                            BEQ     EXIT3           ;BRANCH IF YES
 2689  007050  005037  001124                    CLR     $GDDAT          ;ZEROS WERE EXPECTED
 2690  007054  104020                            ERROR   20              ;CONTENTS OF THE FIRST SECTOR OF THE
 2691                                                                    ;PACK CHANGED AFTER FORCING EOP
 2692  007056  000004                    EXIT3:  SCOPE
 2693
 2694
 2695                                    ;;**************************************************************
 2696                                    ;*TEST 4         TEST 'PROG' ERROR
 2697
```

```
                                        ;*VERIFY THAT THE RP11E GENERATES A 'PROG' ERROR IF A COMMAND IS ISSUED
                                        ;*WHILE THE CONTROLLER IS BUSY.
                                        ;
                                        ;;*****************************************************************
007060                                  TST4:
007060  033737  001410  001276                 BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
007066  001002                               BNE     .+6                  ;BR IF YES
007070  000137  007310                       JMP     TST5                 ;GO TO THE NEXT TEST
007074  012737  007154  001110               MOV     #TEST4,$LPERR        ;SETUP THE ERROR LOOP ADDRESS
007102  012737  007060  001106               MOV     #TST4,$LPADR         ;SETUP THE SCOPE LOOP ADDRESS
007110  012737  000004  001102               MOV     #4,$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                                  TST5:
007310  033737  001412  001276                 BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
```

```
 2754  007316  001002                          BNE     .+6             ;BR IF YES
 2755  007320  000137  010034                  JMP     TST6            ;GO TO THE NEXT TEST
 2756  007324  012737  007404  001110          MOV     #TEST5,$LPERR   ;SETUP THE ERROR LOOP ADDRESS
 2757  007332  012737  007310  001106          MOV     #TST5,$LPADR    ;SETUP THE SCOPE LOOP ADDRESS
 2758  007340  012737  000005  001102          MOV     #5,$TSTNM       ;SETUP TEST NUMBER AND
 2759                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
 2760  007346  013777  001102  171566          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
 2761  007354  012737  010032  001200          MOV     #EXIT5,BYPASS  ;SETUP BYPASS ADDRESS
 2762  007362  013704  001324                  MOV     RPADR,R4        ;RP11E BUS ADDRESS
 2763  007366  105037  002501                  CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
 2764  007372  105037  002503                  CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
 2765  007376  012737  000144  001162          MOV     #100.,$TIMES    ;;DO 100. ITERATIONS
 2766  007404  012706  001100          TEST5:  MOV     #STACK,SP       ;SETUP THE STACK POINTER
 2767  007410  012737  007404  001110          MOV     #TEST5,$LPERR   ;INITIAL LOOP ON ERROR ADDRESS
 2768  007416  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
 2769  007422  012737  000001  043656          MOV     #1,BUFFER       ;SETUP INVALID HEADER IMAGE
 2770  007430  012737  000001  043660          MOV     #1,BUFFER+2     ;CYLINDER/TRACK
 2771  007436  012737  000001  043662          MOV     #1,BUFFER+4     ;SECTOR
 2772  007444  012737  000003  002504          MOV     #3,$WC          ;LOAD WORD COUNT
 2773  007452  005037  002512                  CLR     $SEC            ;CLEAR SECTOR/TRACK ADDRESS
 2774  007456  005037  002510                  CLR     $CYL            ;CLEAR THE CYLINDER ADDRESS
 2775  007462  012737  014000  002502          MOV     #MODE!HDR,DPB+2 ;'MODE' AND 'HEADER' BITS
 2776  007470  112737  000003  002502          MOVB    #WRTSEK,DPB+2   ;WRITE ORDER
 2777  007476  004737  022770                  JSR     PC,RP11         ;MISFORMAT SECTOR 0
 2778  007502  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR THE OPERATION TO COMPLETE
 2779  007506  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
 2780  007512  032737  100000  001340          BIT     #ERR,$RPCS      ;DID AN ERROR OCCUR ?
 2781  007520  001402                          BEQ     1$              ;BR IF NOT
 2782  007522  104023                          ERROR   23              ;ERROR OCCURED ATTEMPTING TO MISFORMAT SECTOR 0
 2783  007524  000506                          BR      TST5B           ;BYPASS TEST & REFORMAT SECTOR 0
 2784  007526  012737  007526  001110  1$:     MOV     #1$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
 2785  007534  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
 2786  007540  112737  000005  002502          MOVB    #RDSEK,DPB+2    ;CHANGE COMMAND
 2787  007546  004737  022770                  JSR     PC,RP11         ;EXECUTE THE COMMAND
 2788  007552  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
 2789  007556  032737  100000  001340          BIT     #ERR,$RPCS      ;ANY ERRORS?
 2790  007564  001401                          BEQ     2$              ;BRANCH IF NOT
 2791  007566  104024                          ERROR   24              ;ERROR WHILE READING THE HEADER
 2792                                                                  ;ON SECTOR ZERO
 2793  007570  012737  000001  001124  2$:     MOV     #1,$GDDAT       ;EXPECTED RESULT
 2794  007576  005000                          CLR     R0              ;CLEAR THE INDEX
 2795  007600  023760  001124  043656  3$:     CMP     $GDDAT,BUFFER(R0)  ;CHECK DATA READ BACK
 2796  007606  001006                          BNE     4$              ;BRANCH ON NON COMPARE
 2797  007610  062700  000002                  ADD     #2,R0           ;UPDATE MODIFIER
 2798  007614  022700  000006                  CMP     #6,R0           ;END OF BUFFER?
 2799  007620  001367                          BNE     3$              ;BR IF NOT
 2800  007622  000404                          BR      TST5A           ;CONTINUE WITH TEST
 2801  007624  016037  043656  001126  4$:     MOV     BUFFER(R0),$BDDAT  ;GET BAD DATA
 2802  007632  104025                          ERROR   25              ;DATA DID NOT VERIFY AFTER READING
 2803                                                                  ;THE HEADER OF SECTOR ZERO
 2804  007634  012737  007634  001110  TST5A:  MOV     #TST5A,$LPERR   ;CHANGE LOOP ON ERROR ADDRESS
 2805  007642  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
 2806  007646  012737  000003  002502          MOV     #WRTSEK,DPB+2   ;CHANGE COMMAND (AND CLEAR 'MODE' & 'HDR')
 2807  007654  004737  022770                  JSR     PC,RP11         ;EXECUTE THE COMMAND
 2808  007660  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
 2809  007664  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
```

```
2810  007670  032737  010000  001334  1$:     BIT     #BIT12,$RPDS    ;DID HEADER NOT FOUND SET?
2811  007676  001002                          BNE     2$              ;BRANCH IF YES
2812  007700  104026                          ERROR   26              ;'HNF' DID NOT SET
2813  007702  000417                          BR      TST5B           ;BYPASS REST OF TEST
2814  007704  032737  000001  001336  2$:     BIT     #BIT00,$RPER    ;DID DISK ERROR SET?
2815  007712  001001                          BNE     3$              ;BRANCH IF YES
2816  007714  104027                          ERROR   27              ;'DSK' DID NOT SET WITH 'HNF'
2817  007716  032737  040000  001340  3$:     BIT     #HE,$RPCS       ;DID HARD ERROR SET?
2818  007724  001001                          BNE     4$              ;BRANCH IF YES
2819  007726  104030                          ERROR   30              ;HARD ERROR NOT SET AFTER 'HNF'
2820  007730  032737  100000  001340  4$:     BIT     #ERR,$RPCS      ;DID 'ERR' SET ?
2821  007736  001001                          BNE     TST5B           ;BRANCH IF YES
2822  007740  104031                          ERROR   31              ;'ERR' DID NOT SET WITH 'HNF'
2823  007742  012737  007742  001110  TST5B:  MOV     #TST5B,$LPERR   ;CHANGE LOOP ON ERROR ADDRESS
2824  007750  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
2825  007754  005037  043656                  CLR     BUFFER          ;SETUP HEADER IMAGE
2826  007760  005037  043660                  CLR     BUFFER+2        ;CYLINDER/TRACK = 0
2827  007764  005037  043662                  CLR     BUFFER+4        ;SECTOR = 0
2828  007770  012737  014000  002502          MOV     #MODE!HDR,DPB+2 ;'MODE' AND 'HDR' BITS
2829  007776  112737  000003  002502          MOVB    #WRTSEK,DPB+2   ;WRITE COMMAND
2830  010004  004737  022770                  JSR     PC,RP11         ;RESTORE THE HEADER
2831  010010  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR THE OPERATION TO COMPLETE
2832  010014  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
2833  010020  032737  100000  001340          BIT     #ERR,$RPCS      ;DID AN ERROR OCCUR ?
2834  010026  001557                          BEQ     EXIT6           ;BR IF NOT
2835  010030  104032                          ERROR   32              ;ERROR ATTEMPTING TO RESTORE HEADER
2836  010032  000004                  EXIT5:  SCOPE
2837
2838
2839                                  ;;**************************************************************
2840                                  ;*TEST 6        TEST COMMAND ISSUED TO A SEEKING DRIVE
2841
2842                                  ;*ISSUE A SEEK COMMAND AND WAIT FOR DONE TO SET. THEN ISSUE A READ
2843                                  ;*COMMAND WHILE THE HEADS ARE STILL MOVING. THE RP11E SHOULD
2844                                  ;*HOLD THE WRITE COMMAND TILL THE SEEK IS COMPLETE.
2845
2846                                  ;;**************************************************************
2847  010034                          TST6:
2848  010034  033737  001414  001276          BIT     BITS+<$TN+2>,TSTNMS  ;IS THIS TEST SELECTED
2849  010042  001002                          BNE     .+6             ;BR IF YES
2850  010044  000137  010370                  JMP     TST7            ;GO TO THE NEXT TEST
2851  010050  012737  010130  001110          MOV     #TEST6,$LPERR   ;SETUP THE ERROR LOOP ADDRESS
2852  010056  012737  010034  001106          MOV     #TST6,$LPADR    ;SETUP THE SCOPE LOOP ADDRESS
2853  010064  012737  000006  001102          MOV     #6,$TSTNM       ;SETUP TEST NUMBER AND
2854                                          ;CLEAR THE ERROR FLAG ($ERFLG)
2855  010072  013777  001102  171042          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2856  010100  012737  010366  001200          MOV     #EXIT6,BYPASS  ;SETUP BYPASS ADDRESS
2857  010106  013704  001324                  MOV     RPADR,R4        ;RP11E BUS ADDRESS
2858  010112  105037  002501                  CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
2859  010116  105037  002503                  CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
2860  010122  012737  000031  001162          MOV     #25.,$TIMES     ;;DO 25. ITERATIONS
2861  010130  012706  001100          TEST6:  MOV     #STACK,SP       ;SETUP THE STACK POINTER
2862  010134  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
2863  010140  005037  002512                  CLR     $SEC            ;SECTOR/TRACK 0
2864  010144  005037  002510                  CLR     $CYL            ;CYLINDER 0
2865  010150  112737  000015  002502          MOVB    #HOMSEK,DPB+2   ;HOME SEEK COMMAND
```

```
2866  010156  004737  022770                    JSR     PC,RP11          ;DO THE COMMAND
2867  010162  004737  023546                    JSR     PC,DRVRDY        ;WAIT FOR DRIVE READY
2868  010166  012737  000300  002510            MOV     #192.,$CYL       ;CHANGE CYLINDER TO CYLINDER 192
2869  010174  112737  000011  002502            MOVB    #SEEK,DPB+2      ;SEEK COMMAND
2870  010202  004737  022770                    JSR     PC,RP11          ;INITIATE THE SEEK
2871  010206  012737  000001  001220            MOV     #1,STALLT        ;LOAD STALL VALUE
2872  010214  004737  023202                    JSR     PC,STALL         ;STALL FOR 1 MILLISECONDS
2873  010220  012764  177775  000006            MOV     #-3,RPWC(R4)     ;SET WORD COUNT TO THREE WORDS
2874  010226  012764  043656  000010            MOV     #BUFFER,RPBA(R4)  ;BUFFER ADDRESS
2875  010234  052737  014000  002502            BIS     #MODE!HDR,DPB+2 ;SETUP TO READ THE HEADER
2876  010242  112764  000017  000004            MOVB    #READ,RPCS(R4)   ;TRY TO DO A READ
2877  010250  004737  023546                    JSR     PC,DRVRDY        ;WAIT FOR DRIVE READY
2878  010254  004737  023240                    JSR     PC,SAVRP         ;STORE THE REGISTERS
2879  010260  032737  000200  001340            BIT     #RDY,$RPCS       ;DID CONTROLLER BECOME READY ALSO ?
2880  010266  001402                            BEQ     1$               ;BR IF NOT
2881  010270  104033                            ERROR   33               ;READ DIDN'T START DURING SEEK
2882  010272  000435                            BR      EXIT6            ;EXIT
2883  010274  004737  023664          1$:       JSR     PC,CONRDY        ;WAIT FOR CONTROLLER READY
2884  010300  004737  023240                    JSR     PC,SAVRP         ;STORE THE REGISTERS
2885  010304  032737  100000  001340            BIT     #ERR,$RPCS       ;ANY ERRORS?
2886  010312  001401                            BEQ     2$               ;BRANCH IF NO
2887  010314  104034                            ERROR   34               ;ERROR OCCURED DURING A READ ISSUED
2888                                                                     ;WHILE THE DRIVE IS SEEKING
2889  010316  012737  014000  001124  2$:       MOV     #<192.*32.>,$GDDAT  ;EXPECTED CYLINDER VALUE
2890  010324  013737  043660  001126            MOV     BUFFER+2,$BDDAT ;RECEIVED DATA
2891  010332  023737  001124  001126            CMP     $GDDAT,$BDDAT    ;CORRECT ?
2892  010340  001366                            BNE     2$               ;BR IF NOT
2893  010342  005037  001124                    CLR     $GDDAT           ;EXPECTED VALUE FOR SECTOR
2894  010346  013737  043662  001126            MOV     BUFFER+4,$BDDAT ;RECEIVED SECTOR
2895  010354  023737  001124  001126            CMP     $GDDAT,$BDDAT    ;CORRECT ?
2896  010362  001401                            BEQ     EXIT6            ;BR IF OK
2897  010364  104035                            ERROR   35               ;HEADER DATA INCORRECT
2898  010366  000004                    EXIT6:  SCOPE
2899
2900
2901                                      ;;*****************************************************************
2902                                      ;*TEST 7        TEST 'NXME' BIT
2903
2904                                      ;*THE NON-EXISTENT MEMORY ERROR BIT IS TESTED BY ATTEMPTING TO READ 1
2905                                      ;*WORD INTO LOCATION 760000.  'NXME', 'HE', & 'ERR' SHOULD ALL BE SET.
2906
2907                                      ;;*****************************************************************
2908  010370                              TST7:
2909  010370  033737  001416  001276            BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
2910  010376  001002                            BNE     .+6              ;BR IF YES
2911  010400  000137  010612                    JMP     TST10            ;GO TO THE NEXT TEST
2912  010404  012737  010464  001110            MOV     #TEST7,$LPERR    ;SETUP THE ERROR LOOP ADDRESS
2913  010412  012737  010370  001106            MOV     #TST7,$LPADR     ;SETUP THE SCOPE LOOP ADDRESS
2914  010420  012737  000007  001102            MOV     #7,$TSTNM        ;SETUP TEST NUMBER AND
2915                                                                     ;CLEAR THE ERROR FLAG ($ERFLG)
2916  010426  013777  001102  170506            MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
2917  010434  012737  010610  001200            MOV     #EXIT7,BYPASS   ;SETUP BYPASS ADDRESS
2918  010442  013704  001324                    MOV     RPADR,R4         ;RP11E BUS ADDRESS
2919  010446  105037  002501                    CLRB    DPB+1            ;CLEAR ANY EXTENDED MEMORY BITS
2920  010452  105037  002503                    CLRB    DPB+3            ;CLEAR 'MODE' & 'HDR' BITS
2921  010456  012737  000144  001162            MOV     #100.,$TIMES     ;;DO 100. ITERATIONS
```

```
 2922  010464  012706  001100          TEST7:  MOV     #STACK,SP       ;SETUP THE STACK POINTER
 2923  010470  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
 2924  010474  012737  000001  002504          MOV     #1,$WC          ;WORD COUNT
 2925  010502  005037  002512                  CLR     $SEC            ;SECTOR/TRACK ZERO
 2926  010506  005037  002510                  CLR     $CYL            ;CYLINDER ZERO
 2927  010512  112737  000060  002501          MOVB    #MEX1!MEX0,DPB+1  ;LOAD BOTH EXTENDED MEMORY BITS
 2928  010520  012737  160000  002506          MOV     #160000,$BUF    ;LOAD ADDR TO FORCE NON EX MEMORY
 2929  010526  112737  000005  002502          MOVB    #RDSEK,DPB+2    ;TRY TO READ INTO THE LOCATION
 2930  010534  004737  022770                  JSR     PC,RP11         ;INITIATE THE OPERATION
 2931  010540  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
 2932  010544  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
 2933  010550  032737  000004  001336          BIT     #BIT02,$RPER    ;DID NON EX MEMORY SET ?
 2934  010556  001002                          BNE     2$              ;BRANCH IF SET
 2935  010560  104036                          ERROR   36              ;'NXME' ERROR DIDN'T SET
 2936  010562  000412                          BR      EXIT7           ;DON'T CHECK THE RPCS BITS
 2937  010564  032737  040000  001340  2$:     BIT     #HE,$RPCS       ;DID 'HE' SET ON NON EX MEMORY
 2938  010572  001001                          BNE     3$              ;BRANCH IF SET
 2939  010574  104037                          ERROR   37              ;'HE' DIDN'T SET
 2940  010576  032737  100000  001340  3$:     BIT     #ERR,$RPCS      ;DID 'ERR' SET WITH NON EX MEMORY
 2941  010604  001001                          BNE     EXIT7           ;BRANCH IF SET
 2942  010606  104040                          ERROR   40              ;'ERR' DIDN'T SET
 2943  010610  000004                  EXIT7:  SCOPE
 2944
 2945
 2946                                  ;;**********************************************************************
 2947                                  ;*TEST 10       SECTOR ADDRESSING TEST
 2948
 2949                                  ;*THE SECTOR ADDRESSING TEST TESTS THE CAPABILTIY OF THE RP11E TO LOCATE
 2950                                  ;*A SECTOR BY USING THE 'SOT' COUNTER (FOR 'HDR' OPERATIONS) AND TO LOCATE
 2951                                  ;*A SECTOR BY COMPARING THE HEADER CONTENTS AGAINST THE CYLINDER AND TRACK/
 2952                                  ;*SECTOR ADDRESS REGISTERS (NORMAL MODE).  THE TEST IS PERFORMED IN 3 PARTS.
 2953                                  ;*
 2954                                  ;*      1.  WRITE ALL HEADERS ON TRACK 0, CYLINDER 0 IN ASCENDING SEQUENCE
 2955                                  ;*      FROM INDEX (0, 1, 2 ... 9).
 2956                                  ;*
 2957                                  ;*      2.  READ EACH HEADER, BEGINNING WITH SECTOR 0, AND VERIFY THAT THE
 2958                                  ;*      SECTOR FIELD IN THE HEADER IS CORRECT FOR THE HEADER EXPECTED.
 2959                                  ;*
 2960                                  ;*      VERIFY THAT THE 'SOT' COUNTER AND THE SECTOR ADDRESS IN 'RPDA' ARE
 2961                                  ;*      EQUAL AND ARE THE EXPECTED VALUE.  THE EXPECTED VALUE IS 1 GREATER
 2962                                  ;*      THAN THE SECTOR READ.  THE SECTOR ADDRESS REGISTER AND THE 'SOT'
 2963                                  ;*      WILL BE INCREMENTED BY THE TIME THE PROGRAM IS ABLE TO READ THE
 2964                                  ;*      REGISTER.
 2965                                  ;*
 2966                                  ;*      WRITE THE SECTOR'S ADDRESS IN THE DATA FIELD.  (FOR SECTOR 0, AN
 2967                                  ;*      OCTAL 12 IS WRITTEN.)
 2968                                  ;*
 2969                                  ;*      CONTINUE THIS SEQUENCE FOR THE REMAINING SECTORS ON THE TRACK.
 2970                                  ;*
 2971                                  ;*      3.  READ THE DATA FROM EACH SECTOR USING A NORMAL (I.E., NON-HEADER)
 2972                                  ;*      READ AND VERIFY THAT THE DATA IS CORRECT FOR THE EXPECTED SECTOR.
 2973
 2974                                  ;;**********************************************************************
 2975  010612                          TST10:
 2976  010612  033737  001420  001276          BIT     BITS+<$TN+2>,TSTNMS  ;IS THIS TEST SELECTED
 2977  010620  001002                          BNE     .+6             ;BR IF YES
```

```
 2978  010622  000137  011532                    JMP     TST11           ;GO TO THE NEXT TEST
 2979  010626  012737  010706  001110            MOV     #TEST10,$LPERR  ;SETUP THE ERROR LOOP ADDRESS
 2980  010634  012737  010612  001106            MOV     #TST10,$LPADR   ;SETUP THE SCOPE LOOP ADDRESS
 2981  010642  012737  000010  001102            MOV     #10,$TSTNM      ;SETUP TEST NUMBER AND
 2982                                                                    ;CLEAR THE ERROR FLAG ($ERFLG)
 2983  010650  013777  001102  170264            MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
 2984  010656  012737  011530  001200            MOV     #EXIT10,BYPASS  ;SETUP BYPASS ADDRESS
 2985  010664  013704  001324                    MOV     RPADR,R4        ;RP11E BUS ADDRESS
 2986  010670  105037  002501                    CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
 2987  010674  105037  002503                    CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
 2988  010700  012737  000031  001162            MOV     #25.,$TIMES     ;;DO 25. ITERATIONS
 2989  010706  012706  001100          TEST10:   MOV     #STACK,SP       ;RESET THE STACK POINTER
 2990  010712  012737  010706  001110            MOV     #TEST10,$LPERR  ;LOOP ON ERROR ADDRESS
 2991  010720  004737  025654                    JSR     PC CLRP         ;CLEAR THE RP11
 2992  010724  012737  043656  002506            MOV     #BUFFER,$BUF    ;SETUP BUFFER ADDRESS
 2993  010732  012737  000036  002504            MOV     #30.,$WC        ;SETUP WORD COUNT FOR 10 SECTORS
 2994  010740  005037  002510                    CLR     $CYL            ;SETUP FOR CYLINDER 0
 2995  010744  005037  002512                    CLR     $SEC            ;TRACK/SECTOR 0
 2996  010750  004737  024304                    JSR     PC,LODSEC       ;SETUP FORMAT BUFFER
 2997  010754  012737  014000  002502            MOV     #MODE!HDR,DPB+2 ;SETUP THE HEADER CONTROL BITS
 2998  010762  112737  000003  002502            MOVB    #WRTSEK,DPB+2   ;THE COMMAND CODE
 2999  010770  004737  022770                    JSR     PC,RP11         ;FORMAT THE TRACK
 3000  010774  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
 3001  011000  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
 3002  011004  032737  100000  001340            BIT     #ERR,$RPCS      ;ERROR ?
 3003  011012  001403                            BEQ     1$              ;BR IF NOT
 3004  011014  104041                            ERROR   41              ;ERROR TRYING TO FORMAT THE TRACK
 3005  011016  000137  011530                    JMP     EXIT10          ;BYPASS REST OF THE TEST
 3006  011022  105037  002512          1$:       CLRB    $SEC            ;START CHECK WITH SECTOR 0
 3007  011026  012737  011026  001110  2$:       MOV     #2$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
 3008  011034  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
 3009  011040  012737  014000  002502            MOV     #MODE'HDR,DPB+2 ;SET 'MODE' AND 'HDR' BITS
 3010  011046  112737  000005  002502            MOVB    #RDSEK,DPB+2    ;CHANGE THE OPERATION TO A READ
 3011  011054  012737  000003  002504            MOV     #3,$WC          ;CHANGE THE WORD COUNT TO 3
 3012  011062  004737  022770                    JSR     PC,RP11         ;READ THE SECTOR'S HEADER
 3013  011066  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
 3014  011072  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
 3015  011076  012737  000000  001220            MOV     #0,STALLT       ;LOAD STALL VALUE
 3016  011104  004737  023202                    JSR     PC,STALL        ;STALL FOR 0 MILLISECONDS
 3017  011110  016437  000014  001350            MOV     RPDA(R4),$RPDA  ;GET THE CONTENTS OF THE ADDRESS REGISTER
 3018  011116  032737  100000  001340            BIT     #ERR,$RPCS      ;ERROR ?
 3019  011124  001402                            BEQ     3$              ;BR IF NOT
 3020  011126  104042                            ERROR   42              ;ERROR READING THE HEADER
 3021  011130  000457                            BR      6$              ;BYPASS DATA CHECK
 3022  011132  005037  001124          3$:       CLR     $GDDAT          ;CLEAR FOR EXPECTED DATA
 3023  011136  113737  002512  001124            MOVB    $SEC,$GDDAT     ;MOVE SECTOR ADDRESS
 3024  011144  023737  001124  043662            CMP     $GDDAT,BUFFER+4 ;CHECK THE SECTOR FIELD
 3025  011152  001401                            BEQ     4$              ;BR IF CORRECT
 3026  011154  104043                            ERROR   43              ;SECTOR FIELD FROM HEADER NOT CORRECT
 3027  011156  005237  001124          4$:       INC     $GDDAT          ;INCREMENT SECTOR ADDRESS FOR 'SOT' CHECK
 3028  011162  023727  001124  000012            CMP     $GDDAT,#10.     ;EXCEED THE MAXIMUM ?
 3029  011170  103402                            BLO     .+6             ;BR IF NOT
 3030  011172  005037  001124                    CLR     $GDDAT          ;RESET COMPARISON VALUE
 3031  011176  013737  001350  001126            MOV     $RPDA,$BDDAT    ;CHECK THE 'SOT'
 3032  011204  042737  177417  001126            BIC     # C360,$BDDAT   ;CLEAR THE OTHER BITS
 3033  011212  006237  001126                    ASR     $BDDAT          ;SHIFT THE 'SOT'
```

```
3034  011216  006237  001126              ASR     $BDDAT          ;SHIFT THE 'SOT'
3035  011222  006237  001126              ASR     $BDDAT          ;SHIFT THE 'SOT'
3036  011226  006237  001126              ASR     $BDDAT          ;SHIFT THE 'SOT'
3037  011232  013737  001350  001122      MOV     $RPDA,$BDADR    ;CHECK THE SECTOR ADDRESS
3038  011240  042737  177760  001122      BIC     # C17,$BDADR    ;LEAVE THE SECTOR ADDRESS
3039  011246  023737  001124  001126      CMP     $GDDAT,$BDDAT   ;IS THE 'SOT' CORRECT ?
3040  011254  001004                      BNE     5$              ;BR IF NOT
3041  011256  023737  001126  001122      CMP     $BDDAT,$BDADR   ;DOES THE SECTOR ADDRESS EQUAT THE 'SOT' ?
3042  011264  001401                      BEQ     6$              ;BR IF IT DOES
3043  011266  104045              5$:     ERROR   45              ;'SOT' OR SECTOR ADDRESS IS NOT CORRECT
3044  011270  012737  011270  001110  6$: MOV     #6$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
3045  011276  004737  025654              JSR     PC,CLRP         ;CLEAR THE RP11
3046  011302  105037  002503              CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
3047  011306  004737  024336              JSR     PC,FILSEC       ;SETUP TO WRITE SECTOR ADDRESS AS DATA
3048  011312  112737  000003  002502      MOVB    #WRTSEK,DPB+2   ;CHANGE OPERATION
3049  011320  004737  022770              JSR     PC,RP11         ;WRITE THE SECTOR
3050  011324  004737  023664              JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
3051  011330  004737  023240              JSR     PC,SAVRP        ;STORE THE REGISTERS
3052  011334  032737  100000  001340      BIT     #ERR,$RPCS      ;ERROR ?
3053  011342  001401                      BEQ     7$              ;BR IF NOT
3054  011344  104044                      ERROR   44              ;ERROR TRYING TO WRITE SECTOR ADDRESS AS DATA
3055  011346  112737  000005  002502  7$: MOVB    #RDSEK,DPB+2    ;CHANGE THE OPERATION
3056  011354  105237  002512              INCB    $SEC            ;INCREMENT THE SECTOR
3057  011360  123727  002512  000012      CMPB    $SEC,#10.       ;AT MAXIMUM SECTOR ADDRESS ?
3058  011366  001217                      BNE     2$              ;BR IF NOT
3059  011370  012737  011424  001110      MOV     #8$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
3060  011376  012737  000400  002504      MOV     #256.,$WC       ;CHANGE WORD COUNT
3061  011404  105037  002512              CLRB    $SEC            ;START WITH SECTOR ZERO
3062  011410  012737  000012  001124      MOV     #10.,$GDDAT     ;SECTOR ZERO PATTERN
3063  011416  112737  000005  002502      MOVB    #RDSEK,DPB+2    ;CHANGE OPERATION CODE
3064  011424  004737  025654          8$: JSR     PC,CLRP         ;CLEAR THE RP11
3065  011430  004737  022770              JSR     PC,RP11         ;READ THE SECTOR
3066  011434  004737  023664              JSR     PC,CONRDY       ;WAIT FOR THE ORDER TO COMPLETE
3067  011440  004737  023240              JSR     PC,SAVRP        ;STORE THE REGISTERS
3068  011444  032737  100000  001340      BIT     #ERR,$RPCS      ;ERROR ?
3069  011452  001402                      BEQ     9$              ;BR IF NOT
3070  011454  104017                      ERROR   17              ;ERROR TRYING TO READ THE SECTOR
3071  011456  000412                      BR      10$             ;BYPASS DATA CHECK
3072  011460  005037  001120          9$: CLR     $GDADR          ;CLEAR FOR EXPECTED DATA
3073  011464  113737  002512  001120      MOVB    $SEC,$GDADR     ;EXPECTED DATA
3074  011472  023737  001124  043656      CMP     $GDDAT,BUFFER   ;CORRECT CONTENTS ?
3075  011500  001401                      BEQ     10$             ;BR IF CORRECT
3076  011502  104046                      ERROR   46              ;SECTOR PATTERN NOT CORRECT FOR SECTOR ADDRESS
3077  011504  105237  002512          10$:INCB    $SEC            ;INCREMENT SECTOR ADDRESS
3078  011510  123727  002512  000012      CMPB    $SEC,#10.       ;FINISHED SECTOR NINE ?
3079  011516  001404                      BEQ     EXIT10          ;BR IF COMPLETED
3080  011520  113737  002512  001124      MOVB    $SEC,$GDDAT     ;NEXT SECTOR
3081  011526  000736                      BR      8$              ;CONTINUE CHECKING
3082  011530  000004              EXIT10: SCOPE                   ;LOOP ?
3083
3084
3085                                      ;;*****************************************************************
3086                                      ;*TEST 11       TRACK ADDRESSING TEST
3087
3088                                      ;*TRACK ADDRESSING IS TESTED BY WRITING THE TRACK'S ADDRESS AS DATA IN
3089                                      ;*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 ?
```

```
3146
3147
3148                                    ;;*****************************************************************
3149                                    ;*TEST 12       EXTENDED MEMORY ADDRESS TEST
3150
3151                                    ;*THIS TEST TESTS THE OPERATION OF THE EXTENDED MEMORY ADDRESS BITS.  IF THE
3152                                    ;*SYSTEM DOES NOT HAVE MEMORY MANAGEMENT OR HAS MEMORY MANAGEMENT AND ONLY
3153                                    ;*32K, THE TEST WILL NOT BE PERFORMED.
3154                                    ;*
3155                                    ;*      1.  THE PROGRAM WRITES A 2 WORD TEST SECTOR OF ALL ONES.
3156                                    ;*
3157                                    ;*      2.  EXTENDED ADDRESS BIT 'MEX00' IS TESTED BY CLEARING LOCATION
3158                                    ;*      200000 AND READING THE TEST SECTOR INTO LOCATION 177776.  LOCATION
3159                                    ;*      200000 IS CHECKED TO VERIFY THAT THE DATA IS CORRECT (ONES).  THE
3160                                    ;*      PROGRAM VERIFIES THAT 'MEX00' HAS SET AND THAT 'MEX01' IS NOT SET.
3161                                    ;*
3162                                    ;*      3.  IF THE SYSTEM HAS AT LEAST 64K, 'MEX01' IS TESTED.  LOCATION
3163                                    ;*      400000 IS CLEARED AND THE TEST SECTOR IS READ INTO LOCATION 377776
3164                                    ;*      ('MEX00' IS SET FOR THE READ).  LOCATION 400000 IS CHECKED FOR THE
3165                                    ;*      PROPER CONTENTS (ONES).  'MEX00' SHOULD HAVE RESET AND 'MEX01' SHOULD
3166                                    ;*      BE SET.
3167
3168                                    ;;*****************************************************************
3169  012062                            TST12:
3170  012062  033737  001424  001276            BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
3171  012070  001002                            BNE     .+6                  ;BR IF YES
3172  012072  000137  012720                    JMP     TST13                ;GO TO THE NEXT TEST
3173  012076  012737  012202  001110            MOV     #TEST12,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3174  012104  012737  012062  001106            MOV     #TST12,$LPADR        ;SETUP THE SCOPE LOOP ADDRESS
3175  012112  012737  000012  001102            MOV     #12,$TSTNM           ;SETUP TEST NUMBER AND
3176                                                                         ;CLEAR THE ERPOR FLAG ($ERFLG)
3177  012120  013777  001102  167014            MOV     $TSTNM,@DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3178  012126  012737  012712  001200            MOV     #EXIT12,BYPASS       ;SETUP BYPASS ADDRESS
3179  012134  013704  001324                    MOV     RPADR,R4             ;RP11E BUS ADDRESS
3180  012140  105037  002501                    CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
3181  012144  105037  002503                    CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
3182  012150  012737  000144  001162            MOV     #100.,$TIMES         ;;DO 100. ITERATIONS
3183  012156  005737  022232                    TST     $KT11                ;MEMORY MANAGEMENT ON THE SYSTEM ?
3184  012162  100402                            BMI     1$                   ;BR IF ON SYSTEM
3185  012164  000137  012720                    JMP     TST13                ;GO TO THE NEXT TEST
3186  012170  005737  001232          1$:       TST     MAXMEM+2             ;ENOUGH MEMORY FOR THIS TEST ?
3187  012174  001002                            BNE     TEST12               ;BR IF YES
3188  012176  000137  012720                    JMP     TST13                ;GO TO THE NEXT TEST
3189  012202  012706  001100          TEST12:   MOV     #STACK,SP            ;SETUP THE STACK POINTER
3190  012206  012737  012202  001110            MOV     #TEST12,$LPERR       ;INITIAL ERROR LOOP ADDRESS
3191  012214  004737  025654                    JSR     PC,CLRP              ;CLEAR THE RP11
3192  012220  005037  043656                    CLR     BUFFER               ;TEST PATTERN
3193  012224  012737  177777  043660            MOV     #-1,BUFFER+2         ;SECOND WORD OF PATTERN
3194  012232  012737  043656  002506            MOV     #BUFFER,$BUF         ;SETUP THE BUFFER ADDRESS
3195  012240  012737  000002  002504            MOV     #2,$WC               ;WORD COUNT OF 2
3196  012246  005037  002512                    CLR     $SEC                 ;SECTOR/TRACK ZERO
3197  012252  005037  002510                    CLR     $CYL                 ;CYLINDER ZERO
3198  012256  012737  000003  002502            MOV     #WRTSEK,DPB+2        ;WRITE COMMAND
3199  012264  004737  022770                    JSR     PC,RP11              ;INITIATE THE COMMAND
3200  012270  004737  023664                    JSR     PC,CONRDY            ;WAIT FOR CONTROLLER READY
3201  012274  004737  023240                    JSR     PC,SAVRP             ;STORE THE REGISTERS
```

```
3202  012300  032737  100000  001340        BIT     #ERR,$RPCS      ;DID OPERATION TERMINATE WITH AN ERROR ?
3203  012306  001403                        BEQ     1$              ;BR IF NOT
3204  012310  104004                        ERROR   4               ;ERROR AFTER 2 WORD WRITE
3205  012312  000137  012712                JMP     EXIT12          ;BYPASS REST OF TEST
3206  012316  012737  012316  001110  1$:   MOV     #1$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
3207  012324  004737  025654                JSR     PC,CLRP         ;CLEAR THE RP11
3208  012330  012737  177600  172356        MOV     #177600,KIPAR7  ;OPEN I/O REGISTERS
3209  012336  005037  172340                CLR     KIPAR0          ;ENABLE FIRST 4K
3210  012342  012737  000200  172342        MOV     #200,KIPAR1     ;ENABLE SECOND 4K
3211  012350  012737  002000  172344        MOV     #2000,KIPAR2    ;ENABLE FIRST TEST LOCATION
3212  012356  012737  077406  172300        MOV     #77406,KIPDR0   ;LOAD DESCRIPTOR REGISTER 0
3213  012364  012737  077406  172302        MOV     #77406,KIPDR1   ;LOAD DESCRIPTOR REGISTER 1
3214  012372  012737  077406  172304        MOV     #77406,KIPDR2   ;LOAD DESCRIPTOR REGISTER 2
3215  012400  012737  077406  172316        MOV     #77406,KIPDR7   ;LOAD DESCRIPTOR REGISTER 7
3216  012406  012737  000001  177572        MOV     #1,@#SR0        ;TURN ON MEMORY MANAGEMENT
3217  012414  005037  040000                CLR     @#40000         ;CLEAR LOCATION 200000
3218  012420  012737  177776  002506        MOV     #177776,$BUF    ;SETUP BUS ADDR
3219  012426  112737  000005  002502        MOVB    #RDSEK,DPB+2    ;CHANGE COMMAND TO READ
3220  012434  004737  022770                JSR     PC,RP11         ;START THE COMMAND
3221  012440  004737  023664                JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
3222  012444  004737  023240                JSR     PC,SAVRP        ;STORE THE REGISTERS
3223  012450  032737  100000  001340        BIT     #ERR,$RPCS      ;ANY ERRORS?
3224  012456  001401                        BEQ     2$              ;BRANCH IF NOT
3225  012460  104050                        ERROR   50              ;ERROR AFTER READING 2 WORDS
3226  012462  032737  000020  001340  2$:   BIT     #MEX0,$RPCS     ;DID 'MEX0' SET ?
3227  012470  001001                        BNE     3$              ;BRANCH IF IT DID
3228  012472  104051                        ERROR   51              ;'MEX0' DID NOT SET AFTER 2 WORD READ
3229                                                                ;STARTING AT LOCATION 177776
3230  012474  032737  000040  001340  3$:   BIT     #MEX1,$RPCS     ;'MEX1' SHOULD NOT BE SET
3231  012502  001401                        BEQ     4$              ;BRANCH IF NOT SET
3232  012504  104052                        ERROR   52              ;'MEX1' IS SET - SHOULD NOT BE
3233  012506  022737  177777  040000  4$:   CMP     #177777,@#40000 ;WAS DATA READ INTO LOCATION
3234  012514  001407                        BEQ     TST12B          ;200000 CORRECTLY? - BRANCH IF YES
3235  012516  012737  177777  001124        MOV     #177777,$GDDAT  ;EXPECTED CONTENTS
3236  012524  013737  040000  001126        MOV     @#40000,$BDDAT  ;CONTENTS OF LOCATION 200000
3237  012532  104053                        ERROR   53              ;DATA COMPARE ERROR AT LOC 200000
3238  012534  023727  001232  000002  TST12B: CMP    MAXMEM+2,#2     ;ENOUGH MEMORY FOR THE NEXT PART OF THE TEST ?
3239  012542  103463                        BLO     EXIT12          ;BR IF NOT
3240  012544  012737  004000  172344        MOV     #4000,KIPAR2    ;CHANGE TEST LOCATION
3241  012552  012737  012552  001110  1$:   MOV     #1$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
3242  012560  004737  025654                JSR     PC,CLRP         ;CLEAR THE RP11
3243  012564  005037  040000                CLR     @#40000         ;CLEAR LOCATION 400000
3244  012570  012737  177776  002506        MOV     #177776,$BUF    ;BUFFER ADDRESS
3245  012576  112737  000020  002501        MOVB    #MEX0,DPB+1     ;SETUP EXTENDED MEMORY ADDRESS BIT
3246  012604  112737  000005  002502        MOVB    #RDSEK,DPB+2    ;READ COMMAND
3247  012612  004737  022770                JSR     PC,RP11         ;START THE COMMAND
3248  012616  004737  023664                JSR     PC,CONRDY       ;WAIT FOR CONTROLLER READY
3249  012622  004737  023240                JSR     PC,SAVRP        ;STORE THE REGISTERS
3250  012626  032737  100000  001340        BIT     #ERR,$RPCS      ;ANY ERRORS?
3251  012634  001401                        BEQ     2$              ;BRANCH IF NOT
3252  012636  104054                        ERROR   54              ;ERROR WHILE READING TWO WORDS
3253  012640  032737  000020  001340  2$:   BIT     #MEX0,$RPCS     ;IS 'MEX0' SET ?
3254  012646  001401                        BEQ     3$              ;BRANCH IF NOT
3255  012650  104055                        ERROR   55              ;'MEX0' DID NOT CLEAR AFTER 2 WORD
3256                                                                ;READ STARTING AT LOCATION 377776
3257  012652  032737  000040  001340  3$:   BIT     #MEX1,$RPCS     ;DID 'MEX1' SET ?
```

```
3258  012660  001001                          BNE     4$              ;BR IF SET
3259  012662  104056                          ERROR   56              ;'MEX1' DID NOT SET AFTER 2 WORD TRANSFER
3260                                                                  ;STARTING AT LOCATION 377776
3261  012664  022737  177777  040000  4$:     CMP     #177777,@#40000 ;WAS DATA READ INTO LOCATION 400000
3262  012672  001407                          BEQ     EXIT12          ;CORRECTLY ? - BRANCH IF YES
3263  012674  012737  177777  001124          MOV     #177777,$GDDAT  ;EXPECTED DATA
3264  012702  013737  040000  001126          MOV     @#40000,$BDDAT  ;DATA FROM LOCATION 400000
3265  012710  104057                          ERROR   57              ;DATA COMPARE ERROR AT LOC 400000
3266  012712  005037  177572          EXIT12: CLR     @#SR0           ;TURN OFF MEMORY MANAGEMENT
3267  012716  000004                          SCOPE                   ;LOOP ?
3268
3269
3270                                  ;;**************************************************************
3271                                  ;*TEST 13       DATA BUFFER REGISTER BIT TEST
3272
3273                                  ;*THE DATA BUFFER REGISTER TEST VERIFIES THAT ALL 36 BITS OF THE DATA
3274                                  ;*BUFFER REGISTER CAN BE SET AND CLEARED.
3275                                  ;*
3276                                  ;*      1.  THE TEST FIRST WRITES THE FOLLOWING TEST PATTERN IN CYLINDER 0,
3277                                  ;*      TRACK 0, SECTOR 0 USING 10/15 MODE ('MODE' BIT SET).
3278                                  ;*
3279                                  ;*                      052525
3280                                  ;*                      052525
3281                                  ;*                      000005
3282                                  ;*                      125252
3283                                  ;*                      125252
3284                                  ;*                      000012
3285                                  ;*                      052525
3286                                  ;*                      052525
3287                                  ;*                      000005
3288                                  ;*                      125252
3289                                  ;*                        .
3290                                  ;*                        .
3291                                  ;*                        .
3292                                  ;*                       ETC
3293                                  ;*
3294                                  ;*      2.  THE TEST SECTOR IS THEN READ AND THE DATA IS COMPARED.  AT THE
3295                                  ;*      COMPLETION OF THE TEST, THE PROGRAM RESTORES THE TEST SECTOR TO
3296                                  ;*      PDP-11 MODE.
3297
3298                                  ;;**************************************************************
3299  012720                          TST13:
3300  012720  033737  001426  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
3301  012726  001002                          BNE     .+6             ;BR IF YES
3302  012730  000137  013574                  JMP     TST14           ;GO TO THE NEXT TEST
3303  012734  012737  013014  001110          MOV     #TEST13,$LPERR  ;SETUP THE ERROR LOOP ADDRESS
3304  012742  012737  012720  001106          MOV     #TST13,$LPADR   ;SETUP THE SCOPE LOOP ADDRESS
3305  012750  012737  000013  001102          MOV     #13,$TSTNM      ;SETUP TEST NUMBER AND
3306                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
3307  012756  013777  001102  166156          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3308  012764  012737  013572  001200          MOV     #EXIT13,BYPASS  ;SETUP BYPASS ADDRESS
3309  012772  013704  001324                  MOV     RPADR,R4        ;RP11E BUS ADDRESS
3310  012776  105037  002501                  CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
3311  013002  105037  002503                  CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
3312  013006  012737  000144  001162          MOV     #100.,$TIMES    .;DO 100. ITERATIONS
3313  013014  012706  001100          TEST13: MOV     #STACK,SP       ;SETUP THE STACK POINTER
```

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

```
  3326  013070  012737  013070  001110  2$:     MOV     #2$,$LPERR      ;LOOP ON ERROR ADDRESS
  3327  013076  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
```

```
 3328  013102  005037  002512                    CLR     $SEC            ;SECTOR/TRACK ADDRESS OF ZERO
 3329  013106  005037  002510                    CLR     $CYL            ;CYLINDER ZERO
 3330  013112  012737  000600  002504            MOV     #384.,$WC       ;1 SECTOR IN 10/15 MODE
 3331  013120  012737  043656  002506            MOV     #BUFFER,$BUF    ;BUFFER ADDRESS
 3332  013126  012737  010000  002502            MOV     #MODE,DPB+2     ;SET THE 'MODE' BIT
 3333  013134  112737  000003  002502            MOVB    #WRTSEK,DPB+2   ;WRITE COMMAND
 3334  013142  004737  022770                    JSR     PC,RP11         ;DO THE COMMAND
 3335  013146  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE COMMAND TO COMPLETE
 3336  013152  004737  023240                    JSR     PC,SAVRP        ;SAVE THE REGISTERS
 3337  013156  032737  100000  001340            BIT     #ERR,$RPCS      ;OPERATION GO OK ?
 3338  013164  001402                            BEQ     3$              ;BR IF IT DID
 3339  013166  104004                            ERROR   4               ;SETUP OPERATION TERMINATED WITH AN ERROR
 3340  013170  000541                            BR      TST13A          ;CORRECT THE SECTOR AND BYPASS REST OF TEST
 3341  013172  012737  013172  001110  3$:       MOV     #3$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
 3342  013200  004737  025654                    JSR     PC,CLRP         ;CLEAR THE RP11
 3343  013204  012701  043656                    MOV     #BUFFER,R1      ;CLEAR THE INPUT BUFFER
 3344  013210  063701  002504                    ADD     $WC,R1          ;BEGINNING ADDRESS
 3345  013214  013702  002504                    MOV     $WC,R2          ;SIZE
 3346  013220  005001                    4$:     CLR     R1              ;CLEAR THE BUFFER
 3347  013222  005302                            DEC     R2              ;FINISHED ?
 3348  013224  001375                            BNE     4$              ;BR IF NOT
 3349  013226  013737  002504  002506            MOV     $WC,$BUF        ;CHANGE BUFFER ADDRESS
 3350  013234  063737  002504  002506            ADD     $WC,$BUF        ;DOUBLE THE WORD COUNT
 3351  013242  062737  043656  002506            ADD     #BUFFER,$BUF    ;ADD BUFFER BASE ADDRESS
 3352  013250  112737  000005  002502            MOVB    #RDSEK,DPB+2    ;CHANGE COMMAND
 3353  013256  004737  022770                    JSR     PC,RP11         ;READ THE SECTOR
 3354  013262  004737  023664                    JSR     PC,CONRDY       ;WAIT FOR THE OPERATION TO COMPLETE
 3355  013266  004737  023240                    JSR     PC,SAVRP        ;STORE THE REGISTERS
 3356  013272  032737  100000  001340            BIT     #ERR,$RPCS      ;DID AN ERROR OCCUR ?
 3357  013300  001402                            BEQ     5$              ;BR IF NOT
 3358  013302  104017                            ERROR   17              ;ERROR OCCURED READING THE TEST SECTOR
 3359  013304  000473                            BR      TST13A          ;BYPASS THE COMPARSION
 3360  013306  012701  043656          5$:       MOV     #BUFFER,R1      ;GOOD DATA
 3361  013312  013702  002504                    MOV     $WC,R2          ;SIZE
 3362  013316  013703  002504                    MOV     $WC,R3          ;FORM ADDRESS OF TEST DATA
 3363  013322  063703  002504                    ADD     $WC,R3          ;DOUBLE THE WORD COUNT
 3364  013326  062703  043656                    ADD     #BUFFER,R3      ;ADD THE BUFFER START ADDRESS
 3365  013332  005037  013472                    CLR     10$             ;CLEAR THE COMPARSION ERROR COUNTER
 3366  013336  032777  000010  165574  6$:       BIT     #SW03,@SWR      ;COMPARE THE DATA ?
 3367  013344  001053                            BNE     TST13A          ;BR IF NOT
 3368  013346  022123                            CMP     (R1)+,(R3)+     ;COMPARE THE DATA
 3369  013350  001445                            BEQ     9$              ;BR IF OK
 3370  013352  016137  177776  001124            MOV     -2(R1),$GDDAT   ;GOOD DATA
 3371  013360  016337  177776  001126            MOV     -2(R3),$BDDAT   ;BAD DATA
 3372  013366  010337  001122                    MOV     R3,$BDADR       ;FORM BAD ADDRESS
 3373  013372  162737  000002  001122            SUB     #2,$BDADR       ;DECREMENT IT
 3374  013400  005737  013472                    TST     10$             ;ANY COMPARE ERROR ALREADY ?
 3375  013404  001010                            BNE     7$              ;BR IF SOME
 3376  013406  012737  013406  001116            MOV     #.,$ERRPC       ;PC FOR ERROR MESSAGE
 3377  013414  112737  000102  001114            MOVB    #102,$ITEMB     ;ERROR TABLE INDEX
 3378  013422  004737  017254                    JSR     PC,TYPERR       ;REPORT COMPARSION ERROR
 3379  013426  112737  000104  001114  7$:       MOVB    #104,$ITEMB     ;INDEX OF DATA DISPLAY MESSAGE
 3380  013434  004737  017254                    JSR     PC,TYPERR       ;TYPE THE MESSAGE
 3381  013440  005237  013472          8$:       INC     10$             ;INCREMENT COMPARSION ERROR COUNT
 3382  013444  023737  001320  013472            CMP     CMP.CT,10$      ;COMPARE AGAINST THE LIMIT
 3383  013452  103004                            BHIS    9$              ;BR IF NOT GREATER
```

```
3384  013454  032777  000020  165456          BIT     #SWU4,@SWR      ;CONTINUE COMPARING ANYWAY ?
3385  013462  001404                          BEQ     TST13A          ;BR IF NOT
3386  013464  005302                  9$:     DEC     R2              ;FINISHED COMPARING ?
3387  013466  001323                          BNE     6$              ;BR IF NOT
3388  013470  000401                          BR      TST13A          ;CHANGE MODE OF THE SECTOR
3389  013472  000000                  10$:    .WORD   0               ;COMPARSION ERROR COUNTER
3390  013474  012737  013474  001110  TST13A: MOV     #TST13A,$LPERR  ;CHANGE LOOP ON ERROR ADDRESS
3391  013502  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
3392  013506  105037  002503                  CLRB    DPB+3           ;CLEAR THE 'MODE' & 'HDR' BITS
3393  013512  012737  000002  002504          MOV     #2,$WC          ;CHANGE WORD COUNT
3394  013520  112737  000003  002502          MOVB    #WRTSEK,DPB+2   ;CHANGE COMMAND
3395  013526  005037  043656                  CLR     BUFFER          ;SET DATA TO ZERO
3396  013532  005037  043660                  CLR     BUFFER+2        ;OTHER DATA WORD
3397  013536  012737  043656  002506          MOV     #BUFFER,$BUF    ;BUFFER ADDRESS
3398  013544  004737  022770                  JSR     PC,RP11         ;RE-WRITE THE SECTOR
3399  013550  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR THE OPERATION TO COMPLETE
3400  013554  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
3401  013560  032737  100000  001340          BIT     #ERR,$RPCS      ;DID THE WRITE FINISH WITHOUT ERROR ?
3402  013566  001401                          BEQ     EXIT13          ;BR IF IT DID
3403  013570  104060                          ERROR   60              ;ERROR RE-WRITING THE SECTOR
3404  013572  000004                  EXIT13: SCOPE                   ;LOOP ?
3405
3406                                  ;;**************************************************************
3407                                  ;*TEST 14        DATA STORAGE/RETRIEVAL TEST
3408
3409                                  ;*THE DATA TEST PERFORMS DATA STORAGE AND RETRIEVAL TESTING USING THE CONTROLLER
3410                                  ;*AND THE CURRENTLY SELECTED DRIVE.  UNLESS ALTERED BY THE OPERATOR, THE
3411                                  ;*TEST WILL WRITE AND WRITE CHECK THE ENTIRE DISK PACK USING ALL 16 DATA
3412                                  ;*PATTERNS IN A ROTATING MANNER.  THE BUFFER SIZE USED FOR THE WRITE AND
3413                                  ;*WRITE CHECK ORDERS IS DETERMINED BY THE AVAILABLE MEMORY ON THE SYSTEM
3414                                  ;*(MINUS THE SPACE REQUIRED BY THE PROGRAM LOADER).  THE PROGRAM WILL USE
3415                                  ;*THE AVAILABLE MEMORY SIZE OR 8K, WHICHEVER IS LESS, AS THE BUFFER
3416                                  ;*SIZE.
3417                                  ;*
3418                                  ;*AFTER THE PACK HAS BEEN WRITTEN, THE TEST WILL READ AND COMPARE THE
3419                                  ;*DATA.  THE BUFFER SIZE USED FOR THE READ IS THE SAME SIZE AS THAT USED
3420                                  ;*FOR THE WRITE/WRITE CHECK PART OF THE TEST.  FOR THE READ, THE TEST WILL
3421                                  ;*ROTATE THE READ BUFFER THROUGH MEMORY SO THAT ALL OF THE AVAILABLE MEMORY
3422                                  ;*IS USED BY THE READ AND COMPARE OPERATIONS.  THE TEST USES MEMORY MANAGEMENT
3423                                  ;*ON SYSTEMS WITH MORE THAN 28K.  (MEMORY MANAGEMENT WILL NOT BE USED IF
3424                                  ;*ON MEMORY MANAGEMENT SYSTEMS WITH 28K OR LESS.)
3425                                  ;*
3426                                  ;*THE DATA PATTERNS USED BY THE DATA TEST ARE GIVEN BELOW:
3427                                  ;*
3428                                  ;*      PAT 0   PAT 1   PAT 2   PAT 3   PAT 4   PAT 5   PAT 6   PAT 7
3429                                  ;*      ------  ------  ------  ------  ------  ------  ------  ------
3430                                  ;*      000000  000001  177776  000000  000000  052525  007417  026455
3431                                  ;*      000000  000003  177774  000000  010421  052525  007417  026455
3432                                  ;*      000000  000007  177770  000000  021042  052525  007417  026455
3433                                  ;*      000000  000017  177760  177777  031463  125252  170360  151322
3434                                  ;*      000000  000037  177740  177777  042104  125252  170360  151322
3435                                  ;*      000000  000077  177700  177777  052525  125252  170360  151322
3436                                  ;*      000000  000177  177600  000000  063146  052525  007417  026455
3437                                  ;*      000000  000377  177400  000000  073567  052525  007417  026455
3438                                  ;*      000000  000777  177000  177777  104210  125252  170360  151322
3439                                  ;*      000000  001777  176000  177777  114631  125252  170360  151322
```

```
3440                                    ;*      000000  003777  174000  000000  125252  052525  007417  026455
3441                                    ;*      000000  007777  170000  177777  135673  125252  170360  151322
3442                                    ;*      000000  017777  160000  000000  146314  052525  007417  026455
3443                                    ;*      000000  037777  140000  177777  156735  125252  170360  151322
3444                                    ;*      000000  077777  100000  177777  167356  125252  170360  151322
3445                                    ;*      000000  177777  000000  000000  177777  052525  007417  026455
3446                                    ;*
3447                                    ;*      PAT 8   PAT 9   PAT 10  PAT 11  PAT 12  PAT 13  PAT 14  PAT 15
3448                                    ;*      ------  ------  ------  ------  ------  ------  ------  ------
3449                                    ;*      077577  000001  177776  172666  077777  177777  000000  177777
3450                                    ;*      077577  000002  177775  155555  137777  177777  000000  177777
3451                                    ;*      077577  000004  177773  172666  157777  177777  177777  000000
3452                                    ;*      077577  000010  177767  155555  167777  177777  177777  000000
3453                                    ;*      077577  000020  177757  172666  173777  177777  177777  000000
3454                                    ;*      077577  000040  177737  155555  175777  177777  177777  000000
3455                                    ;*      077577  000100  177677  172666  176777  177777  177777  000000
3456                                    ;*      077577  000200  177577  155555  177377  177777  177777  000000
3457                                    ;*      077577  000400  177377  172666  177577  177777  177777  000000
3458                                    ;*      077577  001000  176777  155555  177677  177777  177777  000000
3459                                    ;*      077577  002000  175777  172666  177737  177777  177777  000000
3460                                    ;*      077577  004000  173777  155555  177757  177777  177777  000000
3461                                    ;*      077577  010000  167777  172666  177767  177777  177777  000000
3462                                    ;*      077577  020000  157777  155555  177773  177777  177777  000000
3463                                    ;*      077577  040000  137777  172666  177775  177777  177777  000000
3464                                    ;*      077577  100000  077777  155555  177776  177777  177777  000000
3465                                    ;*
3466
3467                                    ;;**********************************************************************
3468  013574                            TST14:
3469  013574  033737  001430  001276            BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
3470  013602  001002                            BNE     .+6                 ;BR IF YES
3471  013604  000137  014630                    JMP     TST15               ;GO TO THE NEXT TEST
3472  013610  012737  014066  001110            MOV     #TEST14,$LPERR      ;SETUP THE ERROR LOOP ADDRESS
3473  013616  012737  013574  001106            MOV     #TST14,$LPADR       ;SETUP THE SCOPE LOOP ADDRESS
3474  013624  012737  000014  001102            MOV     #14,$TSTNM          ;SETUP TEST NUMBER AND
3475                                            ;CLEAR THE ERROR FLAG ($ERFLG)
3476  013632  013777  001102  165302            MOV     $TSTNM,@DISPLAY     ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3477  013640  012737  014616  001200            MOV     #EXIT14,BYPASS      ;SETUP BYPASS ADDRESS
3478  013646  013704  001324                    MOV     RPADR,R4            ;RP11E BUS ADDRESS
3479  013652  105037  002501                    CLRB    DPB+1               ;CLEAR ANY EXTENDED MEMORY BITS
3480  013656  105037  002503                    CLRB    DPB+3               ;CLEAR 'MODE' & 'HDR' BITS
3481  013662  012706  001100                    MOV     #STACK,SP           ;SETUP THE STACK POINTER
3482  013666  032777  000100  165244            BIT     #SW06,@SWR          ;USE MEMORY MANAGEMENT ?
3483  013674  001021                            BNE     2$                  ;BR IF NOT
3484  013676  005737  022232          1$:       TST     $KT11               ;MEMORY MANAGEMENT AVAILABLE ?
3485  013702  100016                            BPL     2$                  ;BR IF NOT
3486  013704  012737  177777  001222            MOV     #-1,MMACTV          ;SET MEMORY MANAGEMENT ACTIVE INDICATOR
3487  013712  004737  023346                    JSR     PC,MOVLDR           ;MOVE THE LOADER
3488  013716  012637  001240                    MOV     (SP)+,LOMEM         ;BEGINNING ADDRESS OF BUFFER SPACE
3489  013722  013737  001230  001234            MOV     MAXMEM,HIMEM        ;HIGHEST MEMORY ADDRESS USED BY THE TEST
3490  013730  013737  001232  001236            MOV     MAXMEM+2,HIMEM+2    ;UPPER ADDRESS BITS
3491  013736  000412                            BR      3$                  ;CONTINUE
3492  013740  005037  001222          2$:       CLR     MMACTV              ;CLEAR MEMORY MANAGEMENT ACTIVE INDICATOR
3493  013744  005037  001236                    CLR     HIMEM+2             ;CLEAR UPPER 'HIMEM' ADDRESS BITS
3494  013750  013737  001302  001234            MOV     MEMSIZ,HIMEM        ;HIGHEST NON MEMORY MANAGENEMT LOCATION
3495  013756  012737  043656  001240            MOV     #BUFFER,LOMEM       ;DEFAULT BUFFER STARTING ADDRESS
```

```
 3496  013764  005037  001242          3$:     CLR     LOMEM+2          ;CLEAR UPPER STORAGE WORD
 3497  013770  013737  001240  001224          MOV     LOMEM,ACTMEM     ;STARTING ADDRESS OF FIRST BUFFER
 3498  013776  013737  001242  001226          MOV     LOMEM+2,ACTMEM+2  ;UPPER ADDRESS BITS
 3499  014004  013737  001256  001254          MOV     WC14,BLKS14      ;CONVERT THE WORD COUNT TO A SECTOR COUNT
 3500  014012  000337  001254                  SWAB    BLKS14           ;RIGHT JUSTIFY THE SECTOR COUNT
 3501  014016  105037  001255                  CLRB    BLKS14+1         ;CLEAR THE RESIDUE
 3502  014022  005237  001254                  INC     BLKS14           ;CHANGE TO ABSOLUTE COUNT
 3503  014026  013746  001254                  MOV     BLKS14,-(SP)     ;CONVERT BLOCK COUNT TO SECTOR/TRACK INCREMENTS
 3504  014032  005037  001252                  CLR     INCTRK           ;CLEAR TRACK STORAGE
 3505  014036  005037  001250                  CLR     INCSEC           ;CLEAR SECTOR STORAGE
 3506  014042  162716  000012          4$:     SUB     #10.,(SP)        ;SUBTRACT A FULL SECTOR
 3507  014046  100403                          BMI     5$               ;BR IF MINUS
 3508  014050  005237  001252                  INC     INCTRK           ;COUNT THE TRACK
 3509  014054  000772                          BR      4$               ;CONTINUE
 3510  014056  062716  000012          5$:     ADD     #10.,(SP)        ;CORRECT THE SECTOR RESIDUE
 3511  014062  012637  001250                  MOV     (SP)+,INCSEC     ;SECTOR INCREMENT VALUE
 3512  014066  012706  001100          TEST14: MOV     #STACK,SP        ;LOAD THE STACK POINTER
 3513  014072  004737  024002                  JSR     PC,SECNBR        ;GET THE NUMBER OF SECTORS ON THE PACK
 3514                                                                   ;BEING USED
 3515  014076  113737  001272  002512  1$:     MOVB    SSEC,$SEC        ;STARTING SECTOR ADDRESS
 3516  014104  113737  001270  002513          MOVB    STRK,$TRK        ;STARTING TRACK ADDRESS
 3517  014112  013737  001266  002510          MOV     SCYL,$CYL        ;STARTING CYLINDER ADDRESS
 3518  014120  013737  001240  002506          MOV     LOMEM,$BUF       ;BUFFER ADDRESS
 3519  014126  013737  001256  002504          MOV     WC14,$WC         ;STARTING WORD COUNT
 3520  014134  012737  000017  001204          MOV     #15.,PATNUM      ;PRESET THE PATTERN INDEX
 3521  014142  005737  001260                  TST     OPRSEL           ;USING AN OPERATOR SPECIFIED ADDRESS ?
 3522  014146  001005                          BNE     2$               ;BR IF YES
 3523  014150  163737  001254  001244          SUB     BLKS14,PAKSIZ    ;DECREMENT THE SECTOR COUNT FOR THE
 3524  014156  005637  001246                  SBC     PAKSIZ+2         ;FIRST OPERATION
 3525  014162  004737  024050          2$:     JSR     PC,PATSEL        ;SELECT THE PATTERN
 3526  014166  012737  014166  001110  3$:     MOV     #3$,$LPERR       ;LOOP ON ERROR ADDRESS
 3527  014174  004737  025654                  JSR     PC,CLRP          ;CLEAR THE RP11
 3528  014200  017746  164734                  MOV     @SWR,-(SP)       ;INHIBIT WRITE AND
 3529  014204  005116                          COM     (SP)             ;WRITE CHECK ?
 3530  014206  032726  000003                  BIT     #SW00!SW01,(SP)+
 3531  014212  001453                          BEQ     TST14A           ;YES--BRANCH
 3532  014214  004737  024432                  JSR     PC,SETBUF        ;MOVE DATA PATTERN INTO THE BUFFER
 3533  014220  032777  000001  164712          BIT     #SW00,@SWR       ;INHIBIT WRITE ?
 3534  014226  001016                          BNE     4$               ;YES--BRANCH
 3535  014230  112737  000003  002502          MOVB    #WRTSEK,DPB+2     ;COMMAND=WRITE
 3536  014236  004737  022770                  JSR     PC,RP11          ;DO THE COMMAND
 3537  014242  004737  023664                  JSR     PC,CONRDY        ;WAIT FOR THE COMMAND TO COMPLETE
 3538  014246  004737  023240                  JSR     PC,SAVRP         ;STORE THE REGISTERS
 3539  014252  032737  100000  001340          BIT     #ERR,$RPCS       ;ERROR OCCURED ?
 3540  014260  001401                          BEQ     4$               ;BR IF NOT
 3541  014262  104061                          ERROR   61               ;REPORT THE ERROR
 3542  014264  032777  000002  164646  4$:     BIT     #SW01,@SWR       ;INHIBIT WRITE CHECK ?
 3543  014272  001020                          BNE     5$               ;YES--BRANCH
 3544  014274  004737  025654                  JSR     PC,CLRP          ;CLEAR THE RP11
 3545  014300  112737  000007  002502          MOVB    #WRTCHK,DPB+2    ;COMMAND=WRITE CHECK
 3546  014306  004737  022770                  JSR     PC,RP11          ;DO THE WRITE CHECK
 3547  014312  004737  023664                  JSR     PC,CONRDY        ;WAIT FOR THE COMMAND TO FINISH
 3548  014316  004737  023240                  JSR     PC,SAVRP         ;STORE THE REGISTERS
 3549  014322  032737  100000  001340          BIT     #ERR,$RPCS       ;ERROR ?
 3550  014330  001401                          BEQ     5$               ;BR IF NOT
 3551  014332  104005                          ERROR   5                ;REPORT THE WRITE CHECK ERROR
```

```
3552  014334  004737  024114          5$:     JSR     PC,INCADR       ;INCREMENT THE ADDRESS
3553  014340  000710                  TST14A: BR      2$              ;CONTINUE
3554  014342  004737  024002                  JSR     PC,SECNBR       ;GET NUMBER OF SECTORS ON THE DRIVE
3555                                                                    ;BEING TESTED
3556  014346  113737  001272  002512          MOVB    SSEC,$SEC       ;STARTING SECTOR
3557  014354  113737  001270  002513          MOVB    STRK,$TRK       ;STARTING TRACK
3558  014362  013737  001266  002510          MOV     SCYL,$CYL       ;STARTING CYLINDER
3559  014370  013737  001256  002504          MOV     WC14,$WC        ;INITIAL WORD COUNT
3560  014376  012737  000017  001204          MOV     #15.,PATNUM     ;PRESET PATTERN NUMBER
3561  014404  005737  001260                  TST     OPRSEL          ;USING AN OPERATOR SPECIFIED PATTERN
3562  014410  001005                          BNE     1$              ;BR IF YES
3563  014412  163737  001254  001244          SUB     BLKS14,PAKSIZ   ;DECREMENT THE SECTOR COUNT FOR
3564  014420  005637  001246                  SBC     PAKSIZ+2        ;THE FIRST OPERATION
3565  014424  005037  001314          1$:     CLR     RETRY           ;CLEAR THE RETRY COUNTER
3566  014430  004737  024050                  JSR     PC,PATSEL       ;SELECT A PATTERN
3567  014434  004737  024502                  JSR     PC,GETBUF       ;GENERATE A BUFFER ADDRESS
3568  014440  012737  014440  001110  2$:     MOV     #2$,$LPERR      ;LOOP ON ERROR ADDRESS
3569  014446  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
3570  014452  032777  000004  164460          BIT     #SW02,@SWR      ;INHIBIT READ DATA AND SOFTWARE COMPARE ?
3571  014460  001056                          BNE     EXIT14          ;YES--BRANCH
3572  014462  112737  000005  002502          MOVB    #RDSEK,DPB+2    ;LOAD READ COMMAND
3573  014470  004737  022770                  JSR     PC,RP11         ;START THE READ
3574  014474  004737  023664                  JSR     PC,CONRDY       ;WAIT FOR THE READ TO COMPLETE
3575  014500  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
3576  014504  032737  100000  001340          BIT     #ERR,$RPCS      ;ERROR ?
3577  014512  001420                          BEQ     5$              ;BR IF NOT
3578  014514  005737  001314                  TST     RETRY           ;RETRY ATTEMPT ?
3579  014520  001404                          BEQ     3$              ;BR IF NOT
3580  014522  032777  000040  164410          BIT     #SW05,@SWR      ;DISPLAY ALL RETRY ATTEMPTS
3581  014530  001401                          BEQ     4$              ;BR IF NOT
3582  014532  104062                  3$:     ERROR   62              ;REPORT THE ERROR
3583  014534  005237  001314          4$:     INC     RETRY           ;INCREMENT THE RETRY COUNT
3584  014540  023737  001314  001316          CMP     RETRY,LRETRY    ;RETRY LIMIT ?
3585  014546  101734                          BLOS    2$              ;BR IF NOT
3586  014550  104110                          ERROR   110             ;REPORT UNSUCESSFUL RETRY
3587  014552  000410                          BR      6$              ;CONTINUE
3588  014554  005737  001314          5$:     TST     RETRY           ;COMING FROM A SUCESSFUL RETRY ?
3589  014560  001405                          BEQ     6$              ;BR IF NOT
3590  014562  112737  000111  001114          MOVB    #111,$ITEMB     ;RETRY COUNT MESSAGE NUMBER
3591  014570  004737  017254                  JSR     PC,TYPERR       ;REPORT THE RETRY COUNT
3592  014574  032777  000010  164336  6$:     BIT     #SW03,@SWR      ;COMPARE THE DATA ?
3593  014602  001002                          BNE     7$              ;NO--BRANCH
3594  014604  004737  024614                  JSR     PC,DATCMP       ;YES--DO IT
3595  014610  004737  024114          7$:     JSR     PC,INCADR       ;INCREMENT THE DISK ADDRESS
3596  014614  000703                          BR      1$              ;CONTINUE
3597  014616  000004                  EXIT14: SCOPE                   ;LOOP ?
3598  014620  004737  023452                  JSR     PC,RESLDR       ;RESTORE THE LOADER IF IT WAS MOVED
3599  014624  000137  016616                  JMP     $EOP            ;GO TO END OF TEST
3600
3601                                  ;;*****************************************************************
3602                                  ;*TEST 15        POWER FAIL TEST
3603
3604                                  ;*TEST THE ABILITY OF THE RP11E TO SENSE POWER FAILURE AND FOR THE DRIVES
3605                                  ;*TO RETRACT THEIR HEADS.  WHEN POWER IS TURNED ON AGAIN
3606                                  ;*THE CYLINDER ADDRESS IS TESTED FOR ZERO.  AFTER TYPING THE MESSAGE
3607                                  ;*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
```

```
3664  015146  004737  022666              JSR     PC,DRVINT       ;CHECK THE DRIVE'S STATUS
3665  015152  105761  001370              TSTB    DRVSTA(R1)      ;IS THE DRIVE ONLINE ?
3666  015156  001761                      BEQ     1$              ;CONTINUE WAITING FOR DRIVE TO COME ONLINE
3667  015160  003002                      BGT     2$              ;BR IF IT IS ONLINE
3668  015162  104112                      ERROR   112             ;DRIVE UNSAFE AFTER POWER FAIL
3669  015164  000446                      BR      EXIT15          ;DRIVE UNSAFE, EXIT
3670  015166  113764  001176  000005  2$: MOVB    CHKDRV,RPCS+1(R4)  ;SELECT THE DRIVE
3671  015174  004737  023240              JSR     PC,SAVRP        ;STORE THE REGISTERS
3672  015200  005737  001354              TST     $SUCA           ;SELECTED CYLINDER SHOULD BE ZERO
3673  015204  001404                      BEQ     3$              ;BR IF IT DID
3674  015206  012737  015216  001164      MOV     #3$,$ESCAPE     ;;ESCAPE TO 3$ ON ERROR
3675  015214  104063                      ERROR   63              ;DRIVE DID NOT RETURN TO CYLINDER 0
3676                                                              ;ON POWER FAILURE
3677  015216                          3$:
3678  015216  012737  015276  001164      MOV     #6$,$ESCAPE     ;;ESCAPE TO 6$ ON ERROR
3679  015224  013700  001240              MOV     LOMEM,R0        ;BUFFER ADDRESS
3680  015230  022720  125252          4$: CMP     #125252,(R0)+   ;DID MEMORY RETAIN PATTERN ON POWER FAIL ?
3681  015234  001004                      BNE     5$              ;BRANCH IF NOT
3682  015236  020037  001302              CMP     R0,MEMSIZ       ;END OF MEMORY ?
3683  015242  001372                      BNE     4$              ;BR IF NOT
3684  015244  000414                      BR      6$              ;EXIT
3685  015246  012737  125252  001124  5$: MOV     #125252,$GDDAT  ;EXPECTED MEMORY CONTENTS
3686  015254  016037  177776  001126      MOV     -2(R0),$BDDAT   ;ACTUAL MEMORY CONTENTS
3687  015262  010037  001122              MOV     R0,$BDADR       ;ADDRESS
3688  015266  162737  000002  001122      SUB     #2,$BDADR       ;CORRECT IT
3689  015274  104064                      ERROR   64              ;CONTENTS OF MEMORY CHANGED AFTER POWER FAIL
3690  015276  104401  027617          6$: TYPE    ,MSG15C         ;'END OF POWER FAIL TEST'
3691  015302  000004                  EXIT15: SCOPE               ;LOOP ?
3692  015304  004737  023452              JSR     PC,RESLDR       ;RESTORE THE LOADER
3693  015310  000137  003676              JMP     START2          ;RETURN TO 'CONVERSATION MODE' ENTRANCE
3694
3695                                  ;POWER FAIL TRAP HANDLER
3696
3697  015314  012737  015324  000024  TST15B: MOV    #TST15C,@#24    ;SET UP POWER UP VECTOR
3698  015322  000774                      BR      TST15B          ;LOOP
3699
3700                                  ;POWER UP TRAP HANDLER
3701
3702  015324  012737  000026  000024  TST15C: MOV    #26,@#24        ;RESTORE TRAP CATCHER
3703  015332  005037  000026              CLR     @#26            ;CATCHER HALT
3704  015336  012706  001100              MOV     #STACK,SP       ;SETUP STACK POINTER
3705  015342  005037  177776              CLR     PS              ;RETURN TO PRIORITY ZERO
3706  015346  012737  011610  001220      MOV     #5000.,STALLT   ;LOAD STALL VALUE
3707  015354  004737  023202              JSR     PC,STALL        ;STALL FOR 5000. MILLISECONDS
3708  015360  004737  020416              JSR     PC,$TKINT       ;INITIALIZE THE TTY KEYBOARD
3709  015364  013704  001324              MOV     RPADR,R4        ;RP11E ADDRESS
3710  015370  000650                      BR      TST15A          ;VERIFY POWER DOWN SEQUENCE
3711
3712
3713                                  ;;*****************************************************************
3714                                  ;*TEST 16       HEAD ALIGNMENT ROUTINE
3715
3716                                  ;*THIS ROUTINE PERFORMS HEAD SELECTION AS DIRECTED FROM THE KEYBOARD.
3717                                  ;*THE ALIGNMENT OF THE SELECTED HEAD MAY BE CHECKED OR ADJUSTED.
3718
3719                                  ;;*****************************************************************
```

```
3720  015372                          TST16:
3721  015372  033737  001434  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
3722  015400  001002                          BNE     .+6                  ;BR IF YES
3723  015402  000137  015624                  JMP     TST17                ;GO TO THE NEXT TEST
3724  015406  012737  015460  001110          MOV     #TEST16,$LPERR       ;SETUP THE ERROR LOOP ADDRESS
3725  015414  012737  015372  001106          MOV     #TST16,$LPADR        ;SETUP THE SCOPE LOOP ADDRESS
3726  015422  012737  000016  001102          MOV     #16,$TSTNM           ;SETUP TEST NUMBER AND
3727                                                                       ;CLEAR THE ERROR FLAG ($ERFLG)
3728  015430  013777  001102  163504          MOV     $TSTNM,@DISPLAY      ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
3729  015436  012737  015620  001200          MOV     #EXIT16,BYPASS       ;SETUP BYPASS ADDRESS
3730  015444  013704  001324                  MOV     RPADR,R4             ;RP11E BUS ADDRESS
3731  015450  105037  002501                  CLRB    DPB+1                ;CLEAR ANY EXTENDED MEMORY BITS
3732  015454  105037  002503                  CLRB    DPB+3                ;CLEAR 'MODE' & 'HDR' BITS
3733  015460  012706  001100          TEST16: MOV     #STACK,SP            ;SETUP THE STACK POINTER
3734  015464  104401  027743                  TYPE    ,MSG16A              ;'HEAD ALIGNMENT ROUTINE'
3735  015470  000410                          BR      3$                   ;GET THE HEAD ADDRESS WITHOUT THE SW07 NONESENSE
3736  015472  032777  000200  163440  1$:     BIT     #SW07,@SWR           ;SWITCH 7 SET ?
3737  015500  001774                          BEQ     1$                   ;BR IF NOT
3738  015502  032777  000200  163430  2$:     BIT     #SW07,@SWR           ;SWITCH 7 RESET ?
3739  015510  001374                          BNE     2$                   ;BR IF NOT
3740  015512  104401  030255          3$:     TYPE    ,MSG16B              ;'ENTER HEAD ADDRESS (IN OCTAL)'
3741  015516  104411                          RDLIN                        ;GET THE ENTRY
3742  015520  012601                          MOV     (SP)+,R1             ;STARTING ADDRESS OF INPUT ASCII STRING
3743  015522  004037  027136                  JSR     R0,CK.NUM            ;CHECK THE NUMBER
3744  015526  015512                          3$                           ;ILLEGAL INPUT
3745  015530  015534                          4$                           ;TERMINATED WITH A 'CR'
3746  015532  015512                          3$                           ;NO ENTRY - 'CR' ONLY
3747  015534  122702  000024          4$:     CMPB    #20.,R2              ;VALID ?
3748  015540  101764                          BLOS    3$                   ;BR IF NOT
3749  015542  110237  002513                  MOVB    R2,$TRK              ;LOAD NEW HEAD ADDRESS
3750  015546  136237  001360  001304          BITB    ATABIT(R2),DRVTYP    ;RP02 OR RP03 ?
3751  015554  001004                          BNE     5$                   ;BR IF RP03
3752  015556  012737  000111  002510          MOV     #73.,$CYL            ;RP02 ALIGNMENT CYLINDER
3753  015564  000403                          BR      6$                   ;CONTINUE
3754  015566  012737  000222  002510  5$:     MOV     #146.,$CYL           ;RP03 ALIGNMENT CYLINDER
3755  015574  112737  000011  002502  6$:     MOVB    #SEEK,DPB+2          ;SEEK COMMAND
3756  015602  004737  025654                  JSR     PC,CLRP              ;CLEAR THE RP11
3757  015606  004737  022770                  JSR     PC,RP11              ;SEEK AND LOAD THE HEAD
3758  015612  004737  023546                  JSR     PC,DRVRDY            ;WAIT FOR THE DRIVE TO BECOME READY
3759  015616  000725                          BR      1$                   ;WAIT FOR SWITCH 7 TO BE TOGGLED
3760  015620  000000                  EXIT16: HALT                         ;'EXIT' FOR COMPATABILITY WITH OTHER TESTS
3761  015622  000776                          BR      .-2                  ;LOCK THE HALT
3762
3763
3764                                  ;;*******************************************************************
3765                                  ;*TEST 17       RP11E/DRIVE CONTROL PANEL SWITCH TEST
3766
3767                                  ;*THIS TEST TEST THE 'ENABLE/DISABLE' AND 'READ ONLY  SWITCHES ON
3768                                  ;*THE DRIVE CONTROL PANEL AND TESTS THE 'WRITE LOCKOUT' AND 'LOA'
3769                                  ;*SWITCHES ON THE RP11E/DRIVE CONTROL PANEL.
3770
3771                                  ;;*******************************************************************
3772  015624                          TST17:
3773  015624  033737  001436  001276          BIT     BITS+<$TN*2>,TSTNMS  ;IS THIS TEST SELECTED
3774  015632  001002                          BNE     .+6                  ;BR IF YES
3775  015634  000137  016616                  JMP     @#$EOP               ;GO TO END OF TEST
```

```
 3776  015640  012737  015716  001110          MOV     #TEST17,$LPERR  ;SETUP THE ERROR LOOP ADDRESS
 3777  015646  012737  015624  001106          MOV     #TST17,$LPADR   ;SETUP THE SCOPE LOOP ADDRESS
 3778  015654  012737  000017  001102          MOV     #17,$TSTNM      ;SETUP TEST NUMBER AND
 3779                                                                  ;CLEAR THE ERROR FLAG ($ERFLG)
 3780  015662  013777  001102  163252          MOV     $TSTNM,@DISPLAY  ;LOAD THE TEST NUMBER INTO THE DISPLAY REGISTER
 3781  015670  012737  016604  001200          MOV     #EXIT17,BYPASS  ;SETUP BYPASS ADDRESS
 3782  015676  013704  001324                  MOV     RPADR,R4        ;RP11E BUS ADDRESS
 3783  015702  105037  002501                  CLRB    DPB+1           ;CLEAR ANY EXTENDED MEMORY BITS
 3784  015706  105037  002503                  CLRB    DPB+3           ;CLEAR 'MODE' & 'HDR' BITS
 3785  015712  104401  030317                  TYPE    ,MSG17A         ;'RP11E CONTROL PANEL SWITCH TEST'
 3786  015716  012706  001100          TEST17: MOV     #STACK,SP       ;SETUP THE STACK POINTER
 3787
 3788                                  ;CHECK THE ENABLE/DISABLE SWITCH ON THE DRIVE
 3789
 3790  015722  104401  030362                  TYPE    ,MSG17B ;'TYPE A CR TO START TEST AFTER EACH SETUP'
 3791  015726  104401  030441          2$:     TYPE    ,MSG17C         ;'DISABLE DRIVE # '
 3792  015732  013746  001176                  MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
 3793                                                                  ;;TYPE DRIVE NUMBER
 3794  015736  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 3795  015740     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
 3796  015741     001                          .BYTE   1               ;;TYPE LEADING ZEROS
 3797  015742  104411                          RDLIN                   ;WAIT FOR THE 'CR'
 3798  015744  012737  015744  001110  3$:     MOV     #3$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
 3799  015752  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
 3800  015756  113764  001176  000005          MOVB    CHKDRV,RPCS+1(R4)  ;SELECT DRIVE
 3801  015764  004737  023240                  JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
 3802  015770  032737  040000  001334          BIT     #SUOL,$RPDS     ;IS THE UNIT ON LINE?
 3803  015776  001401                          BEQ     4$              ;BRANCH IF NOT
 3804  016000  104065                          ERROR   65              ;SELECTED UNIT ON LINE WITH THE
 3805                                                                  ;DRIVE DISABLED
 3806  016002  104401  030464          4$:     TYPE    ,MSG17D         ;'ENABLE DRIVE # '
 3807  016006  013746  001176                  MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
 3808                                                                  ;;TYPE DRIVE NUMBER
 3809  016012  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 3810  016014     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
 3811  016015     001                          .BYTE   1               ;;TYPE LEADING ZEROS
 3812  016016  104411                          RDLIN                   ;WAIT FOR THE 'CR'
 3813  016020  012737  016020  001110  5$:     MOV     #5$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
 3814  016026  004737  025654                  JSR     PC,CLRP         ;CLEAR THE RP11
 3815  016032  113764  001176  000005          MOVB    CHKDRV,RPCS+1(R4)  ;RESELECT THE UNIT
 3816  016040  004737  023240                  JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
 3817  016044  032737  040000  001334          BIT     #SUOL,$RPDS     ;IS THE UNIT ON LINE?
 3818  016052  001001                          BNE     RDSW            ;BR IF ONLINE
 3819  016054  104066                          ERROR   66              ;SELECTED UNIT NOT ON LINE WITH
 3820                                                                  ;THE DRIVE ENABLED
 3821
 3822                                  ;CHECK THE DRIVE'S WRITE LOCKOUT SWITCH
 3823
 3824  016056  104401  030506          RDSW:   TYPE    ,MSG17E         ;'SET READ ONLY ON DRIVE '
 3825  016062  013746  001176                  MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
 3826                                                                  ;;TYPE DRIVE NUMBER
 3827  016066  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 3828  016070     001                          .BYTE   1               ;;TYPE 1 DIGIT(S)
 3829  016071     001                          .BYTE   1               ;;TYPE LEADING ZEROS
 3830  016072  104411                          RDLIN                   ;WAIT FOR 'CR'
 3831  016074  012737  016074  001110  2$:     MOV     #2$,$LPERR      ;LOAD LOOP ON ERROR ADDRESS
```

```
3832  016102  004737  025554          JSR     PC,CLRP           ;CLEAR THE RP11
3833  016106  113764  001176  000005  MOVB    CHKDRV,RPCS+1(R4)  ;SELECT THE DRIVE
3834  016114  004737  023240          JSR     PC,SAVRP          ;SAVE THE RP11 REGISTERS
3835  016120  032737  000400  001334  BIT     #SUWP,$RPDS       ;IS SELECTED UNIT WRITE PROTECTED ?
3836  016126  001001                  BNE     3$                ;BR IF IT IS
3837  016130  104067                  ERROR   67                ;SELECTED UNIT WRITE PROTECT DID
3838                                                            ;NOT SET WITH READ ONLY SET
3839  016132  104401  030542  3$:     TYPE    ,MSG17F           ;'SET READ/WRITE ON DRIVE '
3840  016136  013746  001176          MOV     CHKDRV,-(SP)      ;;SAVE CHKDRV FOR TYPEOUT
3841                                                            ;;TYPE DRIVE NUMBER
3842  016142  104403                  TYPOS                     ;;GO TYPE--OCTAL ASCII
3843  016144     001                  .BYTE   1                 ;;TYPE 1 DIGIT(S)
3844  016145     001                  .BYTE   1                 ;;TYPE LEADING ZEROS
3845  016146  104411                  RDLIN                     ;WAIT FOR 'CR'
3846  016150  012737  016150  001110  4$:     MOV     #4$,$LPERR        ;CHANGE THE LOOP ON ERROR ADDRESS
3847  016156  004737  025654          JSR     PC,CLRP           ;CLEAR THE RP11
3848  016162  113764  001176  000005  MOVB    CHKDRV,RPCS+1(R4)  ;RESELECT THE DRIVE
3849  016170  004737  023240          JSR     PC,SAVRP          ;SAVE THE RP11 REGISTERS
3850  016174  032737  000400  001334  BIT     #SUWP,$RPDS       ;IS SELECTED UNIT WRITE PROTECTED ?
3851  016202  001401                  BEQ     WRSW              ;BR IF NOT
3852  016204  104070                  ERROR   70                ;SELECTED UNIT WRITE PROTECT SET
3853                                                            ;WITH DRIVE WRITE ENABLED
3854
3855                          ;TEST THE SETTING OF WRITE LOCKOUT AND THE 'LOA' SWITCHES ON THE
3856                          ;RP11E CONTROL PANEL.
3857
3858  016206  104401  030577  WRSW:   TYPE    ,MSG17G           ;'SET WRITE LOCKOUT ' ,ETC
3859  016212  104411                  RDLIN                     ;WAIT FOR 'CR' FROM THE OPERATOR
3860  016214  012737  016214  001110  1$:     MOV     #1$,$LPERR        ;CHANGE LOOP ON ERROR ADDRESS
3861  016222  005064  000012          CLR     RPCA(R4)          ;CLEAR THE CYLINDER REGISTER
3862  016226  004737  023240          JSR     PC,SAVRP          ;SAVE THE RP11 REGISTERS
3863  016232  032737  000400  001334  BIT     #SUWP,$RPDS       ;IS SELECTED UNIT WRITE PROTECT SET ?
3864  016240  001001                  BNE     2$                ;BR IF SET
3865  016242  104071                  ERROR   71                ;SELECTED UNIT WRITE PROTECT SHOULD
3866                                                            ;BE SET - RPCA EQUAL TO ZERO
3867  016244  012764  000002  000012  2$:     MOV     #2,RPCA(R4)       ;LOAD A 2 INTO RPCA
3868  016252  004737  023240          JSR     PC,SAVRP          ;SAVE THE RP11 REGISTERS
3869  016256  032737  000400  001334  BIT     #SUWP,$RPDS       ;IS SELECTED UNIT WRITE PROTECTED ?
3870  016264  001401                  BEQ     WRSW1             ;BR IF NOT PROTECTED
3871  016266  104072                  ERROR   72                ;SELECTED UNIT WRITE PROTECT IS SET
3872                                                            ;WITH EQUAL TO 2.
3873
3874                          ;CHECK THE CYLINDER 'LOA' SWITCHES
3875
3876  016270  012700  000002  WRSW1:  MOV     #2,R0             ;INITIALIZE SWITCH PATTERN
3877  016274  104401  030714          TYPE    ,MSG17H           ;'SET FOLLOWING VALUES IN  THE
3878                                                            ;CYLINDER LOA SWITCHES'
3879  016300  104401  031014  1$:     TYPE    ,MSG17I           ;'SET '
3880  016304  006200                  ASR     R0                ;SETUP PATTERN FOR TYPOUT
3881  016306  010046                  MOV     R0,-(SP)          ;;SAVE R0 FOR TYPEOUT
3882                                                            ;;TYPE THE SWITCH VALUE
3883  016310  104403                  TYPOS                     ;;GO TYPE--OCTAL ASCII
3884  016312     003                  .BYTE   3                 ;;TYPE 3 DIGIT(S)
3885  016313     001                  .BYTE   1                 ;;TYPE LEADING ZEROS
3886  016314  006300                  ASL     R0                ;RESTORE THE PATTERN
3887  016316  104411                  RDLIN                     ;WAIT FOR 'CR' FROM THE OPERATOR
```

```
 3888  016320  012737  016320  001110  2$:     MOV     #2$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
 3889  016326  010064  000012          2$:     MOV     R0,RPCA(R4)     ;LOAD PATTERN INTO RPCA
 3890  016332  004737  023240          2$:     JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
 3891  016336  032737  000400  001334  2$:     BIT     #SUWP,$RPDS     ;IS UNIT WRITE PROTECT SET ?
 3892  016344  001001          2$:     BNE     3$              ;BR IF SET
 3893  016346  104073          2$:     ERROR   73              ;UNIT WRITE PROTECT IS NOT SET
 3894                                                          ;RPCA EQUALS CONTENTS OF BOUNDARY
 3895                                                          ;SWITCHES
 3896  016350  062764  000002  000012  3$:     ADD     #2,RPCA(R4)     ;INCREMENT CYLINDER ADDRESS
 3897  016356  012737  016356  001110  4$:     MOV     #4$,$LPERR      ;CHANGE THE LOOP ON ERROR ADDRESS
 3898  016364  004737  023240          4$:     JSR     PC,SAVRP        ;SAVE THE RP11 REGISTERS
 3899  016370  032737  000400  001334  4$:     BIT     #SUWP,$RPDS     ;IS UNIT WRITE PROTECT SET ?
 3900  016376  001401          4$:     BEQ     5$              ;BR IF NOT
 3901  016400  104074          4$:     ERROR   74              ;UNIT WRITE PROTECT IS SET WITH
 3902                                                          ;RPCA ONE GREATER THAN BOUNDARY
 3903                                                          ;SWITCH SETTINGS
 3904  016402  006300          5$:     ASL     R0              ;UPDATE TEST PATTERN
 3905  016404  032700  001000          5$:     BIT     #BIT09,R0       ;IS PATTERN EXCEEDED?
 3906  016410  001733          5$:     BEQ     1$              ;BRANCH IF NOT
 3907
 3908                          ;CHECK DRIVE 'LOA' SWITCHES
 3909
 3910  016412  005037  001176  WRSW2:  CLR     CHKDRV          ;START WITH DRIVE 0
 3911  016416  104401  031037          TYPE    ,MSG17J         ;'RESET THE CYLINDER LOA SWITCHES
 3912                                                          ;'SET THE FOLLOWING VALUES IN THE DRIVE
 3913                                                          ;'LOA SWITCHES '
 3914  016422  012764  000002  000012          MOV     #2,RPCA(R4)     ;LOAD CYLINDER ADDRESS
 3915  016430  113764  001176  000005  1$:     MOVB    CHKDRV,RPCS+1(R4)  ;SELECT THE FIRST DRIVE NUMBER
 3916  016436  104401  031014          TYPE    ,MSG17I         ;'SET SWITCHES TO '
 3917  016442  013746  001176          MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
 3918                                                          ;;TYPE SWITCH SETTING
 3919  016446  104403                  TYPOS                   ;;GO TYPE--OCTAL ASCII
 3920  016450     001                  .BYTE   1               ;;TYPE 1 DIGIT(S)
 3921  016451     001                  .BYTE   1               ;;TYPE LEADING ZEROS
 3922  016452  104411                  RDLIN                   ;WAIT FOR THE OPERATOR TO ENTER A 'CR'
 3923  016454  012737  016454  001110  2$:     MOV     #2$,$LPERR      ;LOOP ON ERROR ADDRESS
 3924  016462  004737  023240          2$:     JSR     PC,SAVRP        ;STORE THE RP11 REGISTERS
 3925  016466  032737  000400  001334  2$:     BIT     #SUWP,$RPDS     ;IS THE SELECTED DRIVE WRITE PROTECTED ?
 3926  016474  001401          2$:     BEQ     3$              ;BR IF NOT
 3927  016476  104075          2$:     ERROR   75              ;'SUWP' SET WHEN DRIVE LOA SWITCHES
 3928                                                          ;'EQUAL THE SELECTED DRIVE'
 3929  016500  005737  001176  3$:     TST     CHKDRV          ;THROUGH HERE THE FIRST TIME ?
 3930  016504  001004          3$:     BNE     4$              ;BR IF NOT
 3931  016506  012737  000001  001176  3$:     MOV     #1,CHKDRV       ;SET FOR DRIVE 1
 3932  016514  000406          3$:     BR      5$              ;CONTINUE
 3933  016516  006337  001176  4$:     ASL     CHKDRV          ;SHIFT TEST ADDRESS
 3934  016522  032737  000010  001176  4$:     BIT     #BIT03,CHKDRV   ;FINISHED WITH ADDRESSES ?
 3935  016530  001025          4$:     BNE     EXIT17          ;BR IF FINISHED
 3936  016532  104401  031014  5$:     TYPE    ,MSG17I         ;'SET SWITCHES TO '
 3937  016536  013746  001176          MOV     CHKDRV,-(SP)    ;;SAVE CHKDRV FOR TYPEOUT
 3938                                                          ;;TYPE SWITCH SETTING
 3939  016542  104403                  TYPOS                   ;;GO TYPE--OCTAL ASCII
 3940  016544     001                  .BYTE   1               ;;TYPE 1 DIGIT(S)
 3941  016545     001                  .BYTE   1               ;;TYPE LEADING ZEROS
 3942  016546  104411                  RDLIN                   ;WAIT FOR THE OPERATOR TO ENTER A 'CR'
 3943  016550  012737  016550  001110  6$:     MOV     #6$,$LPERR      ;CHANGE ERROR LOOP ADDRESS
```

```
3944  016556  004737  023240                  JSR     PC,SAVRP        ;STORE THE REGISTERS
3945  016562  032737  000400  001334          BIT     #SUWP,$RPDS     ;IS THE DRIVE WRITE PROTECTED ?
3946  016570  001001                          BNE     7$              ;BR IF IT IS
3947  016572  104113                          ERROR   113             ;'SUWP' NOT SET WHEN DRIVE LOA SWITCHES
3948                                                                  ;'GREATER THAN SELECTED DRIVE NUMBER
3949  016574  113764  001176  000005  7$:     MOVB    CHKDRV,RPCS+1(R4)  ;SELECT THE NEXT DRIVE
3950  016602  000724                          BR      2$              ;CONTINUE WITH THE TEST
3951  016604  000004                  EXIT17: SCOPE                   ;LOOP ?
3952  016606  104401  031176                  TYPE    ,MSG17K         ;'END OF CONTROL PANEL TEST - RETURN SWITCHES TO NORMAL'
3953  016612  000137  003676                  JMP     START2          ;RETURN TO THE 'CONVERSATION MODE' ENTRANCE
3954
3955                                  .SBTTL  END OF PASS ROUTINE
3956
3957                                  ;;*****************************************************************
3958                                  ;*INCREMENT THE PASS NUMBER ($PASS)
3959                                  ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
3960                                  ;*IF THERES A MONITOR GO TO IT
3961                                  ;*IF THERE ISN'T JUMP TO RESTART
3962
3963  016616                          $EOP:
3964  016616  104401  016624                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3965  016622  000410                          BR      64$             ;;GET OVER THE ASCIZ
3966                                  ;;65$:  .ASCIZ  <15><12><12>/END OF PASS/
3967  016644                          64$:
3968  016644  005737  001312                  TST     @#DRVSEL        ;ANY DRIVES SELECTED
3969  016650  001436                          BEQ     1$              ;NO, BRANCH
3970  016652  104401  016660                  TYPE    ,67$            ;;TYPE ASCIZ STRING
3971  016656  000407                          BR      66$             ;;GET OVER THE ASCIZ
3972                                  ;;67$:  .ASCIZ  / USING DRIVE/
3973  016676                          66$:
3974  016676  013746  001176                  MOV     @#CHKDRV,-(SP)  ;;SAVE @#CHKDRV FOR TYPEOUT
3975  016702  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
3976  016704     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
3977  016705     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
3978  016706  104401  016714                  TYPE    ,69$            ;;TYPE ASCIZ STRING
3979  016712  000412                          BR      68$             ;;GET OVER THE ASCIZ
3980                                  ;;69$:  .ASCIZ  /   ERRORS DETECTED=/
3981  016740                          68$:
3982  016740  013746  001112                  MOV     @#$ERTTL,-(SP)  ;;SAVE @#$ERTTL FOR TYPEOUT
3983  016744  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3984  016746  005037  001112          1$:     CLR     @#$ERTTL        ;ZERO ERROR TOTAL
3985  016752  005037  001102                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
3986  016756  005037  001162                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
3987  016762  005237  001100                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
3988  016766  042737  100000  001100          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
3989  016774  005327                          DEC     (PC)+           ;;LOOP?
3990  016776  000001                  $EOPCT: .WORD   1
3991  017000  003027                          BGT     $DOAGN          ;;YES
3992  017002  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
3993  017004  000001                  $ENDCT: .WORD   1
3994  017006  016776                          $EOPCT
3995  017010  104401  017016                  TYPE    ,65$            ;;TYPE ASCIZ STRING
3996  017014  000407                          BR      64$             ;;GET OVER THE ASCIZ
3997                                  ;;65$:  .ASCIZ  <15><12>/END OF TEST/
3998  017034                          64$:
3999  017034  104401  017064                  TYPE    ,$ENULL         ;TYPE NULL CHARACTER
```

```
 4000  017040  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 4001  017044  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 4002  017046  000005                          RESET                   ;;CLEAR THE WORLD
 4003  017050  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
 4004  017052  000240                          NOP                     ;;SAVE ROOM
 4005  017054  000240                          NOP                     ;;FOR
 4006  017056  000240                          NOP                     ;;ACT11
 4007  017060                          $DOAGN:
 4008  017060  000137                          JMP     @(PC)+          ;;RETURN
 4009  017062  004762                  $RTNAD: .WORD   RESTART
 4010  017064     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
 4011          017070                          .EVEN
 4012
 4013
 4014
 4015                                  .SBTTL  *** SUBROUTINES ***
 4016
 4017                                  .SBTTL  ERROR HANDLER ROUTINE
 4018
 4019                                  ;;**********************************************************************
 4020                                  ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 4021                                  ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 4022                                  ;*AND GO TO TYPERR ON ERROR
 4023                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 4024                                  ;*SW15=1         HALT ON ERROR
 4025                                  ;*SW13=1         INHIBIT ERROR TYPEOUTS
 4026                                  ;*SW10=1         BELL ON ERROR
 4027                                  ;*SW09=1         LOOP ON ERROR
 4028                                  ;*CALL
 4029                                  ;*       ERROR   N       ,;ERROR=EMT AND N=ERROR ITEM NUMBER
 4030
 4031  017070                          $ERROR:
 4032  017070  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 4033  017072  105237  001103          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
 4034  017076  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
 4035  017100  013777  001102  162034        MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
 4036  017106  032777  002000  162024        BIT     #BIT10,@SWR     ;;BELL ON ERROR?
 4037  017114  001402                          BEQ     1$              ;;NO - SKIP
 4038  017116  104401  001166                  TYPE    ,$BELL          ;;RING BELL
 4039  017122  005237  001112          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
 4040  017126  011637  001116                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
 4041  017132  162737  000002  001116        SUB     #2,$ERRPC
 4042  017140  117737  161752  001114        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
 4043  017146  032777  020000  161764        BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
 4044  017154  001004                          BNE     20$             ;;SKIP TYPEOUTS
 4045  017156  004737  017254                  JSR     PC,TYPERR       ;;GO TO USER ERROR ROUTINE
 4046  017162  104401  001173                  TYPE    ,$CRLF
 4047  017166                          20$:
 4048  017166  005777  161746          2$:     TST     @SWR            ;;HALT ON ERROR
 4049  017172  100002                          BPL     3$              ;;SKIP IF CONTINUE
 4050  017174  000000                          HALT                    ;;HALT ON ERROR!
 4051  017176  104407                          CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
 4052  017200  032777  001000  161732  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
 4053  017206  001402                          BEQ     4$              ;;BR IF NO
 4054  017210  013716  001110                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
 4055  017214  005737  001164          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
```

```
 4056  017220  001402                        BEQ     5$              ;;BR IF NONE
 4057  017222  013716  001164                MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
 4058  017226                          5$:
 4059  017226  022737  017050  000042        CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
 4060  017234  001001                        BNE     6$              ;;BRANCH IF NO
 4061  017236  000000                        HALT                    ;;YES
 4062  017240                          6$:
 4063  017240  023737  000042  000046        CMP     @#42,@#46               ;ACT11 AUTO MODE?
 4064  017246  001001                        BNE     12$                     ;NO, RETURN
 4065  017250  000000                        HALT                            ;HALT ON ERROR
 4066  017252  000002                  12$:  RTI                             ;RETURN
 4067
 4068                                    ;;*****************************************************************
 4069
 4070                                    ;THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE
 4071                                    ;WHICH ERROR IS TO BE REPORTED, IT THEN OBTAINS, FROM THE "ERROR
 4072                                    ;TABLE" ($ERRTB), AND REPORTS THE APPROPIATE INFORMATION
 4073                                    ;CONCERNING THE ERROR.
 4074                                    ;CALL
 4075                                    ;       JSR     PC,TYPERR
 4076                                    ;       RETURN
 4077
 4078  017254  104412                  TYPERR: SAVREG                  ;SAVE R0-R5
 4079  017256  032777  020000  161654        BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS ?
 4080  017264  001105                        BNE     20$             ;BR IF YES
 4081  017266  005037  001160                CLR     $TMP0           ;CLEAR LOCATION FOR TEST NUMBER
 4082  017272  113737  001102  001160        MOVB    $TSTNM,$TMP0    ;LOAD TEST NUMBER FOR TYPEOUT
 4083  017300  005000                        CLR     R0              ;CLEAR R0 FOR ERROR NUMBER
 4084  017302  113700  001114                MOVB    $ITEMB,R0       ;ERROR NUMBER
 4085  017306  005300                        DEC     R0              ;FORM INDEX FOR ERROR TABLE
 4086  017310  006300                        ASL     R0
 4087  017312  006300                        ASL     R0
 4088  017314  006300                        ASL     R0
 4089  017316  062700  002514          1$:   ADD     #$ERRTB,R0      ;FORM ADDRESS
 4090  017322  012037  017336                MOV     (R0)+,2$        ;GET ERROR MESSAGE (EM) POINTER
 4091  017326  001404                        BEQ     3$              ;BRANCH IF THERE ISN'T ONE
 4092  017330  104401  001173                TYPE    ,$CRLF          ;"CARRIAGE RETURN - LINE FEED
 4093  017334  104401                        TYPE
 4094  017336  000000                  2$:   .WORD   0               ;"EM" POINTER GOES HERE
 4095  017340  012037  017354          3$:   MOV     (R0)+,4$        ;PICK UP DATA HEADER (DH) POINTER
 4096  017344  001404                        BEQ     5$              ;BRANCH IF NONE
 4097  017346  104401  001173                TYPE    ,$CRLF          ;CARRIAGE RETURN-LINE FEED
 4098  017352  104401                        TYPE
 4099  017354  000000                  4$:   .WORD   0               ;"DH" POINTER GOES HERE
 4100  017356  012001                  5$:   MOV     (R0)+,R1        ;PICKUP DATA TABLE (DT) POINTER
 4101  017360  001447                        BEQ     20$             ;BRANCH IF NONE
 4102  017362  005005                        CLR     R5              ;SET INDENT SWITCH
 4103  017364  012000                        MOV     (R0)+,R0        ;DATA FORMAT (DF) POINTER
 4104  017366  012002                        MOV     (R0)+,R2        ;NUMBER OF DH'S TO TYPE
 4105  017370  001443                        BEQ     20$             ;BRANCH IF DH NUMBER IS 0
 4106  017372  005105                        COM     R5              ;NO INDENT
 4107  017374  112003                 10$:   MOVB    (R0)+,R3        ;NUMBER OF DATA WORDS TO TYPE
 4108  017376  001440                        BEQ     20$             ;BR IF NO WORDS TO TYPE
 4109  017400  112004                        MOVB    (R0)+,R4        ;AND HOW TO TYPE THEM
 4110  017402  104401  001173                TYPE    ,$CRLF          ;CR-LF
 4111  017406  005705                        TST     R5              ;INDENT ?
```

```
 4112  017410  001002                  BNE     11$             ;BR IF NOT
 4113  017412  104401  027430          TYPE    ,BLNKS2         ;BLANKS
 4114  017416  006004          11$:    ROR     R4              ;OCTAL OR DECIMAL?
 4115  017420  103403                  BCS     12$             ;DECIMAL--BRANCH
 4116  017422  013146                  MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
 4117  017424  104402                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4118  017426  000402                  BR      13$
 4119  017430                  12$:
 4120  017430  013146                  MOV     @(R1)+,-(SP)    ;;SAVE @(R1)+ FOR TYPEOUT
 4121  017432  104405                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 4122  017434  005303          13$:    DEC     R3              ;MORE NUMBERS TO TYPE?
 4123  017436  001403                  BEQ     14$             ;NO--BRANCH
 4124  017440  104401  027430          TYPE    ,BLNKS2         ;YES--TYPE SEPARATORS
 4125  017444  000764                  BR      11$             ;LOOP
 4126  017446  005302          14$:    DEC     R2              ;MORE DH'S?
 4127  017450  003413                  BLE     20$             ;NO--BRANCH
 4128  017452  104401  001173          TYPE    ,$CRLF          ;YES--START A NEW LINE
 4129  017456  005105                  COM     R5              ;INDENT?
 4130  017460  001002                  BNE     15$             ;NO--BRANCH
 4131  017462  104401  027430          TYPE    ,BLNKS2         ;YES--TYPE SPACES
 4132  017466  012037  017474  15$:    MOV     (R0)+,16$       ;GET NEXT DH
 4133  017472  104401                  TYPE                    ;AND TYPE IT
 4134  017474  000000          16$:    .WORD   0               ;DH POINTER GOES HERE
 4135  017476  000736                  BR      10$             ;LOOP
 4136  017500  104413          20$:    RESREG                  ;RESTORE R0-R5
 4137  017502  000207                  RTS     PC              ;RETURN
 4138
 4139                          .SBTTL  TYPE ROUTINE
 4140
 4141                          ;;**********************************************************************
 4142                          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
 4143                          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
 4144                          ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
 4145                          ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
 4146                          ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
 4147                          ;*
 4148                          ;*CALL:
 4149                          ;*1) USING A TRAP INSTRUCTION
 4150                          ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
 4151                          ;*OR
 4152                          ;*      TYPE
 4153                          ;*      MESADR
 4154                          ;*
 4155
 4156  017504  105737  001157  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
 4157  017510  100002                  BPL     1$              ;;BR IF YES
 4158  017512  000000                  HALT                    ;;HALT HERE IF NO TERMINAL
 4159  017514  000407                  BR      3$              ;;LEAVE
 4160  017516  010046          1$:     MOV     R0,-(SP)        ;;SAVE R0
 4161  017520  017600  000002          MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 4162  017524  112046          2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 4163  017526  001005                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 4164  017530  005726                  TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 4165  017532  012600          60$:    MOV     (SP)+,R0        ;;RESTORE R0
 4166  017534  062716  000002  3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 4167  017540  000002                  RTI                     ;;RETURN
```

```
 4168  017542  122716  000011           4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 4169  017546  001430                            BEQ     8$
 4170  017550  122716  000200                    CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
 4171  017554  001006                            BNE     5$
 4172  017556  005726                            TST     (SP)+           ;;POP  <CR><LF> EQUIV
 4173  017560  104401                            TYPE                    ;;TYPE A CR AND LF
 4174  017562  001173                            $CRLF
 4175  017564  105037  017720                    CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
 4176  017570  000755                            BR      2$              ;;GET NEXT CHARACTER
 4177  017572  004737  017654           5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 4178  017576  123726  001156           6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 4179  017602  001350                            BNE     2$              ;;IF NO GO GET NEXT CHAR.
 4180  017604  013746  001154                    MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
 4181                                                                    ;;AND THE NULL CHAR.
 4182  017610  105366  000001           7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 4183  017614  002770                            BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 4184  017616  004737  017654                    JSR     PC,$TYPEC       ;;GO TYPE A NULL
 4185  017622  105337  017720                    DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
 4186  017626  000770                            BR      7$              ;;LOOP
 4187
 4188                                    ;HORIZONTAL TAB PROCESSOR
 4189
 4190  017630  112716  000040           8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 4191  017634  004737  017654           9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 4192  017640  132737  000007  017720           BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 4193  017646  001372                            BNE     9$              ;;TAB STOP
 4194  017650  005726                            TST     (SP)+           ;;POP SPACE OFF STACK
 4195  017652  000724                            BR      2$              ;;GET NEXT CHARACTER
 4196  017654  105777  161270           $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 4197  017660  100375                            BPL     $TYPEC
 4198  017662  116677  000002  161262           MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 4199  017670  122766  000015  000002           CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 4200  017676  001003                            BNE     1$              ;;BRANCH IF NO
 4201  017700  105037  017720                    CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 4202  017704  000406                            BR      $TYPEX          ;;EXIT
 4203  017706  122766  000012  000002  1$:      CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 4204  017714  001402                            BEQ     $TYPEX          ;;BRANCH IF YES
 4205  017716  105227                            INCB    (PC)+           ;;COUNT THE CHARACTER
 4206  017720  000000           $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
 4207  017722  000207           $TYPEX: RTS     PC
 4208
 4209
 4210                                    .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 4211
 4212                                    ;;**********************************************************************
 4213                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 4214                                    ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 4215                                    ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 4216                                    ;*CALL:
 4217                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 4218                                    ,*      TYPOS                   ;;CALL FOR TYPEOUT
 4219                                    ,*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
 4220                                    ;*      .BYTE   M               ;;M=1 OR 0
 4221                                    ,*                                      ;;1=TYPE LEADING ZEROS
 4222                                    ,*                                      ;;0=SUPPRESS LEADING ZEROS
 4223                                    ,*
```

```
 4224                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
 4225                                   ;*$TYPOS OR $TYPOC
 4226                                   ;*CALL:
 4227                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 4228                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
 4229                                   ;*
 4230                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
 4231                                   ;*CALL:
 4232                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 4233                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
 4234
 4235  017724  017646  000000           $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
 4236  017730  116637  000001  020147           MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
 4237  017736  112637  020151                   MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
 4238  017742  062716  000002                   ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
 4239  017746  000406                           BR      $TYPON
 4240  017750  112737  000001  020147  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
 4241  017756  112737  000006  020151           MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
 4242  017764  112737  000005  020146  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
 4243  017772  010346                           MOV     R3,-(SP)        ;;SAVE R3
 4244  017774  010446                           MOV     R4,-(SP)        ;;SAVE R4
 4245  017776  010546                           MOV     R5,-(SP)        ;;SAVE R5
 4246  020000  113704  020151                   MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
 4247  020004  005404                           NEG     R4
 4248  020006  062704  000006                   ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
 4249  020012  110437  020150                   MOVB    R4,$OMODE       ;;SAVE IT FOR USE
 4250  020016  113704  020147                   MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
 4251  020022  016605  000012                   MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
 4252  020026  005003                           CLR     R3              ;;CLEAR THE OUTPUT WORD
 4253  020030  006105                   1$:     ROL     R5              ;;ROTATE MSB INTO "C"
 4254  020032  000404                           BR      3$              ;;GO DO MSB
 4255  020034  006105                   2$:     ROL     R5              ;;FORM THIS DIGIT
 4256  020036  006105                           ROL     R5
 4257  020040  006105                           ROL     R5
 4258  020042  010503                           MOV     R5,R3
 4259  020044  006103                   3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
 4260  020046  105337  020150                   DECB    $OMODE          ;;TYPE THIS DIGIT?
 4261  020052  100016                           BPL     7$              ;;BR IF NO
 4262  020054  042703  177770                   BIC     #177770,R3      ;;GET RID OF JUNK
 4263  020060  001002                           BNE     4$              ;;TEST FOR 0
 4264  020062  005704                           TST     R4              ;;SUPPRESS THIS 0?
 4265  020064  001403                           BEQ     5$              ;;BR IF YES
 4266  020066  005204                   4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 4267  020070  052703  000060                   BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 4268  020074  052703  000040           5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 4269  020100  110337  020144                   MOVB    R3,8$           ;;SAVE FOR TYPING
 4270  020104  104401  020144                   TYPE    ,8$             ;;GO TYPE THIS DIGIT
 4271  020110  105337  020146           7$:     DECB    $OCNT           ;;COUNT BY 1
 4272  020114  003347                           BGT     2$              ;;BR IF MORE TO DO
 4273  020116  002402                           BLT     6$              ;;BR IF DONE
 4274  020120  005204                           INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 4275  020122  000744                           BR      2$              ;;GO DO THE LAST DIGIT
 4276  020124  012605                   6$:     MOV     (SP)+,R5        ;;RESTORE R5
 4277  020126  012604                           MOV     (SP)+,R4        ;;RESTORE R4
 4278  020130  012603                           MOV     (SP)+,R3        ;;RESTORE R3
 4279  020132  016666  000002  000004           MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
```

```
4280  020140  012616                          MOV     (SP)+,(SP)
4281  020142  000002                          RTI                             ;;RETURN
4282  020144     000          8$:             .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
4283  020145     000                          .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
4284  020146     000          $OCNT:          .BYTE   0                       ;;OCTAL DIGIT COUNTER
4285  020147     000          $OFILL:         .BYTE   0                       ;;ZERO FILL SWITCH
4286  020150  000000          $OMODE:         .WORD   0                       ;;NUMBER OF DIGITS TO TYPE
4287
4288                          .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
4289
4290                          ;;**********************************************************************
4291                          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
4292                          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
4293                          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
4294                          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
4295                          ;*REPLACED WITH SPACES.
4296                          ;*CALL:
4297                          ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
4298                          ;*      TYPDS                   ;;GO TO THE ROUTINE
4299
4300  020152                  $TYPDS:
4301  020152  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
4302  020154  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
4303  020156  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
4304  020160  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
4305  020162  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
4306  020164  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
4307  020170  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
4308  020174  100004                          BPL     1$              ;;BR IF INPUT IS POS.
4309  020176  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
4310  020200  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
4311  020206  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
4312  020210  012703  020366                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
4313  020214  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
4314  020220  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
4315  020222  016001  020356                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
4316  020226  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
4317  020230  002402                          BLT     4$              ;;BR IF DONE
4318  020232  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
4319  020234  000774                          BR      3$
4320  020236  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
4321  020240  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
4322  020242  001002                          BNE     5$              ;;FALL THROUGH IF 0
4323  020244  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
4324  020246  100407                          BMI     7$              ;;BR IF YES
4325  020250  106316                  5$:     ASLB    (SP)            ;;MSD?
4326  020252  103003                          BCC     6$              ;;BR IF NO
4327  020254  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
4328  020262  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
4329  020266  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
4330  020272  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
4331  020274  005720                          TST     (R0)+           ;;JUST INCREMENTING
4332  020276  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
4333  020302  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
4334  020304  003002                          BGT     8$              ;;GO TO EXIT
4335  020306  010502                          MOV     R5,R2           ;;GET THE LSD
```

```
 4336  020310  000764                        BR      6$              ;;GO CHANGE TO ASCII
 4337  020312  105726                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 4338  020314  100003                        BPL     9$              ;;BR IF NO
 4339  020316  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 4340  020324  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 4341  020326  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
 4342  020330  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 4343  020332  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 4344  020334  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 4345  020336  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 4346  020340  104401  020366                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 4347  020344  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 4348  020352  012616                        MOV     (SP)+,(SP)
 4349  020354  000002                        RTI                     ;;RETURN TO USER
 4350  020356  023420                $DTBL:  10000.
 4351  020360  001750                        1000.
 4352  020362  000144                        100.
 4353  020364  000012                        10.
 4354  020366  000004                $DBLK:  .BLKW   4
 4355
 4356                                  .SBTTL  TTY INPUT ROUTINE
 4357
 4358                                  ;;**************************************************************************
 4359                                  .ENABL  LSB
 4360  020376  000000                $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
 4361  020400  000000                $TKQIN: .WORD   0               ;;INPUT POINTER
 4362  020402  000000                $TKQOUT: .WORD  0               ;;OUTPUT POINTER
 4363  020404  000012                $TKQSRT: .BLKB  10.             ;;TTY KEYBOARD QUEUE
 4364          020416                $TKQEND=.
 4365
 4366                                  ;*TK INITIALIZE ROUTINE
 4367                                  ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
 4368                                  ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
 4369                                  ;
 4370                                  ;*CALL:
 4371                                  ;*      JSR     PC,$TKINT
 4372                                  ;*      RETURN
 4373                                  ;
 4374  020416  005037  020376        $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
 4375  020422  012737  020404  020400        MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
 4376  020430  013737  020400  020402        MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
 4377  020436  012737  020466  000060        MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
 4378  020444  012737  000200  000062        MOV     #200,@#TKVEC+2  ;;"BR" LEVEL 4
 4379  020452  005777  160470                TST     @$TKB           ;;CLEAR DONE FLAG
 4380  020456  012777  000100  160460        MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
 4381  020464  000207                        RTS     PC              ;;RETURN TO CALLER
 4382
 4383                                  ;*TK SERVICE ROUTINE
 4384                                  ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
 4385                                  ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
 4386                                  ;*IT IN THE QUEUE.
 4387                                  ;*IF THE CHARACTER IS A "CONTROL-C" ( C) $TKINT IS CALLED AND
 4388                                  ;*UPON RETURN EXIT IS MADE TO THE "CONTROL-C" RESTART ADDRESS (START2)
 4389                                  ;
 4390  020466  117746  160454        $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
 4391  020472  042716  177600                BIC     # C177,(SP)    ;;STRIP THE JUNK
```

```
 4392  020476  021627  000003            CMP     (SP),#3         ;;IS IT A CONTROL C?
 4393  020502  001007                    BNE     1$              ;;BRANCH IF NO
 4394  020504  104401  021604            TYPE    ,$CNTLC         ;;TYPE A CONTROL-C ( C)
 4395  020510  004737  020416            JSR     PC,$TKINT       ;;INIT THE KEYBOARD
 4396  020514  005726                    TST     (SP)+           ;;CLEAN UP STACK
 4397  020516  000137  003676            JMP     START2          ;;CONTROL C RESTART
 4398  020522  021627  000007    1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
 4399  020526  001004                    BNE     2$              ;;BRANCH IF NO
 4400  020530  022737  000176  001140    CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
 4401  020536  001500                    BEQ     6$              ;;GO TO SWR CHANGE
 4402
 4403  020540                    2$:
 4404  020540  022737  000012  020376    CMP     #10.,$TKCNT     ;;IS THE QUEUE FULL?
 4405  020546  001004                    BNE     3$              ;;BRANCH IF NO
 4406  020550  104401  001166            TYPE    ,$BELL          ;;RING THE TTY BELL
 4407  020554  005726                    TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
 4408  020556  000451                    BR      5$              ;;EXIT
 4409  020560  021627  000023    3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
 4410  020564  001021                    BNE     32$             ;;BRANCH IF NO
 4411  020566  005077  160352            CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 4412  020572  005726                    TST     (SP)+           ;;CLEAN CHAR OFF STACK
 4413  020574  105777  160344    31$:    TSTB    @$TKS           ;;WAIT FOR A CHAR
 4414  020600  100375                    BPL     31$             ;;LOOP UNTIL ITS THERE
 4415  020602  117746  160340            MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
 4416  020606  042716  177600            BIC     # C177,(SP)     ;;MAKE IT 7-BIT ASCII
 4417  020612  022627  000021            CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
 4418  020616  001366                    BNE     31$             ;;BRANCH IF NO
 4419  020620  012777  000100  160316    MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
 4420  020626  000002                    RTI                     ;;RETURN
 4421  020630  005237  020376    32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
 4422  020634  021627  000140            CMP     (SP),#140       ;;IS IT UPPER CASE?
 4423  020640  002405                    BLT     4$              ;;BRANCH IF YES
 4424  020642  021627  000175            CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
 4425  020646  003002                    BGT     4$              ;;BRANCH IF YES
 4426  020650  042716  000040            BIC     #40,(SP)        ;;MAKE IT UPPER CASE
 4427  020654  112677  177520    4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
 4428  020660  005237  020400            INC     $TKQIN          ;;UPDATE THE POINTER
 4429  020664  023727  020400  020416    CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
 4430  020672  001003                    BNE     5$              ;;BRANCH IF NO
 4431  020674  012737  020404  020400    MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
 4432  020702  000002            5$:     RTI                     ;;RETURN
 4433
 4434                            ;;**************************************************************
 4435                            ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
 4436                            ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
 4437                            ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
 4438                            ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
 4439  020704  022737  000176  001140  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
 4440  020712  001124                    BNE     15$             ;;EXIT IF NOT
 4441  020714  105777  160224            TSTB    @$TKS           ;;IS A CHAR WAITING?
 4442  020720  100121                    BPL     15$             ;;IF NOT, EXIT
 4443  020722  117746  160220            MOVB    @$TKB,-(SP)     ;;YES
 4444  020726  042716  177600            BIC     # C177,(SP)     ;;MAKE IT  7-BIT ASCII
 4445  020732  021627  000007            CMP     (SP),#7         ;;IS IT A CONTROL-G?
 4446  020736  001300                    BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
 4447                                                            ;;AND EXIT
```

```
 4448
 4449                                ;;*********************************************************************
 4450                                ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
 4451                                ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
 4452                                ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
 4453  020740  123727  001134  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
 4454  020746  001674                        BEQ     2$              ;;BRANCH IF YES
 4455  020750  005726                        TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
 4456  020752  004737  020416                JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
 4457  020756  005077  160162                CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
 4458  020762  112737  000001  001135        MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR
 4459
 4460  020770  104401  021616                TYPE    ,$CNTLG         ;;ECHO THE CONTROL-G ( G)
 4461  020774  104401  021623  $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
 4462  021000  013746  000176                MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
 4463  021004  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 4464  021006  104401  021634                TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
 4465  021012  005046                19$:     CLR     -(SP)           ;;CLEAR COUNTER
 4466  021014  005046                        CLR     -(SP)           ;;THE NEW SWR
 4467  021016  105777  160122        7$:     TSTB    @$TKS           ;;CHAR THERE?
 4468  021022  100375                        BPL     7$              ;;IF NOT TRY AGAIN
 4469
 4470  021024  117746  160116                MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
 4471  021030  042716  177600                BIC     # C177,(SP)     ;;MAKE IT 7-BIT ASCII
 4472
 4473  021034  021627  000003                CMP     (SP),#3         ;;IS IT A CONTROL-C?
 4474  021040  001015                        BNE     9$              ;;BRANCH IF NOT
 4475  021042  104401  021604                TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C ( C)
 4476  021046  062706  000006                ADD     #6,SP           ;;CLEAN UP STACK
 4477  021052  123727  001135  000001        CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
 4478  021060  001003                        BNE     8$              ;;BRANCH IF NO
 4479  021062  012777  000100  160054        MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
 4480  021070  000137  003676        8$:     JMP     START2          ;;CONTROL-C RESTART
 4481
 4482
 4483  021074  021627  000025        9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
 4484  021100  001005                        BNE     10$             ;;BRANCH IF NOT
 4485  021102  104401  021611                TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U ( U)
 4486  021106  062706  000006        20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
 4487  021112  000737                        BR      19$             ;;LET'S TRY IT AGAIN
 4488
 4489
 4490  021114  021627  000015        10$:    CMP     (SP),#15        ;;IS IT A <CR>?
 4491  021120  001022                        BNE     16$             ;;BRANCH IF NO
 4492  021122  005766  000004                TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
 4493  021126  001403                        BEQ     11$             ;;BRANCH IF YES
 4494  021130  016677  000002  160002        MOV     2(SP),@SWR      ;;SAVE NEW SWR
 4495  021136  062706  000006        11$:    ADD     #6,SP           ;;CLEAR UP STACK
 4496  021142  104401  001173        14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
 4497  021146  123727  001135  000001        CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
 4498  021154  001003                        BNE     15$             ;;BRANCH IF NOT
 4499  021156  012777  000100  157760        MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
 4500  021164  000002                15$:    RTI                     ;;RETURN
 4501  021166  004737  017654        16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
 4502  021172  021627  000060                CMP     (SP),#60        ;;CHAR < 0?
 4503  021176  002420                        BLT     18$             ;;BRANCH IF YES
```

```
 4504  021200  021627  000067            CMP     (SP),#67        ;;CHAR > 7?
 4505  021204  003015                     BGT     18$             ;;BRANCH IF YES
 4506  021206  042726  000060            BIC     #60,(SP)+       ;;STRIP-OFF ASCII
 4507  021212  005766  000002            TST     2(SP)           ;;IS THIS THE FIRST CHAR
 4508  021216  001403                     BEQ     17$             ;;BRANCH IF YES
 4509  021220  006316                     ASL     (SP)            ;;NO, SHIFT PRESENT
 4510  021222  006316                     ASL     (SP)            ;;   CHAR OVER TO MAKE
 4511  021224  006316                     ASL     (SP)            ;;   ROOM FOR NEW ONE.
 4512  021226  005266  000002    17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
 4513  021232  056616  177776            BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
 4514  021236  000667                     BR      7$              ;;GET THE NEXT ONE
 4515  021240  104401  001172    18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
 4516  021244  000720                     BR      20$             ;;SIMULATE CONTROL-U
 4517                          .DSABL  LSB
 4518
 4519
 4520                          ;;*****************************************************************
 4521                          ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
 4522                          ;*CALL:
 4523                          ;*      RDCHR                   ;;GET A CHARACTER FROM THE QUEUE
 4524                          ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
 4525                          ;*                              ;;WITH PARITY BIT STRIPPED OFF
 4526                          ;
 4527
 4528  021246  011646    $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
 4529  021250  016666  000004  000002    MOV     4(SP),2(SP)     ;;THE PS
 4530  021256  005066  000004            CLR     4(SP)           ;;GET READY FOR A CHARACTER
 4531  021262  005046                     CLR     -(SP)           ;;PUT NEW PS ON STACK
 4532  021264  012746  021272            MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
 4533  021270  000002                     RTI                     ;;POP NEW PC AND PS
 4534  021272                    64$:
 4535  021272  005737  020376    1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
 4536  021276  001775                     BEQ     1$
 4537  021300  005337  020376            DEC     $TKCNT          ;;DECREMENT THE COUNTER
 4538  021304  117766  177072  000004    MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
 4539  021312  005237  020402            INC     $TKQOUT         ;;UPDATE THE POINTER
 4540  021316  023727  020402  020416    CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
 4541  021324  001003                     BNE     2$              ;;BRANCH IF NO
 4542  021326  012737  020404  020402    MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
 4543  021334  000002            2$:     RTI                     ;;RETURN
 4544                          ;;*****************************************************************
 4545                          ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
 4546                          ;*CALL:
 4547                          ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
 4548                          ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
 4549                          ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
 4550
 4551  021336  010346    $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
 4552  021340  005046            CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
 4553  021342  012703  021572    1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
 4554  021346  022703  021604    2$:     CMP     #$TTYIN+10.,R3  ;;BUFFER FULL?
 4555  021352  101456                     BLOS    4$              ;;BR IF YES
 4556  021354  104410                     RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
 4557  021356  112613                     MOVB    (SP)+,(R3)      ;;GET CHARACTER
 4558  021360  122713  000177    10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
 4559  021364  001022                     BNE     5$              ;;BR IF NO
```

```
4560  021366  005716                        TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
4561  021370  001007                        BNE     6$              ;;BR IF NO
4562  021372  112737  000134  021570        MOVB    #' ,9$          ;;TYPE A BACK SLASH
4563  021400  104401  021570                TYPE    ,9$
4564  021404  012716  177777                MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
4565  021410  005303                6$:     DEC     R3              ;;BACKUP BY ONE
4566  021412  020327  021572                CMP     R3,#$TTYIN      ;;STACK EMPTY?
4567  021416  103434                        BLO     4$              ;;BR IF YES
4568  021420  111337  021570                MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
4569  021424  104401  021570                TYPE    ,9$             ;;GO TYPE
4570  021430  000746                        BR      2$              ;;GO READ ANOTHER CHAR.
4571  021432  005716                5$:     TST     (SP)            ;;RUBOUT KEY SET?
4572  021434  001406                        BEQ     7$              ;;BR IF NO
4573  021436  112737  000134  021570        MOVB    #' ,9$          ;;TYPE A BACK SLASH
4574  021444  104401  021570                TYPE    ,9$
4575  021450  005016                        CLR     (SP)            ;;CLEAR THE RUBOUT KEY
4576  021452  122713  000025        7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
4577  021456  001003                        BNE     8$              ;;BR IF NO
4578  021460  104401  021611                TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
4579  021464  000726                        BR      1$              ;;GO START OVER
4580  021466  122713  000022        8$:     CMPB    #22,(R3)        ;;IS CHARACTER A " R"?
4581  021472  001011                        BNE     3$              ;;BRANCH IF NO
4582  021474  105013                        CLRB    (R3)            ;;CLEAR THE CHARACTER
4583  021476  104401  001173                TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
4584  021502  104401  021572                TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
4585  021506  000717                        BR      2$              ;;GO PICKUP ANOTHER CHACTER
4586  021510  104401  001172        4$:     TYPE    ,$QUES          ;;TYPE A '?'
4587  021514  000712                        BR      1$              ;;CLEAR THE BUFFER AND LOOP
4588  021516  111337  021570        3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
4589  021522  104401  021570                TYPE    ,9$
4590  021526  122723  000015                CMPB    #15,(R3)+       ;;CHECK FOR RETURN
4591  021532  001305                        BNE     2$              ;;LOOP IF NOT RETURN
4592  021534  105063  177777                CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
4593  021540  104401  001174                TYPE    ,$LF            ;;TYPE A LINE FEED
4594  021544  005726                        TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
4595  021546  012603                        MOV     (SP)+,R3        ;;RESTORE R3
4596  021550  011646                        MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
4597  021552  016666  000004  000002        MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
4598  021560  012766  021572  000004        MOV     #$TTYIN,4(SP)
4599  021566  000002                        RTI                     ;;RETURN
4600  021570     000                9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
4601  021571     000                        .BYTE   0               ;;TERMINATOR
4602  021572  000012                $TTYIN: .BLKB   10.             ;;RESERVE 10. BYTES FOR TTY INPUT
4603  021604  041536  005015     000 $CNTLC: .ASCIZ  / C/<15><12>    ;;CONTROL "C"
4604  021611     136  006525  000012 $CNTLU: .ASCIZ  / U/<15><12>    ;;CONTROL "U"
4605  021616  043536  005015     000 $CNTLG: .ASCIZ  / G/<15><12>    ;;CONTROL "G"
4606  021623     015  051412  051127 $MSWR:  .ASCIZ  <15><12>/SWR = /
4607  021630  036440  000040
4608  021634  020040  042516  020127 $MNEW:  .ASCIZ  /  NEW = /
4609  021642  020075     000
4610          021646                .EVEN
4611
4612                                .SBTTL  SCOPE HANDLER ROUTINE
4613
4614                                ;;**************************************************************
4615                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
```

```
 4616                                   ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
 4617                                   ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
 4618                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 4619                                   ;*SW14=1        LOOP ON TEST
 4620                                   ;*SW11=1        INHIBIT ITERATIONS
 4621                                   ;*SW09=1        LOOP ON ERROR
 4622                                   ;*CALL
 4623                                   ;*      SCOPE           ;;SCOPE=IOT
 4624
 4625  021646                           $SCOPE:
 4626  021646  104407                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
 4627  021650  032777  040000  157262  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
 4628  021656  001101                          BNE     $OVER           ;;YES IF SW14=1
 4629                                   ;#####START OF CODE FOR THE XOR TESTER#####
 4630  021660  000416                   $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
 4631                                                                   ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 4632  021662  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 4633  021666  012737  021706  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
 4634  021674  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
 4635  021700  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 4636  021704  000453                          BR      $SVLAD          ;;GO TO THE NEXT TEST
 4637  021706  022626                   5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
 4638  021710  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 4639  021714  000413                          BR      7$              ;;LOOP ON THE PRESENT TEST
 4640  021716                           6$:;#####END OF CODE FOR THE XOR TESTER#####
 4641  021716  105737  001103          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
 4642  021722  001421                          BEQ     3$              ;;BR IF NO
 4643  021724  123737  001115  001103          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
 4644  021732  101015                          BHI     3$              ;;BR IF NO
 4645  021734  032777  001000  157176          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
 4646  021742  001404                          BEQ     4$              ;;BR IF NO
 4647  021744  013737  001110  001106  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
 4648  021752  000443                          BR      $OVER
 4649  021754  105037  001103          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 4650  021760  005037  001162                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 4651  021764  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
 4652  021766  032777  004000  157144  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
 4653  021774  001011                          BNE     1$              ;;BR IF YES
 4654  021776  005737  001100                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
 4655  022002  001406                          BEQ     1$              ;;       INHIBIT ITERATIONS
 4656  022004  005237  001104                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
 4657  022010  023737  001162  001104          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
 4658  022016  002021                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
 4659  022020  012737  000001  001104  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
 4660  022026  013737  022076  001162          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
 4661  022034  105237  001102          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
 4662  022040  011637  001106                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 4663  022044  011637  001110                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 4664  022050  005037  001164                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 4665  022054  112737  000001  001115          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 4666  022062  013777  001102  157052  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
 4667  022070  013716  001106                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 4668  022074  000002                          RTI                     ;;FIXES PS
 4669  022076  000001                  $MXCNT: 1                       ;;MAX. NUMBER OF ITERATIONS
 4670
 4671                                   .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
```

```
 4784  022450  012637  000004                MOV     (SP)+,@#ERRVEC
 4785  022454  010137  022476                MOV     R1,$LSTAD       ;;LAST ADDRESS
 4786  022460  010237  022500                MOV     R2,$LSTBK       ;;LAST BANK
 4787  022464  012603                        MOV     (SP)+,R3        ;;RESTORE R3
 4788  022466  012602                        MOV     (SP)+,R2        ;;RESTORE R2
 4789  022470  012601                        MOV     (SP)+,R1        ;;RESTORE R1
 4790  022472  012600                        MOV     (SP)+,R0        ;;RESTORE R0
 4791  022474  000207                        RTS     PC
 4792  022476  000000              $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
 4793  022500  000000              $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
 4794
 4795                                .SBTTL  TRAP DECODER
 4796
 4797                                ;;**************************************************************
 4798                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
 4799                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 4800                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 4801                                ;*GO TO THAT ROUTINE.
 4802
 4803  022502  010046              $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
 4804  022504  016600  000002              MOV     2(SP),R0        ;;GET TRAP ADDRESS
 4805  022510  005740                      TST     -(R0)           ;;BACKUP BY 2
 4806  022512  111000                      MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
 4807  022514  006300                      ASL     R0              ;;POSITION FOR INDEXING
 4808  022516  016000  022536              MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
 4809  022522  000200                      RTS     R0              ;;GO TO ROUTINE
 4810
 4811
 4812                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
 4813
 4814  022524  011646              $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
 4815  022526  016666  000004  000002      MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
 4816  022534  000002                      RTI                     ;;RESTORE THE PSW
 4817
 4818                                .SBTTL  TRAP TABLE
 4819
 4820                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 4821                                ;*BY THE "TRAP" INSTRUCTION.
 4822
 4823                                ;       ROUTINE
 4824                                ;       -------
 4825  022536  022524              $TRPAD: .WORD   $TRAP2
 4826  022540  017504                      $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
 4827  022542  017750                      $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 4828  022544  017724                      $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
 4829  022546  017764                      $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
 4830  022550  020152                      $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
 4831
 4832  022552  020774                      $GTSWR  ;;CALL=GTSWR    TRAP+6(104406)  GET SOFT-SWR SETTING
 4833
 4834  022554  020704                      $CKSWR  ;;CALL=CKSWR    TRAP+7(104407)  TEST FOR CHANGE IN SOFT-SWR
 4835  022556  021246                      $RDCHR  ;;CALL=RDCHR    TRAP+10(104410) TTY TYPEIN CHARACTER ROUTINE
 4836  022560  021336                      $RDLIN  ;;CALL=RDLIN    TRAP+11(104411) TTY TYPEIN STRING ROUTINE
 4837  022562  022100                      $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
 4838  022564  022136                      $RESREG ;;CALL=RESREG   TRAP+13(104413) RESTORE R0-R5 ROUTINE
 4839
```

```
 4840
 4841                                   ;THIS ROUTINE CLEARS THE RP11 AND DETERMINES WHICH DRIVES ARE AVAILABLE.
 4842                                   ;THE TABLE 'DRVSTA' IS LOADED TO REFLECT THE SYSTEM STATUS.
 4843
 4844  022566  104412                   RPINIT: SAVREG                          ;SAVE R0-R5
 4845  022570  013701  001326                   MOV     RPVEC,R1               ;VECTOR ADDRESS
 4846  022574  005021                           CLR     (R1)+                  ;SET INTERRUPT ADDRESS TO ZERO
 4847  022576  013711  001332                   MOV     RPPRIO,(R1)            ;RP11 PRIORITY
 4848  022602  005037  001304                   CLR     DRVTYP                 ;CLEAR DRIVE TYPE STORAGE
 4849  022606  005037  001370                   CLR     DRVSTA                 ;SET DRIVE STATUS TO OFFLINE
 4850  022612  005037  001372                   CLR     DRVSTA+2               ;SET DRIVE STATUS TO OFFLINE
 4851  022616  005037  001374                   CLR     DRVSTA+4               ;SET DRIVE STATUS TO OFFLINE
 4852  022622  005037  001376                   CLR     DRVSTA+6               ;SET DRIVE STATUS TO OFFLINE
 4853  022626  013704  001324                   MOV     RPADR,R4               ;PUT RP11 ADDRESS INTO R4
 4854  022632  012764  000001  000004           MOV     #CLEAR,RPCS(R4)         ;CLEAR THE RP11
 4855  022640  012764  000001  000004           MOV     #CLEAR,RPCS(R4)         ;CLEAR THE RP11 AGAIN
 4856  022646  012701  000007                   MOV     #7,R1                  ;'DRVSTA' TABLE INDEX
 4857  022652  004737  022666           1$:     JSR     PC,DRVINT              ;CHECK THE DRIVE'S STATUS
 4858  022656  005301                           DEC     R1                     ;DECREMENT THE TABLE INDEX
 4859  022660  100374                           BPL     1$                     ;BR IF NOT FINISHED
 4860  022662  104413                           RESREG                         ;RESTORE R0-R5
 4861  022664  000207                           RTS     PC                     ;RETURN
 4862
 4863                                   ;ROUTINE TO CHECK THE DRIVE'S STATUS.  DRIVE NUMBER MUST BE IN R1
 4864
 4865  022666  110164  000005           DRVINT: MOVB    R1,RPCS+1(R4)  ;SELECT DRIVE
 4866  022672  032764  001000  000000           BIT     #SUFU,RPDS(R4)  ;SEE IF DRIVE UNSAFE
 4867  022700  001404                           BEQ     1$             ;BR IF NOT UNSAFE
 4868  022702  112761  177777  001370           MOVB    #-1,DRVSTA(R1)  ;SET INDICATOR TO 'UNSAFE'
 4869  022710  000417                           BR      3$             ;EXIT
 4870  022712  032764  040000  000000   1$:     BIT     #SUOL,RPDS(R4)  ;IS DRIVE ONLINE ?
 4871  022720  001422                           BEQ     4$             ;BR IF NOT
 4872  022722  032764  004000  000000           BIT     #SUSI,RPDS(R4)  ;SEEK INCOMPLETE SET ?
 4873  022730  001004                           BNE     2$             ;BR IF SET
 4874  022732  032764  100000  000000           BIT     #SURDY,RPDS(R4)  ;IS DRIVE READY ?
 4875  022740  001412                           BEQ     4$             ;BR IF NOT
 4876  022742  112761  000001  001370   2$:     MOVB    #1,DRVSTA(R1)  ;SET DRIVE INDICATOR TO 'ONLINE'
 4877  022750  032764  020000  000000   3$:     BIT     #SURP03,RPDS(R4)  ;IS THE DRIVE AN RP03 ?
 4878  022756  001403                           BEQ     4$             ;BR IF NOT
 4879  022760  156137  001360  001304           BISB    ATABIT(R1),DRVTYP  ;SET RP03 INDICATOR
 4880  022766  000207                   4$:     RTS     PC             ;RETURN
 4881
 4882                                   ;DO THE REQUESTED COMMAND.  THE RP11 ADDRESS MUST BE IN R4
 4883
 4884  022770  104412                   RP11:   SAVREG                         ;SAVE R0 - R5
 4885  022772  113764  002500  000005           MOVB    DPB,RPCS+1(R4)         ;SELECT THE DRIVE
 4886  023000  005001                           CLR     R1                     ;CLEAR R1
 4887  023002  113701  002500                   MOVB    DPB,R1                 ;DRIVE ADDRESS
 4888  023006  004737  023120                   JSR     PC,CIR                 ;SEE IF DRIVE IS READY
 4889  023012  013746  002504                   MOV     $WC,-(SP)              ;COMPLEMENT THE WORD COUNT
 4890  023016  005416                           NEG     (SP)                   ;COMPLEMENT
 4891  023020  012664  000006                   MOV     (SP)+,RPWC(R4)         ;LOAD THE WORD COUNT REGISTER
 4892  023024  013764  002506  000010           MOV     $BUF,RPBA(R4)          ;BUFFER ADDRESS
 4893  023032  013764  002510  000012           MOV     $CYL,RPCA(R4)          ;CYLINDER ADDRESS
 4894  023040  013764  002512  000014           MOV     $SEC,RPDA(R4)          ;SECTOR/TRACK ADDRESS
 4895  023046  012737  024000  023116           MOV     #24000,2$                      ;SET UP FOR....
```

```
 4896  023054  005337  023116          1$:     DEC     2$                      ;...A DELAY OF
 4897                                                                          ;...AT LEAST
 4898  023060  001375                          BNE     1$                      ;...25 MILLISECONDS.
 4899  023062  113764  002500  000005        MOVB    DPB,RPCS+1(R4)  ;SELECT THE DRIVE
 4900  023070  153764  002503  000005        BISB    DPB+3,RPCS+1(R4)  ;SET 'MODE' & 'HDR' BITS
 4901  023076  013746  002502                MOV     DPB+2,-(SP)     ;COMMAND CODE
 4902  023102  153716  002501                BISB    DPB+1,(SP)      ;'MEX' BITS
 4903  023106  112664  000004                MOVB    (SP)+,RPCS(R4)  ;START THE COMMAND
 4904  023112  104413                        RESREG                  ;RESTORE R0 - R5
 4905  023114  000207                        RTS     PC              ;RETURN
 4906  023116  000000                  2$:   0
 4907
 4908                                  ;SEE IF DRIVE IS STILL ONLINE
 4909
 4910  023120                          CIR:
 4911  023120  012737  023152  001164        MOV     #2$,$ESCAPE     ;;ESCAPE TO 2$ ON ERROR
 4912  023126  004737  022666                JSR     PC,DRVINT       ;CHECK ON THE DRIVE'S STATUS
 4913  023132  004737  023240                JSR     PC,SAVRP        ;STORE RP11 REGISTERS IN CASE OF ERROR
 4914  023136  105761  001370                TSTB    DRVSTA(R1)      ;IS THE DRIVE STILL ONLINE ?
 4915  023142  003014                        BGT     3$              ;BR IF IT IS
 4916  023144  002401                        BLT     1$              ;BR IF DRIVE IS UNSAFE
 4917  023146  104076                        ERROR   76              ;DRIVE HAS GONE OFFLINE
 4918  023150  104077                  1$:   ERROR   77              ;DRIVE HAS BECOME UNSAFE
 4919  023152  032777  001000  155760  2$:   BIT     #SW09,@SWR      ;LOOP ON THE ERROR ?
 4920  023160  001357                        BNE     CIR             ;BR IF LOOP
 4921  023162  156137  001360  001310        BISB    ATABIT(R1),DRVBAD  ;SETUP TO DEASSIGN THE DRIVE
 4922  023170  000137  016616                JMP     @#$EOP          ;GO TO THE END OF PASS ROUTINE
 4923  023174  005037  001164          3$:   CLR     $ESCAPE         ;CLEAR THE 'ESCAPE' FLAG
 4924  023200  000207                        RTS     PC              ;RETURN
 4925
 4926                                  ;THIS ROUTINE WILL PROVIDE A STALL IN MILLISECONDS FOR A SPECIFIC
 4927                                  ;AMOUNT OF TIME.  STALL VALUE IS STORED IN 'STALLT'.
 4928                                  ;       JSR     PC,STALL
 4929
 4930  023202  013746  001220          STALL:  MOV     STALLT,-(SP)    ;PICKUP THE STALL TIME
 4931  023206  005046                          CLR     -(SP)           ;CLEAR TEMP. LOCATION
 4932  023210  162766  000001  000002  1$:     SUB     #1,2(SP)        ;MORE STALL REQUIRED?
 4933  023216  103406                          BLO     3$              ;NO--BRANCH
 4934  023220  012716  000454                  MOV     #300.,(SP)      ;STALL FOR ABOUT 1 MILLISECOND
 4935  023224  005700                  2$:     TST     R0              ;NOP TO KILL TIME
 4936  023226  005316                          DEC     (SP)            ;COUNT
 4937  023230  001375                          BNE     2$              ;LOOP IF MORE COUNTS NEEDED
 4938  023232  000766                          BR      1$
 4939  023234  022626                  3$:     CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
 4940  023236  000207                          RTS     PC              ;EXIT
 4941
 4942                                  ;THIS ROUTINE STORES THE RP11E REGISTERS FOR USE BY THE PROGRAM.
 4943                                  ;THE PROGRAM DOES NOT USE THE 'LIVE' REGISTERS FOR ERROR DETERMINATION.
 4944                                  ;CALL
 4945                                  ;       JSR     PC,SAVRP
 4946                                  ;       RETURN
 4947
 4948  023240  010446                  SAVRP:  MOV     R4,-(SP)        ;SAVE R4
 4949  023242  013704  001324                  MOV     RPADR,R4        ;RP11 ADDRESS
 4950  023246  016437  000000  001334          MOV     RPDS(R4),$RPDS  ;STORE RPDS
 4951  023254  016437  000002  001336          MOV     RPER(R4),$RPER  ;STORE RPER
```

```
 4952  023262  016437  000004  001340          MOV     RPCS(R4),$RPCS  ;STORE RPCS
 4953  023270  016437  000006  001342          MOV     RPWC(R4),$RPWC  ;STORE RPWC
 4954  023276  016437  000010  001344          MOV     RPBA(R4),$RPBA  ;STORE RPBA
 4955  023304  016437  000012  001346          MOV     RPCA(R4),$RPCA  ;STORE RPCA
 4956  023312  016437  000014  001350          MOV     RPDA(R4),$RPDA  ;STORE RPDA
 4957  023320  016437  000016  001352          MOV     RPM1(R4),$RPM1  ;STORE RPM1
 4958  023326  016437  000024  001354          MOV     SUCA(R4),$SUCA  ;STORE SUCA
 4959  023334  016437  000026  001356          MOV     SILO(R4),$SILO  ;STORE SILO
 4960  023342  012604                          MOV     (SP)+,R4        ;RESTORE R4
 4961  023344  000207                          RTS     PC              ;RETURN
 4962
 4963                                  ;ROUTINE TO MOVE THE LOADER FROM HIGH MEMORY TO JUST ABOVE THE PROGRAM
 4964
 4965  023346  105737  000041        MOVLDR: TSTB    41              ;WHO LOADED THE PROGRAM
 4966  023352  001005                          BNE     1$              ;BR IF NOT 'ABS'
 4967  023354  012702  000144                  MOV     #100.,R2        ;'ABS' LOADER SIZE
 4968  023360  012701  044166                  MOV     #BUFFER+200.,R1 ;RELOCATE TO HERE
 4969  023364  000404                          BR      2$              ;FINISH
 4970  023366  012702  002734        1$:     MOV     #1500.,R2       ;'XXDP' SIZE
 4971  023372  012701  051546                  MOV     #BUFFER+3000.,R1  ;RELOCATE TO HERE (UPPER BOUNDARY)
 4972  023376  012703  157776        2$:     MOV     #157776,R3      ;TRY THIS UPPER MEMORY ADDRESS
 4973  023402  005737  022232                  TST     $KT11           ;MEMORY MANAGEMENT ?
 4974  023406  100402                          BMI     3$              ;BR IF YES
 4975  023410  013703  022476                  MOV     $LSTAD,R3       ;USE ACTUAL HIGH ADDRESS
 4976  023414  005046                  3$:     CLR     -(SP)           ;CLEAR THE STACK
 4977  023416  016616  000002                  MOV     2(SP),(SP)      ;MOVE THE RETURN ADDRESS DOWN THE STACK
 4978  023422  010166  000002                  MOV     R1,2(SP)        ;BEGINNING ADDRESS OF BUFFER
 4979  023426  005737  001322                  TST     LDRFLG          ;LOADER ALREADY RELOCATED ?
 4980  023432  001006                          BNE     5$              ;BR IF YES
 4981  023434  014341                  4$:     MOV     -(R3),-(R1)     ;MOVE A WORD
 4982  023436  005302                          DEC     R2              ;COUNT IT
 4983  023440  001375                          BNE     4$              ;BR IF MORE TO GO
 4984  023442  012737  177777  001322          MOV     #-1,LDRFLG      ;SET THE LOADER MOVED FLAG
 4985  023450  000207                  5$:     RTS     PC              ;RETURN
 4986
 4987                                  ;ROUTINE TO RELOCATE THE LOADER TO UPPER MEMORY
 4988
 4989  023452  104412                  RESLDR: SAVREG                  ;SAVE R0 - R5
 4990  023454  005737  001322                  TST     LDRFLG          ;LOADER MOVED ?
 4991  023460  001430                          BEQ     5$              ;BR IF NOT
 4992  023462  105737  000041                  TSTB    41              ;'ABS' LOADER
 4993  023466  001005                          BNE     1$              ;BR IF NOT
 4994  023470  012702  000144                  MOV     #100.,R2        ;'ABS' LOADER SIZE
 4995  023474  012701  044166                  MOV     #BUFFER+200.,R1 ;RELOCATE FROM HERE
 4996  023500  000404                          BR      3$              ;FINISH
 4997  023502  012702  002734        1$:     MOV     #1500.,R2       ;'XXDP' SIZE
 4998  023506  012701  051546                  MOV     #BUFFER+3000.,R1  ;RELOCATE FROM HERE (UPPER BOUNDARY)
 4999  023512  012703  157776        3$:     MOV     #157776,R3      ;TRY THIS UPPER MEMORY ADDRESS
 5000  023516  005737  022232                  TST     $KT11           ;MEMORY MANAGEMENT ?
 5001  023522  100402                          BMI     4$              ;BR IF YES
 5002  023524  013703  022476                  MOV     $LSTAD,R3       ;USE ACTUAL HIGH ADDRESS
 5003  023530  014143                  4$:     MOV     -(R1),-(R3)     ;MOVE A WORD
 5004  023532  005302                          DEC     R2              ;COUNT IT
 5005  023534  001375                          BNE     4$              ;BR IF MORE TO GO
 5006  023536  005037  001322                  CLR     LDRFLG          ;CLEAR THE LOADER MOVED FLAG
 5007  023542  104413                  5$:     RESREG                  ;RESTORE R0 - R5
```

```
5008  023544  000207                          RTS     PC              ;RETURN
5009
5010                                  ;ROUTINE TO WAIT FOR DRIVE READY
5011
5012  023546  032764  100000  000000  DRVRDY: BIT     #SURDY,RPDS(R4)  ;DRIVE READY ?
5013  023554  001042                          BNE     6$              ;BR IF READY
5014  023556  013746  001212                  MOV     TIMOUT,-(SP)    ;WAIT TIME: APPROX 1 SEC
5015  023562  005046                          CLR     -(SP)           ;CLEAR WORKING LOCATION
5016  023564  162766  000001  000002  1$:     SUB     #1,2(SP)        ;ANY TIME LEFT IN WAIT LOOP ?
5017  023572  001411                          BEQ     3$              ;BR IF NOT
5018  023574  012716  000144                  MOV     #100.,(SP)      ;VALUE FOR APPROX 1 MS
5019  023600  032764  100000  000000  2$:     BIT     #SURDY,RPDS(R4) ;DRIVE READY ?
5020  023606  001023                          BNE     5$              ;BR IF READY
5021  023610  005316                          DEC     (SP)            ;COUNT SOME TIME
5022  023612  001372                          BNE     2$              ;BR IF NOT FINISHED
5023  023614  000763                          BR      1$              ;COUNT MAJOR TIME AGAIN
5024  023616  004737  023240          3$:     JSR     PC,SAVRP        ;STORE RP11 REGISTERS FOR ERROR MESSAGE
5025  023622  012737  023632  001164          MOV     #4$,$ESCAPE     ;;ESCAPE TO 4$ ON ERROR
5026  023630  104100                          ERROR   100             ;DRIVE TIMED OUT
5027  023632  005037  001164          4$:     CLR     $ESCAPE         ;CLEAR THE 'ESCAPE' FLAG
5028  023636  062706  000004                  ADD     #4,SP           ;CORRECT  THE STACK POINTER
5029  023642  032777  001000  155270          BIT     #SW09,@SWR      ;LOOP ON THE ERROR ?
5030  023650  001336                          BNE     DRVRDY          ;LOOP IF SET
5031  023652  000177  155322                  JMP     @BYPASS         ;GO THE 'BYPASS' ADDRESS
5032  023656  062706  000004          5$:     ADD     #4,SP           ;CORRECT THE STACK POINTER
5033  023662  000207                  6$:     RTS     PC              ;RETURN
5034
5035
5036                                  ;ROUTINE TO WAIT FOR CONTROLLER READY
5037
5038  023664  032764  000200  000004  CONRDY: BIT     #RDY,RPCS(R4)   ;CONTROLLER READY ?
5039  023672  001042                          BNE     6$              ;BR IF READY
5040  023674  013746  001212                  MOV     TIMOUT,-(SP)    ;WAIT TIME: APPROX 1 SEC
5041  023700  005046                          CLR     -(SP)           ;CLEAR WORKING LOCATION
5042  023702  162766  000001  000002  1$:     SUB     #1,2(SP)        ;ANY TIME LEFT IN WAIT LOOP ?
5043  023710  001411                          BEQ     3$              ;BR IF NOT
5044  023712  012716  000144                  MOV     #100.,(SP)      ;VALUE FOR APPROX 1 MS
5045  023716  032764  000200  000004  2$:     BIT     #RDY,RPCS(R4)   ;CONTROLLER READY ?
5046  023724  001023                          BNE     5$              ;BR IF READY
5047  023726  005316                          DEC     (SP)            ;COUNT SOME TIME
5048  023730  001372                          BNE     2$              ;BR IF NOT FINISHED
5049  023732  000763                          BR      1$              ;COUNT MAJOR TIME AGAIN
5050  023734  004737  023240          3$:     JSR     PC,SAVRP        ;STORE RP11 REGISTERS FOR ERROR MESSAGE
5051  023740  012737  023750  001164          MOV     #4$,$ESCAPE     ;;ESCAPE TO 4$ ON ERROR
5052  023746  104101                          ERROR   101             ;CONTROLLER TIMED OUT
5053  023750  005037  001164          4$:     CLR     $ESCAPE         ;CLEAR THE 'ESCAPE' FLAG
5054  023754  062706  000004                  ADD     #4,SP           ;CORRECT THE STACK POINTER
5055  023760  032777  001000  155152          BIT     #SW09,@SWR      ;LOOP ON THE ERROR ?
5056  023766  001336                          BNE     CONRDY          ;BR IF LOOPING
5057  023770  000177  155204                  JMP     @BYPASS         ;'BYPASS' THE TEST
5058  023774  062706  000004          5$:     ADD     #4,SP           ;CORRECT THE STACK POINTER
5059  024000  000207                  6$:     RTS     PC              ;RETURN
5060
5061                                  ;ROUTINE TO DETERMINE THE NUMBER OF SECTORS ON THE DRIVE BEING TESTED
5062
5063  024002  010046                  SECNBR: MOV     R0,-(SP)        ;SAVE R0
```

```
5064  024004  012737  117230  001244          MOV     #40600.,PAKSIZ  ;START WITH THE NUMBER OF SECTORS ON
5065  024012  005037  001246                  CLR     PAKSIZ+2        ;AN RP02
5066  024016  005000                          CLR     R0              ;USE R0 AS AN INDEX
5067  024020  113700  002500                  MOVB    DPB,R0          ;GET THE DRIVE NUMBER
5068  024024  136037  001360  001304          BITB    ATABIT(R0),DRVTYP  ;IS THE DRIVE AN RP03 ?
5069  024032  001404                          BEQ     1$              ;BR IF NOT
5070  024034  006337  001244                  ASL     PAKSIZ          ;DOUBLE THE SECTOR COUNT
5071  024040  006137  001246                  ROL     PAKSIZ+2        ;GET THE CARRY
5072  024044  012600                  1$:     MOV     (SP)+,R0        ;RESTORE R0
5073  024046  000207                          RTS     PC              ;RETURN
5074
5075                                  ;ROUTINE TO SELECT A PATTERN
5076
5077  024050  010046                  PATSEL: MOV     R0,-(SP)        ;SAVE R0
5078  024052  005237  001204          1$:     INC     PATNUM          ;INCREMENT THE PATTERN NUMBER
5079  024056  022737  000020  001204          CMP     #16.,PATNUM     ;NUMBER AT MAXIMUM
5080  024064  101002                          BHI     2$              ;BR IF NOT
5081  024066  005037  001204                  CLR     PATNUM          ;START AT ZERO AGAIN
5082  024072  013700  001204          2$:     MOV     PATNUM,R0       ;SEE IF THE PATTERN IS IN USE
5083  024076  006300                          ASL     R0              ;CONVERT INTO A TABLE INDEX
5084  024100  036037  001400  001300          BIT     BITS(R0),PATTRN ;INUSE INDICATOR SET FOR THIS PATTERN ?
5085  024106  001761                          BEQ     1$              ;BR IF NOT
5086  024110  012600                          MOV     (SP)+,R0        ;RESTORE R0
5087  024112  000207                          RTS     PC              ;RETURN
5088
5089                                  ;ROUTINE TO INCREMENT THE DISK ADDRESS
5090                                  ;       RETURN + 2 WHEN THE END OF THE PACK IS REACHED
5091
5092  024114  005737  001260          INCADR: TST     OPRSEL          ;USING AN OPERATOR SPECIFIED ADDRESS ?
5093  024120  001066                          BNE     5$              ;BR IF YES
5094  024122  005737  001246                  TST     PAKSIZ+2        ;SECTORS LEFT ?
5095  024126  003017                          BGT     1$              ;BR IF AN UPPER COUNT
5096  024130  005737  001244                  TST     PAKSIZ          ;LOWER COUNT AT ZERO ?
5097  024134  001460                          BEQ     5$              ;BR IF IT IS
5098  024136  023737  001244  001254          CMP     PAKSIZ,BLKS14   ;ENOUGH SECTORS LEFT FOR A FULL TRANSFER ?
5099  024144  103010                          BHIS    1$              ;BR IF THERE ARE
5100  024146  013737  001244  002504          MOV     PAKSIZ,$WC      ;USE THE RESIDUE COUNT FOR THIS TRANSFER
5101  024154  000337  002504                  SWAB    $WC             ;CONVERT TO A WORD COUNT
5102  024160  005037  001244                  CLR     PAKSIZ          ;FORCE COUNT TO ZERO
5103  024164  000410                          BR      2$              ;GO AND INCREMENT THE ADDRESS
5104  024166  013737  001256  002504  1$:     MOV     WC14,$WC        ;USE THE STANDARD WORD COUNT
5105  024174  163737  001254  001244          SUB     BLKS14,PAKSIZ   ;DECREMENT THE TOTAL SECTORS AVAILABLE
5106  024202  005637  001246                  SBC     PAKSIZ+2        ;SUBTRACT ANY CARRY
5107  024206  005046                  2$:     CLR     -(SP)           ;CLEAR THE STACK
5108  024210  113716  002512                  MOVB    $SEC,(SP)       ;GET CURRENT SECTOR
5109  024214  063716  001250                  ADD     INCSEC,(SP)     ;ADD THE SECTOR INCREMENT
5110  024220  121627  000012                  CMPB    (SP),#10.       ;EXCEED THE MAXIMUM ?
5111  024224  002404                          BLT     3$              ;BR IF NOT
5112  024226  162716  000012                  SUB     #10.,(SP)       ;CORRECT THE RESIDUE
5113  024232  105237  002513                  INCB    $TRK            ;INCREMENT THE TRACK
5114  024236  111637  002512          3$:     MOVB    (SP),$SEC       ;NEW SECTOR ADDRESS
5115  024242  113716  002513                  MOVB    $TRK,(SP)       ;GET CURRENT TRACK VALUE
5116  024246  063716  001252                  ADD     INCTRK,(SP)     ;ADD THE TRACK INCREMENT
5117  024252  121627  000024                  CMPB    (SP),#20.       ;EXCEED THE MAXIMUM ?
5118  024256  002404                          BLT     4$              ;BR IF NOT
5119  024260  162716  000024                  SUB     #20.,(SP)       ;CORRECT THE SIZE
```

```
 5120  024264  005237  002510                 INC     $CYL            ;INCREMENT THE CYLINDER ADDRESS
 5121  024270  112637  002513         4$:     MOVB    (SP)+,$TRK      ;NEW TRACK ADDRESS
 5122  024274  000402                        BR      6$              ;EXIT
 5123  024276  062716  000002         5$:     ADD     #2,(SP)         ;INCREMENT RETURN - AT END
 5124  024302  000207                 6$:     RTS     PC              ;RETURN
 5125
 5126                                 ;ROUTINE TO SETUP A BUFFER WITH THE SEQUENTIAL HEADERS FOR CYLINDER 0
 5127                                 ;       TRACK 0
 5128
 5129  024304  104412                 LODSEC: SAVREG                  ;SAVE R0 - R5
 5130  024306  013701  002506                 MOV     $BUF,R1         ;BUFFER ADDRESS
 5131  024312  005003                         CLR     R3              ;SECTOR COUNTER
 5132  024314  005021                 1$:     CLR     (R1)+           ;UPPER WORD OF HEADER
 5133  024316  005021                         CLR     (R1)+           ;CYLINDER/TRACK WORD OF HEADER
 5134  024320  010321                         MOV     R3,(R1)+        ;SECTOR ADDRESS
 5135  024322  005203                         INC     R3              ;INCREMENT THE SECTOR ADDRESS
 5136  024324  022703  000012                 CMP     #10.,R3         ;FINISHED ?
 5137  024330  001371                         BNE     1$              ;BR IF NOT
 5138  024332  104413                         RESREG                  ;RESTORE R0 - R5
 5139  024334  000207                         RTS     PC              ;RETURN
 5140
 5141                                 ;FILL THE BUFFER WITH THE SECTOR ADDRESS (CONTAINED IN '$SEC')
 5142
 5143  024336  104412                 FILSEC: SAVREG                  ;STORE R0-R5
 5144  024340  005003                         CLR     R3              ;CLEAR THE INDEX
 5145  024342  113703  002512                 MOVB    $SEC,R3         ;SECTOR ADDRESS
 5146  024346  001002                         BNE     1$              ;BR IF NOT SECTOR ZERO
 5147  024350  012703  000012                 MOV     #10.,R3         ;USE 10. AS VALUE FOR SECTOR ZERO
 5148  024354  013702  002504         1$:     MOV     $WC,R2          ;WORD COUNT
 5149  024360  013701  002506                 MOV     $BUF,R1         ;BUFFER START ADDRESS
 5150  024364  010321                 2$:     MOV     R3,(R1)+        ;MOVE DATA INTO THE BUFFER
 5151  024366  005302                         DEC     R2              ;DECREMENT THE COUNTER
 5152  024370  001375                         BNE     2$              ;BR IF MORE TO DO
 5153  024372  104413                         RESREG                  ;RESTORE R0-R5
 5154  024374  000207                         RTS     PC              ;RETURN
 5155
 5156                                 ;ROUTINE TO FILL THE BUFFER WITH THE TRACK ADDRESS (CONTAINED IN '$TRK')
 5157
 5158  024376  104412                 FILTRK: SAVREG                  ;STORE R0-R5
 5159  024400  012701  043657                 MOV     #BUFFER+1,R1    ;BUFFER ADDRESS
 5160  024404  013702  002504                 MOV     $WC,R2          ;WORD COUNT
 5161  024410  005003                         CLR     R3              ;CLEAR THE DATA STORAGE
 5162  024412  113711  002513         1$:     MOVB    $TRK,(R1)       ;MOVE TRACK ADDRESS INTO THE BUFFER
 5163  024416  062701  000002                 ADD     #2,R1           ;INCREMENT THE BUFFER POINTER
 5164  024422  005302                         DEC     R2              ;DECREMENT THE COUNTER
 5165  024424  001372                         BNE     1$              ;BR IF MORE TO DO
 5166  024426  104413                         RESREG                  ;RESTORE R0-R5
 5167  024430  000207                         RTS     PC              ;RETURN
 5168
 5169                                 ;THIS ROUTINE WILL MOVE THE 16 WORDS OF THE
 5170                                 ;DESIRED PATTERN INTO THE DATA BUFFER.
 5171
 5172  024432  104412                 SETBUF: SAVREG                  ;SAVE R0 - R5
 5173  024434  013700  001204                 MOV     PATNUM,R0       ;GET THE PATTERN NUMBER
 5174  024440  006300                         ASL     R0              ;CONVERT NUMBER TO INDEX
 5175  024442  013701  002506                 MOV     $BUF,R1         ;FIRST ADDRESS
```

```
 5176  024446  013702  002504              MOV     $WC,R2          ;WORD COUNT
 5177  024452  016003  001440      1$:     MOV     PAT.PT(R0),R3   ;PICKUP PATTERN POINTER
 5178  024456  012704  000020              MOV     #16.,R4         ;PATTERN COUNTER
 5179  024462  012321              2$:     MOV     (R3)+,(R1)+     ;MOVE WORD 'N' INTO DATA BUFFER
 5180  024464  005302                      DEC     R2              ;DECREMENT THE COUNTER
 5181  024466  001403                      BEQ     3$              ;BR IF DONE
 5182  024470  005304                      DEC     R4              ;DECREMENT THE PATTERN COUNTER
 5183  024472  001767                      BEQ     1$              ;START PATTERN AGAIN
 5184  024474  000772                      BR      2$              ;CONTINUE
 5185  024476  104413              3$:     RESREG                  ;RESTORE R0 - R5
 5186  024500  000207                      RTS     PC              ;RETURN
 5187
 5188                          ;THIS ROUTINE LOADS THE PHYSICAL TRANSFER ADDRESS INTO '$BUF'
 5189
 5190  024502  013737  001224  002506  GETBUF: MOV     ACTMEM,$BUF     ;LOWER 16 BITS OF PHYSICAL ADDRESS
 5191  024510  013746  001226              MOV     ACTMEM+2,-(SP)  ;UPPER ADDRESS BITS ON THE STACK
 5192  024514  042716  177774              BIC     # C3,(SP)       ;LEAVE ONLY THE LOWER 2 BITS
 5193  024520  006316                      ASL     (SP)            ;ALIGN UPPER ADDRESS BITS
 5194  024522  006316                      ASL     (SP)            ;ALIGN UPPER ADDRESS BITS
 5195  024524  006316                      ASL     (SP)            ;ALIGN UPPER ADDRESS BITS
 5196  024526  006316                      ASL     (SP)            ;ALIGN UPPER ADDRESS BITS
 5197  024530  112637  002501              MOVB    (SP)+,DPB+1     ;LOAD 'MEX' BITS
 5198  024534  013746  002504              MOV     $WC,-(SP)       ;PUT WORD COUNT ON THE STACK
 5199  024540  006316                      ASL     (SP)            ;DOUBLE THE WORD COUNT
 5200  024542  062637  001224              ADD     (SP)+,ACTMEM    ;FORM NEXT STARTING ADDRESS
 5201  024546  005537  001226              ADC     ACTMEM+2        ;ADD ANY CARRY
 5202  024552  023737  001226  001236      CMP     ACTMEM+2,HIMEM+2  ;EXCEEDING HIGHEST BANK ADDRESS
 5203  024560  101005                      BHI     1$              ;BR IF HIGHER
 5204  024562  103413                      BLO     2$              ;BR IF LOWER
 5205  024564  023737  001224  001234      CMP     ACTMEM,HIMEM    ;CHECK LOWER 16 BITS
 5206  024572  103407                      BLO     2$              ;BR IF LOWER
 5207  024574  013737  001240  001224  1$: MOV     LOMEM,ACTMEM    ;RESET ADDRESS
 5208  024602  013737  001242  001226      MOV     LOMEM+2,ACTMEM+2  ;UPPER ADDRESS BITS
 5209  024610  000734                      BR      GETBUF          ;TRY AGAIN
 5210  024612  000207              2$:     RTS     PC              ;RETURN
 5211
 5212                          ;THIS ROUTINE COMPARES A 16 WORD DATA PATTERN AGAINST THE
```

```
 5213                                   ;DATA BUFFER.  THE PATTERN NUMBER IS IN 'PATNUM'
```

```
 5214
 5215  024614  104412                  DATCMP: SAVREG                  ;SAVE R0-R5
 5216  024616  005737  001222                  TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
 5217  024622  001510                          BEQ     2$              ;BR IF NOT
```

```
 5218  024624  005037  172340            CLR     KIPAR0          ;LOAD ADDRESS REGISTER 0
 5219  024630  012737  000200  172342    MOV     #200,KIPAR1     ;LOAD ADDRESS REGISTER 1
 5220  024636  012737  000400  172344    MOV     #400,KIPAR2     ;LOAD ADDRESS REGISTER 2
 5221  024644  012737  177600  172356    MOV     #177600,KIPAR7  ;ACCESS TO I/O PAGE
 5222  024652  005046                    CLR     -(SP)           ;MAKE ROOM ON THE STACK
 5223  024654  005046                    CLR     -(SP)           ;MAKE ROOM ON THE STACK
 5224  024656  013766  002506  000002    MOV     $BUF,2(SP)      ;LOWER 16 BITS OF STARTING ADDRESS
 5225  024664  113716  002501            MOVB    DPB+1,(SP)      ;UPPER ADDRESS BITS
 5226  024670  006216                    ASR     (SP)            ;RIGHT JUSTIFY THE ADDRESS BITS
 5227  024672  006216                    ASR     (SP)            ;RIGHT JUSTIFY THE ADDRESS BITS
 5228  024674  006216                    ASR     (SP)            ;RIGHT JUSTIFY THE ADDRESS BITS
 5229  024676  006216                    ASR     (SP)            ;RIGHT JUSTIFY THE ADDRESS BITS
 5230  024700  012703  000006            MOV     #6,R3           ;LOAD COUNTER
 5231  024704  006216            1$:     ASR     (SP)            ;CONVERT ADDRESS TO BLOCK ADDRESS
 5232  024706  006066  000002            ROR     2(SP)           ;LOWER 16 BITS
 5233  024712  005303                    DEC     R3              ;DECREMENT THE COUNTER
 5234  024714  001373                    BNE     1$              ;CONTINUE UNTIL FINISHED
 5235  024716  042766  000177  000002    BIC     #177,2(SP)      ;CLEAR LOWER BLOCK ADDRESS BITS
 5236  024724  016637  000002  172346    MOV     2(SP),KIPAR3    ;LOAD ADDRESS REGISTER 3
 5237  024732  062766  000200  000002    ADD     #200,2(SP)      ;INCREMENT BLOCK ADDRESS BY 4K
 5238  024740  016637  000002  172350    MOV     2(SP),KIPAR4    ;LOAD ADDRESS REGISTER # 4
 5239  024746  012737  077406  172300    MOV     #77406,KIPDR0   ;LOAD DESCRIPTOR REGISTER # 0
 5240  024754  012737  077406  172302    MOV     #77406,KIPDR1   ;LOAD DESCRIPTOR REGISTER # 1
 5241  024762  012737  077406  172304    MOV     #77406,KIPDR2   ;LOAD DESCRIPTOR REGISTER # 2
 5242  024770  012737  077406  172306    MOV     #77406,KIPDR3   ;LOAD DESCRIPTOR REGISTER # 3
 5243  024776  012737  077406  172310    MOV     #77406,KIPDR4   ;LOAD DESCRIPTOR REGISTER # 4
 5244  025004  012737  077406  172316    MOV     #77406,KIPDR7   ;LOAD DESCRIPTOR REGISTER # 7
 5245  025012  023727  002504  010000    CMP     $WC,#4096.      ;WAS TRANSFER GREATER THAN 4K ?
 5246  025020  101411                    BLOS    2$              ;BR IF NOT
 5247  025022  062766  000200  000002    ADD     #200,2(SP)      ;INCREMENT THE ADDRESS BY 4K
 5248  025030  016637  000002  172352    MOV     2(SP),KIPAR5    ;LOAD ADDRESS REGISTER # 5
 5249  025036  012737  077406  172312    MOV     #77406,KIPDR5   ;LOAD DESCRIPTOR REGISTER # 5
 5250  025044  013701  002506        2$: MOV     $BUF,R1         ;LOAD STARTING ADDRESS
 5251  025050  005737  001222            TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
 5252  025054  001425                    BEQ     CMPAR           ;BR IF NOT
 5253  025056  042701  140000            BIC     #140000,R1      ;CHANGE PHYSICAL BASE TO VIRTUAL BASE
 5254  025062  052701  060000            BIS     #60000,R1       ;START AT VIRTUAL 12K
 5255  025066  062706  000004            ADD     #4,SP           ;CORRECT THE STACK POINTER
 5256  025072  012737  025110  000004    MOV     #3$,ERRVEC      ;CHANGE THE ERROR VECTOR
 5257  025100  012737  000020  172516    MOV     #20,@#SR3       ;ENABLE 22 BIT ADDRESSING MODE
 5258  025106  000402                    BR      4$              ;BR IF AN 11/70
 5259  025110  062706  000004        3$: ADD     #4,SP           ;CORRECT THE STACK POINTER
 5260  025114  012737  000006  000004  4$: MOV     #ERRVEC+2,ERRVEC  ;RESTORE THE ERROR VECTOR
 5261  025122  012737  000001  177572    MOV     #1,@#SR0        ;ENABLE MEMORY MANAGEMENT
 5262  025130  013737  001342  025420  CMPAR: MOV     $RPWC,CMCNT     ;WORD COUNT TO WORKING LOCATION
 5263  025136  063737  002504  025420        ADD     $WC,CMCNT       ;CALCULATE ACTUAL WORDS TRANSFERED
 5264  025144  013737  002510  025422        MOV     $CYL,CMCYL      ;CYLINDER ADDRESS
 5265  025152  113737  002513  025426        MOVB    $TRK,CMTRK      ;TRACK ADDRESS
 5266  025160  113737  002512  025424        MOVB    $SEC,CMSEC      ;SECTOR ADDRESS
 5267  025166  013737  001320  025416  CMSTR: MOV     CMP.CT,LIMIT    ;DISPLAY LIMIT
 5268  025174  005237  025416                INC     LIMIT           ;CONVERT PARAMETER INTO LIMIT VALUE
 5269  025200  010137  025430                MOV     R1,CMBUF        ;STARTING ADDRESS OF SECTOR BUFFER
 5270  025204  005037  025412                CLR     FRSTER          ;CLEAR 'FIRST ERROR' INDICATOR
 5271  025210  005037  025414                CLR     ERCTR           ;CLEAR ERROR COUNTER
 5272  025214  023727  025420  000400        CMP     CMCNT,#256.     ;IS BUFFER SIZE GREATER THAN ONE SECTOR ?
 5273  025222  101003                        BHI     1$              ;BR IF IT IS
```

```
5274  025224  013702  025420              MOV     CMCNT,R2        ;LESS THAN, USE REMAINING BUFFER
5275  025230  000402                      BR      2$              ;
5276  025232  012702  000400      1$:     MOV     #256.,R2        ;COMPARE SECTOR
5277  025236  160237  025420      2$:     SUB     R2,CMCNT        ;DECREMENT WORD COUNT
5278  025242  013704  001204              MOV     PATNUM,R4       ;PATTERN NUMBER
5279  025246  006304                      ASL     R4              ;GENERATE INTO AN INDEX
5280  025250  012703  000020      CMDAT:  MOV     #16.,R3         ;R3 IS PATTERN POSITION COUNTER
5281  025254  016405  001440              MOV     PAT.PT(R4),R5   ;PATTERN ADDRESS
5282  025260  022125              1$:     CMP     (R1)+,(R5)+     ;COMPARE BUFFER WITH PATTERN
5283  025262  001406                      BEQ     2$              ;BR IF EQUAL
5284  025264  032777  000010  153646      BIT     #SW03,@SWR      ;SWITCH 3 SET ?
5285  025272  001002                      BNE     2$              ;BR IF NOT
5286  025274  004737  025432              JSR     PC,CMPRT        ;TYPE THE ERROR
5287  025300  005302              2$:     DEC     R2              ;DECREMENT SIZE COUNT
5288  025302  001403                      BEQ     3$              ;BR WHEN AT END
5289  025304  005303                      DEC     R3              ;DECREMENT PATT POS COUNT
5290  025306  001364                      BNE     1$              ;BR IF NOT AT END OF PATT
5291  025310  000757                      BR      CMDAT           ;RESTART THE PATTERN
5292  025312  004737  025630      3$:     JSR     PC,ENDCMP       ;PRINT LAST LINE (IF ERRORS)
5293  025316  005737  025420              TST     CMCNT           ;AT END OF BUFFER
5294  025322  003424                      BLE     CMPRX           ;BR IF AT END
5295  025324  005237  025424              INC     CMSEC           ;INCREMENT SECTOR
5296  025330  023727  025424  000012      CMP     CMSEC,#10.      ;SECTOR GREATER THAN MAX ?
5297  025336  103713                      BLO     CMSTR           ;BR IF NOT GREATER THAN MAX
5298  025340  005037  025424              CLR     CMSEC           ;CLEAR SECTOR ADDRESS
5299  025344  005237  025426              INC     CMTRK           ;INCREMENT TRACK
5300  025350  023727  025426  000024      CMP     CMTRK,#20.      ;TRACK GREATER THAN MAX ?
5301  025356  103703                      BLO     CMSTR           ;BR IF NOT GREATER
5302  025360  005037  025426              CLR     CMTRK           ;RESET TRACK ADDRESS
5303  025364  005237  025422              INC     CMCYL           ;INCREMENT CYLINDER ADDRESS
5304  025370  000137  025166              JMP     CMSTR           ;CONTINUE WITH COMPARE
5305  025374  104413              CMPRX:  RESREG                  ;RESTORE R0 - R5
5306  025376  005737  001222              TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
5307  025402  001402                      BEQ     1$              ;BR IF NOT
5308  025404  005037  177572              CLR     @#SR0           ;TURN OFF MEMORY MANAGEMENT
5309  025410  000207              1$:     RTS     PC
5310
5311  025412  000000              FRSTER: .WORD   0               ;FIRST ERROR INDICATOR
5312  025414  000000              ERCTR:  .WORD   0               ;NUMBER OF ERRORS
5313  025416  000000              LIMIT:  .WORD   0               ;DISPLAY LIMIT
5314  025420  000000              CMCNT:  .WORD   0               ;WORD COUNT
5315  025422  000000              CMCYL:  .WORD   0               ;CYLINDER ADDRESS
5316  025424  000000              CMSEC:  .WORD   0               ;SECTOR ADDRESS
5317  025426  000000              CMTRK:  .WORD   0               ;TRACK ADDRESS
5318  025430  000000              CMBUF:  .WORD   0               ;BEGINNING ADDRESS OF SECTOR
5319
5320                              ;TYPE DATA COMPARE ERRORS
5321
5322  025432  005737  025412      CMPRT:  TST     FRSTER          ;FIRST ERROR?
5323  025436  001037                      BNE     CMPRT1          ;BR IF NOT
5324  025440  005737  001222              TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
5325  025444  001011                      BNE     1$              ;BR IF ACTIVE
5326  025446  012737  025446  001116      MOV     #.,$ERRPC       ;PC FOR ERROR MESSAGE
5327  025454  112737  000102  001114      MOVB    #102,$ITEMB     ;COMPARSION ERROR MESSAGE NUMBER
5328  025462  004737  017254              JSR     PC,TYPERR       ;REPORT THE ERROR
5329  025466  000412                      BR      2$              ;CONTINUE
```

```
 5330  025470  005037  177572          1$:     CLR     @#SR0           ;TURN OFF MEMORY MANAGEMENT
 5331  025474  012737  025474  001116          MOV     #.,$ERRPC       ;PC FOR ERROR MESSAGE
 5332  025502  112737  000103  001114          MOVB    #103,$ITEMB     ;COMPARSION ERROR MESSAGE WITH MM REGISTERS
 5333  025510  004737  017254                  JSR     PC,TYPERR       ;REPORT THE COMPARSION ERROR
 5334  025514  012737  177777  025412  2$:     MOV     #-1,FRSTER      ;SET FIRST ERROR INDICATOR
 5335  025522  005737  001222                  TST     MMACTV          ;MEMORY MANAGEMENT ACTIVE ?
 5336  025526  001403                          BEQ     CMPRT1          ;BR IF OT
 5337  025530  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEMORY MANAGEMENT AGAIN
 5338  025536  005737  025416          CMPRT1: TST     LIMIT           ;TYPEOUT LIMIT REACHED ?
 5339  025542  001403                          BEQ     1$              ;BR IF IT HAS
 5340  025544  005337  025416                  DEC      LIMIT          ;DECREMENT LIMIT COUNTER
 5341  025550  001004                          BNE     2$              ;BR IF NOT AT LIMIT
 5342  025552  032777  000020  153360  1$:     BIT     #SW04,@SWR      ;PRINT ALL DATA COMPARE ERRORS ?
 5343  025560  001420                          BEQ     3$              ;BR IF NOT
 5344  025562  010137  001122          2$:     MOV     R1,$BDADR       ;ADDRESS OF BAD WORD
 5345  025566  162737  000002  001122          SUB     #2,$BDADR       ;ADJUST ADDRESS
 5346  025574  016537  177776  001124          MOV     -2(R5),$GDDAT   ;GOOD DATA
 5347  025602  016137  177776  001126          MOV     -2(R1),$BDDAT   ;BAD DATA
 5348  025610  112737  000104  001114          MOVB    #104,$ITEMB     ;ERROR TABLE INDEX
 5349  025616  004737  017254                  JSR     PC,TYPERR       ;DISPLAY WORD THAT DIDN'T COMPARE
 5350  025622  005237  025414          3$:     INC     ERCTR           ;COUNT THE ERROR
 5351  025626  000207                          RTS     PC              ;RETURN
 5352
 5353                                  ;LAST LINE OF COMPARE ERROR REPORTING
 5354
 5355  025630  005737  025414          ENDCMP: TST     ERCTR           ;SEE IF ANY ERRORS
 5356  025634  001406                          BEQ     2$              ;BR IF NONE
 5357  025636  012737  025646  001164          MOV     #1$,$ESCAPE     ;;ESCAPE TO 1$ ON ERROR
 5358  025644  104105                          ERROR   105             ;NUMBER OF COMPARE ERRORS
 5359  025646  005037  001164          1$:     CLR     $ESCAPE         ;CLEAR THE ERROR ESCAPE ADDRESS
 5360  025652  000207                  2$:     RTS     PC              ;RETURN
 5361
 5362                                  ;ROUTINE TO CLEAR THE RP11E
 5363                                  ;
 5364                                  ;THE FIRST 'CLEAR' TERMINATES ANY ORDER WHICH IS IN PROGRESS.  THE SECOND
 5365                                  ;'CLEAR' CLEARS THE 'PROGRAM ERROR' WHICH MAY HAVE BEEN SET BY THE FIRST
 5366                                  ;'CLEAR'.
 5367                                  ;
 5368                                  ;NOTE:  R4 MUST CONTAIN THE BASE ADDRESS OF THE RP11E
 5369
 5370  025654  012764  000001  000004  CLRP:   MOV     #CLEAR,RPCS(R4) ;CLEAR THE RP11
 5371  025662  051616                          BIS     (SP),(SP)       ;DELAY FOR...
 5372  025664  051616                          BIS     (SP),(SP)       ;...AT LEAST
 5373  025666  051616                          BIS     (SP),(SP)       ;...4 MICROSECONDS.
 5374  025670  012764  000001  000004          MOV     #CLEAR,RPCS(R4) ;CLEAR THE RP11
 5375  025676  000207                          RTS     PC              ;RETURN
 5376
 5377                                  ;ROUTINE TO SIZE MEMORY AND TO SETUP FOR TEST 14
 5378
 5379  025700  012737  000200  022232  SIZMEM: MOV     #BIT07,$KT11    ;TELL THE 'SIZE' ROUTINE TO USE MM
 5380  025706  004737  022174          6$:     JSR     PC,$SIZE        ;FIND OUT HOW MUCH MEMORY AND IF THE
 5381                                                                  ;SYSTEM HAS A KT11
 5382  025712  005737  022232                  TST     $KT11           ;DOES THE SYSTEM HAVE A KT11 ?
 5383  025716  100060                          BPL     2$              ;BR IF NOT
 5384  025720  023727  022500  001540          CMP     $LSTBK,#1540    ;MORE THAN 28K ON THE SYSTEM ?
 5385  025726  101003                          BHI     1$              ;BR IF MORE THAN 28K
```

```
 5386  025730  005037  022232                    CLR     $KT11           ;CLEAR MEMORY MANAGEMENT INDICATOR
 5387  025734  000764                            BR      6$              ;DO IT AGAIN WITH MEM. MAN. DESELECTED
 5388  025736  012737  157776  001302  1$:       MOV     #157776,MEMSIZ  ;ADDRESS OF LAST NON MEMORY MANAGEMENT LOCATION
 5389  025744  013737  022500  001230            MOV     $LSTBK,MAXMEM   ;HIGH MEMORY BANK ADDRESS
 5390  025752  005037  001232                    CLR     MAXMEM+2        ;CLEAR UPPER MEMORY ADDRESS BITS
 5391  025756  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
 5392  025762  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
 5393  025766  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
 5394  025772  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
 5395  025776  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
 5396  026002  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
 5397  026006  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
 5398  026012  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
 5399  026016  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
 5400  026022  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
 5401  026026  006337  001230                    ASL     MAXMEM          ;CONVERT THE BANK COUNT INTO AN
 5402  026032  006137  001232                    ROL     MAXMEM+2        ;ABSOLUTE MEMORY ADDRESS
 5403  026036  063737  022476  001230            ADD     $LSTAD,MAXMEM   ;ADD THE SIZE OF THE LAST BANK
 5404  026044  005537  001232                    ADC     MAXMEM+2        ;HANDLE ANY CARRY
 5405  026050  012737  020000  001264            MOV     #8192.,MAXWC    ;SETUP FOR MAXIMUM TRANSFER SIZE
 5406  026056  000433                            BR      5$              ;EXIT
 5407  026060  013737  022476  001302  2$:       MOV     $LSTAD,MEMSIZ   ;HIGH NON MEMORY MANAGEMENT ADDRESS
 5408  026066  013737  022476  001230            MOV     $LSTAD,MAXMEM   ;HIGH NON MEMORY MANAGEMENT ADDRESS
 5409  026074  005037  001232                    CLR     MAXMEM+2        ;DON'T NEED THE UPPER WORD
 5410  026100  013737  001230  001264            MOV     MAXMEM,MAXWC    ;CONVERT MEMORY SIZE TO WORD COUNT
 5411  026106  162737  043656  001264            SUB     #BUFFER,MAXWC   ;SUBTRACT STARTING ADDRESS OF BUFFER
 5412  026114  000241                            CLC                     ;CLEAR THE 'C' BIT
 5413  026116  006037  001264                    ROR     MAXWC           ;CHANGE TO WORD COUNT
 5414  026122  105737  000041                    TSTB    41              ;SEE WHICH LOADER
 5415  026126  001004                            BNE     4$              ;BR IF 'XXDP'
 5416  026130  162737  000144  001264  5$:       SUB     #100.,MAXWC     ;'ABS' LOADER SIZE (PLUS A LITTLE BIT)
 5417  026136  000403                            BR      5$              ;EXIT
 5418  026140  162737  002734  001264  4$:       SUB     #1500.,MAXWC    ;'XXDP' SIZE
 5419  026146  000207                  5$:       RTS     PC              ;RETURN
 5420
 5421                                  ;THIS ROUTINE IS USED TO ENSURE THAT THE BUS ADDRESS
 5422                                  ;OF THE RP11 IS SETUP TO READ THE PROPER VALUE.
 5423                                  ;IT WILL ALSO READ THE ADDRESS FROM THE TTY IF
 5424                                  ;REQUIRED.
 5425                                  ;NOTE: THIS ROUTINE DESTROYS R0 - R4
 5426
 5427  026150  005737  001216          GETADR: TST     BUSADR          ;INPUT FROM TTY REQUESTED?
 5428  026154  001425                          BEQ     3$              ;NO--BRANCH
 5429  026156  005037  001216                  CLR     BUSADR          ;YES--CLEAR THE REQUEST FLAG
 5430  026162  104401  001173          1$:     TYPE    ,$CRLF          ;CR-LF
 5431  026166  104401  032162                  TYPE    ,MRPADR         ;'RPADR='
 5432  026172  013746  001324                  MOV     RPADR,-(SP)     ;;SAVE RPADR FOR TYPEOUT
 5433                                                                  ;;RP11 ADDRESS
 5434  026176  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
 5435  026200     006                          .BYTE   6               ;;TYPE 6 DIGIT(S)
 5436  026201     001                          .BYTE   1               ;;TYPE LEADING ZEROS
 5437  026202  104401  031322                  TYPE    ,SLASH          ;'/'
 5438  026206  104411                          RDLIN                   ;GET THE ENTRY
 5439  026210  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
 5440  026212  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
 5441  026216  026162                          1$                      ;ILLEGAL INPUT
```

```
5442  026220  026224                          2$                      ;TERMINATED WITH A 'CR'
5443  026222  026230                          3$                      ;NO ENTRY - 'CR' ONLY
5444  026224  010237  001324          2$:     MOV     R2,RPADR        ;STORE THE ADDRESS
5445  026230  013701  000004          3$:     MOV     ERRVEC,R1       ;SAVE THE ERROR VECTOR
5446  026234  012737  026254  000004          MOV     #4$,ERRVEC      ;SETUP FOR TRAP
5447  026242  005777  153056                  TST     @RPADR          ;CHECK FOR RP11
5448  026246  010137  000004                  MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
5449  026252  000207                          RTS     PC              ;RETURN
5450  026254  010137  000004          4$:     MOV     R1,ERRVEC       ;RESTORE ERROR VECTOR
5451  026260  022626                          CMP     (SP)+,(SP)+     ;CLEAN OFF THE STACK
5452  026262  012737  026272  001164          MOV     #5$,$ESCAPE     ;;ESCAPE TO 5$ ON ERROR
5453  026270  104001                          ERROR   1               ;REPORT THE ERROR
5454  026272  005037  001164          5$:     CLR     $ESCAPE         ;CLEAR ERROR ESCAPE ADDRESS
5455  026276  005737  000042                  TST     @#42            ;IS THERE A MONITOR?
5456  026302  001727                          BEQ     1$              ;NO--GO ASK FOR ADDRESS
5457  026304  005037  016776                  CLR     $EOPCT          ;NO PASSES
5458  026310  000137  016616                  JMP     $EOP            ;GO TO END OF PROGRAM
5459
5460                                  ;THIS ROUTINE ALLOWS THE OPERATOR TO CONTROL THE PROGRAM
5461                                  ;       (CONVERSATION MODE)
5462
5463  026314  104401  031324          ENTPRM: TYPE    ,ENTNM          ;'ENTER TEST NUMBER'
5464  026320  104411                          RDLIN                   ;GET THE ENTRY
5465  026322  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
5466  026324  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
5467  026330  026314                          ENTPRM                  ;ILLEGAL INPUT
5468  026332  026336                          1$                      ;TERMINATED WITH A 'CR'
5469  026334  026362                          2$                      ;NO ENTRY - 'CR' ONLY
5470  026336  022702  000020          1$:     CMP     #$TN,R2         ;VALID NUMBER ?
5471  026342  101764                          BLOS    ENTPRM          ;BR IF NOT
5472  026344  006302                          ASL     R2              ;CONVERT INTO A BIT INDEX
5473  026346  016237  001400  001276          MOV     BITS(R2),TSTNMS ;TEST SELECTION BIT
5474  026354  006202                          ASR     R2              ;CHANGE BACK INTO A WORD COUNT
5475  026356  010246                          MOV     R2,-(SP)        ;SAVE THE TEST NUMBER ENTRY
5476  026360  000405                          BR      ENTPR1          ;GET THE DRIVE NUMBER
5477  026362  012737  017777  001276  2$:     MOV     #17777,TSTNMS   ;SELECT ALL TESTS
5478  026370  012746  177777                  MOV     #-1,-(SP)       ;ALL TESTS SELECTED
5479  026374  005037  001312          ENTPR1: CLR     DRVSEL          ;CLEAR DRIVE SELECTION WORD
5480  026400  004737  022566                  JSR     PC,RPINIT       ;CHECK THE DRIVE STATUS
5481  026404  012737  000312  001210          MOV     #202.,MAXCYL    ;ASSUME ONLY RP02'S
5482  026412  005737  001304                  TST     DRVTYP          ;SEE WHICH TYPE
5483  026416  001403                          BEQ     1$              ;BR IF ONLY RP02'S
5484  026420  012737  000625  001210          MOV     #405.,MAXCYL    ;RP03'S
5485  026426  104401  031352          1$:     TYPE    ,DRVNM          ;ENTER DRIVE NUMBER
5486  026432  104411                          RDLIN                   ;GET THE ENTRY
5487  026434  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
5488  026436  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
5489  026442  026426                          1$                      ;ILLEGAL INPUT
5490  026444  026450                          2$                      ;TERMINATED WITH A 'CR'
5491  026446  026524                          5$                      ;NO ENTRY - 'CR' ONLY
5492  026450  022702  000010          2$:     CMP     #8.,R2          ;VALID ENTRY ?
5493  026454  101764                          BLOS    1$              ;BR IF NOT
5494  026456  116237  001360  001312          MOVB    ATABIT(R2),DRVSEL  ;DRIVE SELECTION BIT
5495  026464  105762  001370                  TSTB    DRVSTA(R2)      ;IS DRIVE ONLINE
5496  026470  003035                          BGT     8$              ;BR IF ONLINE
5497  026472  104401  031377                  TYPE    ,DRIVE          ;'DRIVE'
```

```
 5498  026476  105762  001370            TSTB    DRVSTA(R2)          ;CHECK DRIVE STATUS AGAIN
 5499  026502  100403                    BMI     3$                  ;BR IF UNSAFE
 5500  026504  104401  027312            TYPE    ,OFFLIN             ;'OFFLINE'
 5501  026510  000402                    BR      4$                  ;CONTINUE
 5502  026512  104401  027276    3$:     TYPE    ,UNSAFE             ;'UNSAFE'
 5503  026516  104401  001173    4$:     TYPE    ,$CRLF              ;CR-LF
 5504  026522  000724                    BR      ENTPR1              ;TRY AGAIN
 5505  026524  005001            5$:     CLR     R1                  ;CLEAR R1 - USE IT AS AN INDEX
 5506  026526  005716                    TST     (SP)                ;ALL TESTS SELECTED ?
 5507  026530  100403                    BMI     6$                  ;BR IF THEY WERE
 5508  026532  022716  000015            CMP     #15,(SP)            ;TEST 15, 16, OR 17 SELECTED ?
 5509  026536  101733                    BLOS    1$                  ;BR IF YES, SPECIFIC DRIVE NUMBER REQUIRED
 5510  026540  105761  001370    6$:     TSTB    DRVSTA(R1)          ;IS THE DRIVE ONLINE ?
 5511  026544  003403                    BLE     7$                  ;BR IF NOT
 5512  026546  156137  001360  001312    BISB    ATABIT(R1),DRVSEL   ;SET THE SELECT BIT FOR THE DRIVE
 5513  026554  005201            7$:     INC     R1                  ;INCREMENT THE DRIVE INDEX
 5514  026556  020127  000010            CMP     R1,#8.              ;REACHED MAXIMUM ?
 5515  026562  103766                    BLO     6$                  ;BR IF NOT
 5516  026564  022726  000014    8$:     CMP     #14,(SP)+           ;IS TEST 14 SELECTED ?
 5517  026570  001137                    BNE     ENTPRX              ;BR IF NOT
 5518  026572  104401  031406    ENTPR2: TYPE    ,CHNGPM             ;CHANGE PARAMETERS FOR TEST 14 ?
 5519  026576  104411                    RDLIN                       ;GET THE RESPONSE
 5520  026600  123627  000131            CMPB    @(SP)+,#'Y          ;'Y' ENTERED ?
 5521  026604  001131                    BNE     ENTPRX              ;BR IF NOT
 5522  026606  104401  031447            TYPE    ,MXWRDS             ;'MAXIMUM WORD COUNT FOR TEST 14: '
 5523  026612  013746  001264            MOV     MAXWC,-(SP)         ;;SAVE MAXWC FOR TYPEOUT
 5524                                                                ;;MAXIMUM WORD COUNT FOR TEST 14
 5525  026616  104403                    TYPOS                       ;;GO TYPE--OCTAL ASCII
 5526  026620     005                    .BYTE   5                   ;;TYPE 5 DIGIT(S)
 5527  026621     000                    .BYTE   0                   ;;SUPPRESS LEADING ZEROS
 5528  026622  104401  001173            TYPE    ,$CRLF              ;CR-LF
 5529  026626  104401  031523    1$:     TYPE    ,ENTWC              ;'ENTER WORD COUNT FOR TEST 14'
 5530  026632  104411                    RDLIN                       ;GET THE ENTRY
 5531  026634  012601                    MOV     (SP)+,R1            ;STARTING ADDRESS OF INPUT ASCII STRING
 5532  026636  004037  027136            JSR     R0,CK.NUM           ;CHECK THE NUMBER
 5533  026642  026626                    1$                          ;ILLEGAL INPUT
 5534  026644  026650                    2$                          ;TERMINATED WITH A 'CR'
 5535  026646  026670                    ENTPR3                      ;NO ENTRY - 'CR' ONLY
 5536  026650  020237  001264    2$:     CMP     R2,MAXWC            ;VALUE TOO LARGE ?
 5537  026654  101364                    BHI     1$                  ;BR IF TOO LARGE
 5538  026656  020237  001262            CMP     R2,LMT8K            ;NEW WORD COUNT LARGER THAN 8K ?
 5539  026662  101361                    BHI     1$                  ;BR IF LARGER
 5540  026664  010237  001256            MOV     R2,WC14             ;LOAD NEW WORD COUNT
 5541  026670  104401  031565    ENTPR3: TYPE    ,CHNGPT             ;ENTER PATTERN CODE
 5542  026674  104411                    RDLIN                       ;READ THE ENTRY
 5543  026676  012601                    MOV     (SP)+,R1            ;STARTING ADDRESS OF INPUT ASCII STRING
 5544  026700  004037  027136            JSR     R0,CK.NUM           ;CHECK THE NUMBER
 5545  026704  026670                    ENTPR3                      ;ILLEGAL INPUT
 5546  026706  026712                    1$                          ;TERMINATED WITH A 'CR'
 5547  026710  026716                    ENTPR4                      ;NO ENTRY - 'CR' ONLY
 5548  026712  010237  001300    1$:     MOV     R2,PATTRN           ;PATTERN CODE
 5549  026716  104401  031640    ENTPR4: TYPE    ,CHNGAD             ;USE A SPECIFIC DISK ADDRESS ?
 5550  026722  104411                    RDLIN                       ;READ THE ENTRY
 5551  026724  123627  000131            CMPB    @(SP)+,#'Y          ;'Y' ENTERED ?
 5552  026730  001057                    BNE     ENTPRX              ;BR IF NOT
 5553  026732  104401  031713            TYPE    ,ONEPAT             ;'A SINGLE DATA PATTERN MUST HAVE BEEN SPECIFIED;
```

```
 5554                                                    ;'IF NOT, PATTERN 0 (ALL ZEROS) WILL BE USED'
 5555  026736  104401  032054          1$:     TYPE    ,ENTCYL         ;'ENTER CYLINDER ADDRESS'
 5556  026742  104411                          RDLIN                   ;GET THE ENTRY
 5557  026744  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
 5558  026746  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
 5559  026752  026736                          1$                      ;ILLEGAL INPUT
 5560  026754  026760                          2$                      ;TERMINATED WITH A 'CR'
 5561  026756  026736                          1$                      ;NO ENTRY - 'CR' ONLY
 5562  026760  020237  001210          2$:     CMP     R2,MAXCYL       ;TOO LARGE ?
 5563  026764  101364                          BHI     1$              ;BR IF TOO LARGE
 5564  026766  010237  001266                  MOV     R2,SCYL         ;LOAD CYLINDER ADDRESS
 5565  026772  104401  032105          3$:     TYPE    ,ENTRK          ;'ENTER TRACK ADDRESS'
 5566  026776  104411                          RDLIN                   ;GET THE ENTRY
 5567  027000  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
 5568  027002  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
 5569  027006  026772                          3$                      ;ILLEGAL INPUT
 5570  027010  027014                          4$                      ;TERMINATED WITH A 'CR'
 5571  027012  026772                          3$                      ;NO ENTRY - 'CR' ONLY
 5572  027014  020227  000023          4$:     CMP     R2,#19.         ;VALID ENTRY ?
 5573  027020  101364                          BHI     3$              ;BR IF NOT
 5574  027022  010237  001270                  MOV     R2,STRK         ;TRACK ADDRESS
 5575  027026  104401  032133          5$:     TYPE    ,ENTSEC         ;'ENTER SECTOR ADDRESS'
 5576  027032  104411                          RDLIN
 5577  027034  012601                          MOV     (SP)+,R1        ;STARTING ADDRESS OF INPUT ASCII STRING
 5578  027036  004037  027136                  JSR     R0,CK.NUM       ;CHECK THE NUMBER
 5579  027042  027026                          5$                      ;ILLEGAL INPUT
 5580  027044  027050                          6$                      ;TERMINATED WITH A 'CR'
 5581  027046  027026                          5$                      ;NO ENTRY - 'CR' ONLY
 5582  027050  020227  000011          6$:     CMP     R2,#9.          ;VALID ENTRY ?
 5583  027054  101364                          BHI     5$              ;BR IF NOT
 5584  027056  010237  001272                  MOV     R2,SSEC         ;LOAD SECTOR ADDRESS
 5585  027062  012737  177777  001260        MOV     #-1,OPRSEL      ;SET ADDRESS FLAG
 5586  027070  000207                ENTPRX: RTS     PC              ;RETURN
 5587
 5588                                          ;THIS ROUTINE IS USED TO CHECK IF AN ASCII CHARACTER IS A DIGIT
 5589                                          ;BETWEEN 0 AND 7.
 5590                                          ;
 5591                                          ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
 5592                                          ;       JSR     R0,CK.OCT       ;CHECK THE CHARACTER
 5593                                          ;       RETURN1                 ;CHARACTER IS IN R2 AS A
 5594                                          ;                               ;OCTAL DIGIT
 5595                                          ;       RETURN2                 ;CHARACTER IS NOT BETWEEN 0-7
 5596
 5597  027072  121127  000060          CK.OCT: CMPB    (R1),#'0        ;LESS THAN ZERO?
 5598  027076  103407                          BLO     1$              ;YES -- BRANCH
 5599  027100  121127  000067                  CMPB    (R1),#'7        ;GREATER THAN SEVEN?
 5600  027104  101004                          BHI     1$              ;YES -- BRANCH
 5601  027106  111102                          MOVB    (R1),R2         ;GET THE CHARACTER
 5602  027110  042702  177770                  BIC     # C7,R2         ;STRIP AWAY THE ASCII
 5603  027114  000401                          BR      2$              ;BYPASS RETURN ADJUST
 5604  027116  005720                  1$:     TST     (R0)+           ;ADJUST FOR RETURN
 5605  027120  000200                  2$:     RTS     R0              ;RETURN
 5606
 5607                                          ;THIS ROUTINE WILL CHECK AN ASCII CHARACTER TO
 5608                                          ;DETERMINE WHAT IT IS.
 5609                                          ;
```

```
5610                                    ;       MOV     #ADR,R1         ;ADDRESS OF ASCII CHARACTER
5611                                    ;       JSR     R0,CK.CHR       ;CHECK CHARACTER
5612                                    ;       RETURN  ADR1            ;UNKNOWN CHARACTER
5613                                    ;       RETURN  ADR2            ;CARRIAGE RETURN * (R1)=ADR+1
5614
5615  027122  105711                    CK.CHR: TSTB    (R1)            ;"CARRIAGE RETURN"?
5616  027124  001002                            BNE     1$              ;NO -- BRANCH
5617  027126  005720                            TST     (R0)+           ;CARRIAGE RETURN
5618  027130  005201                            INC     R1              ;MOVE POINTER TO NEXT CHARACTER
5619  027132  011000                    1$:     MOV     (R0),R0         ;UNKNOWN CHARACTER
5620  027134  000200                            RTS     R0              ;RETURN
5621
5622
5623                                    ;THIS ROUTINE CHECKS AN ASCIZ STRING FOR LEGAL CHARACTERS
5624                                    ;AND FORMS AN OCTAL NUMBER IN R2
5625                                    ;
5626                                    ;       MOV     #ADR,R1         ;ADDRESS OF ASCIZ STRING
5627                                    ;       JSR     R0,CK.NUM       ;GO FORM THE NUMBER
5628                                    ;       RETURN  ADR1            ;ILLEGAL CHARACTER IN THE INPUT STRING
5629                                    ;       RETURN  ADR2            ;"CR"--R2=NUMBER
5630                                    ;       RETURN  ADR3            ;'CR' WAS FIRST ENTRY
5631
5632  027136  010346                    CK.NUM: MOV     R3,-(SP)        ;SAVE R3
5633  027140  005003                            CLR     R3              ;START NUMBER AT ZERO
5634  027142  004037  027072                    JSR     R0,CK.OCT       ;OCTAL DIGIT?
5635  027146  000404                            BR      1$              ;YES--BRANCH
5636  027150  004037  027122                    JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
5637  027154  027230                            6$                      ;ILLEGAL CHARACTER
5638  027156  027222                            8$                      ;CARRIAGE RETURN
5639  027160  005201                    1$:     INC     R1              ;MOVE TO NEXT CHARACTER
5640  027162  006303                            ASL     R3              ;FOR THE OCTAL NUMBER IN R3
5641  027164  103421                            BCS     6$              ;DON'T LET IT GET TO BIG
5642  027166  006303                            ASL     R3
5643  027170  103417                            BCS     6$
5644  027172  006303                            ASL     R3
5645  027174  103415                            BCS     6$
5646  027176  060203                            ADD     R2,R3
5647  027200  004037  027072                    JSR     R0,CK.OCT       ;IS THIS AN OCTAL DIGIT?
5648  027204  000765                            BR      1$              ;YES--MAKE IT PART OF THE NUMBER
5649  027206  010302                    2$:     MOV     R3,R2           ;SAVE THE OCTAL NUMBER
5650  027210  005003                            CLR     R3              ;START WITH ZERO INDEX
5651  027212                            3$:
5652  027212  004037  027122                    JSR     R0,CK.CHR       ;CHECK ONE CHARACTER
5653  027216  027230                            6$                      ;ILLEGAL CHARACTER
5654  027220  027224                            5$                      ;CARRIAGE RETURN
5655  027222  005723                    8$:     TST     (R3)+           ;'CR' ONLY
5656  027224  005723                    5$:     TST     (R3)+           ;"CR"
5657  027226  060300                            ADD     R3,R0           ;YES--SAVE THE OCTAL NUMBER
5658  027230  012603                    6$:     MOV     (SP)+,R3        ;RESTORE R3
5659  027232  011000                            MOV     (R0),R0         ;PICKUP EXIT ADDRESS
5660  027234  000200                            RTS     R0              ;RETURN
5661
5662                                    .SBTTL  TELETYPE MESSAGES
5663
5664  027236  005015  042012  044522  DRSTAT: .ASCIZ  <15><12><12>@DRIVE STATUS:@<15><12><12>
5665  027244  042526  051440  040524
```

```
 5666  027252  052524  035123  005015
 5667  027260  000012
 5668  027262  020040  020040  047440  ONLINE: .ASCIZ  @     ONLINE@
 5669  027270  046116  047111  000105
 5670  027276  020040  020040  052440  UNSAFE: .ASCIZ  @     UNSAFE@
 5671  027304  051516  043101  000105
 5672  027312  020040  020040  047440  OFFLIN: .ASCIZ  @     OFFLINE@
 5673  027320  043106  044514  042516
 5674  027326     000
 5675  027327     040  051040  030120  RP02:   .ASCIZ  @  RP02@
 5676  027334  000062
 5677  027336  020040  050122  031460  RP03:   .ASCIZ  @  RP03@
 5678  027344     000
 5679  027345     104  044522  042526  DRVTST: .ASCIZ  @DRIVE(S) TO BE USED @
 5680  027352  051450  020051  047524
 5681  027360  041040  020105  051525
 5682  027366  042105  000040
 5683  027372  047516  042516     000  NONE:   .ASCIZ  @NONE@
 5684  027377     054     000          COMMA:  .ASCIZ  @,@
 5685  027401     015  005012  042524  TSTING: .ASCIZ  <15><12><12>@TESTING WITH DRIVE @
 5686  027406  052123  047111  020107
 5687  027414  044527  044124  042040
 5688  027422  044522  042526  000040
 5689  027430  020040     000          BLNKS2: .ASCIZ  @  @
 5690
 5691  027433     015  005012  047520  MSG15A: .ASCIZ  <15><12><12>@POWER FAIL TEST@
 5692  027440  042527  020122  040506
 5693  027446  046111  052040  051505
 5694  027454  000124
 5695  027456  005015  052012  051125  MSG15B: .ASCII  <15><12><12>@TURN CPU POWER  O F F  AFTER THIS MESSAGE, WAIT@<15><12>
 5696  027464  020116  050103  020125
 5697  027472  047520  042527  020122
 5698  027500  047440  043040  043040
 5699  027506  020040  043101  042524
 5700  027514  020122  044124  051511
 5701  027522  046440  051505  040523
 5702  027530  042507  020054  040527
 5703  027536  052111  005015
 5704  027542  020065  042523  047503          .ASCIZ  @5 SECONDS, THEN TURN CPU POWER  O N  AGAIN@<15><12>
 5705  027550  042116  026123  052040
 5706  027556  042510  020116  052524
 5707  027564  047122  041440  052520
 5708  027572  050040  053517  051105
 5709  027600  020040  020117  020116
 5710  027606  040440  040507  047111
 5711  027614  005015     000
 5712  027617     015  042412  042116  MSG15C: .ASCIZ  <15><12>@END OF POWER FAIL TEST@<15><12>
 5713  027624  047440  020106  047520
 5714  027632  042527  020122  040506
 5715  027640  046111  052040  051505
 5716  027646  006524  000012
 5717  027652  005015  050012  047522  MSG15D: .ASCIZ  <15><12><12>@PROGRAM WILL LOOP, WAITING FOR DRIVE TO COME ONLINE@<15><12
 5718  027660  051107  046501  053440
 5719  027666  046111  020114  047514
 5720  027674  050117  020054  040527
 5721  027702  052111  047111  020107
```

```
 5722  027710  047506  020122  051104
 5723  027716  053111  020105  047524
 5724  027724  041440  046517  020105
 5725  027732  047117  044514  042516
 5726  027740  005015     000
 5727
 5728  027743     015  005012  042510  MSG16A: .ASCII  <15><12><12>@HEAD ALIGNMENT ROUTINE@
 5729  027750  042101  040440  044514
 5730  027756  047107  042515  052116
 5731  027764  051040  052517  044524
 5732  027772  042516
 5733  027774  005015  041412  047117          .ASCII  <15><12><12>@CONNECT A JUMPER BETWEEN PINS J13M2 & J13C2 ON THE RP11E BA
 5734  030002  042516  052103  040440
 5735  030010  045040  046525  042520
 5736  030016  020122  042502  053524
 5737  030024  042505  020116  044520
 5738  030032  051516  045040  031461
 5739  030040  031115  023040  045040
 5740  030046  031461  031103  047440
 5741  030054  020116  044124  020105
 5742  030062  050122  030461  020105
 5743  030070  040502  045503  046120
 5744  030076  047101     105
 5745  030101     015  020012  051050          .ASCII  <15><12>@ (REFER TO THE APPLICABLE MAINTENANCE MANUAL FOR THE DRIVE SETU
 5746  030106  043105  051105  052040
 5747  030114  020117  044124  020105
 5748  030122  050101  046120  041511
 5749  030130  041101  042514  046440
 5750  030136  044501  052116  047105
 5751  030144  047101  042503  046440
 5752  030152  047101  040525  020114
 5753  030160  047506  020122  044124
 5754  030166  020105  051104  053111
 5755  030174  020105  042523  052524
 5756  030202  027120     051
 5757  030205     015  005012  047524          .ASCIZ  <15><12><12>@TOGGLE SWITCH 7 FOR HEAD SELECTION@<15><12>
 5758  030212  043507  042514  051440
 5759  030220  044527  041524  020110
 5760  030226  020067  047506  020122
 5761  030234  042510  042101  051440
 5762  030242  046105  041505  044524
 5763  030250  047117  005015     000
 5764  030255     015  042412  052116  MSG16B: .ASCIZ  <15><12>@ENTER HEAD ADDRESS (IN OCTAL): @
 5765  030262  051105  044040  040505
 5766  030270  020104  042101  051104
 5767  030276  051505  020123  044450
 5768  030304  020116  041517  040524
 5769  030312  024514  020072     000
 5770
 5771  030317     015  005012  050122  MSG17A: .ASCIZ  <15><12><12>@RP11E CONTROL PANEL SWITCH TEST@
 5772  030324  030461  020105  047503
 5773  030332  052116  047522  020114
 5774  030340  040520  042516  020114
 5775  030346  053523  052111  044103
 5776  030354  052040  051505  000124
 5777  030362  005015  054524  042520  MSG17B: .ASCIZ  <15><12>@TYPE A 'CR' TO START TEST AFTER EACH SETUP@<15><12>
```

```
 5778  030370  040440  023440  051103
 5779  030376  020047  047524  051440
 5780  030404  040524  052122  052040
 5781  030412  051505  020124  043101
 5782  030420  042524  020122  040505
 5783  030426  044103  051440  052105
 5784  030434  050125  005015     000
 5785  030441     015  042012  051511  MSG17C: .ASCIZ  <15><12>@DISABLE DRIVE # @
 5786  030446  041101  042514  042040
 5787  030454  044522  042526  021440
 5788  030462  000040
 5789  030464  005015  047105  041101  MSG17D: .ASCIZ  <15><12>@ENABLE DRIVE # @
 5790  030472  042514  042040  044522
 5791  030500  042526  021440  000040
 5792  030506  005015  042523  020124  MSG17E: .ASCIZ  <15><12>@SET 'READ ONLY' ON DRIVE @
 5793  030514  051047  040505  020104
 5794  030522  047117  054514  020047
 5795  030530  047117  042040  044522
 5796  030536  042526  000040
 5797  030542  005015  042523  020124  MSG17F: .ASCIZ  <15><12>@SET 'READ-WRITE' ON DRIVE @
 5798  030550  051047  040505  026504
 5799  030556  051127  052111  023505
 5800  030564  047440  020116  051104
 5801  030572  053111  020105     000
 5802  030577     015  051412  052105  MSG17G: .ASCII  <15><12>@SET 'WRITE LOCKOUT' AND CLEAR THE LOA SWITCHES@
 5803  030604  023440  051127  052111
 5804  030612  020105  047514  045503
 5805  030620  052517  023524  040440
 5806  030626  042116  041440  042514
 5807  030634  051101  052040  042510
 5808  030642  046040  040517  051440
 5809  030650  044527  041524  042510
 5810  030656     123
 5811  030657     015  047412  020116          .ASCIZ  <15><12>@ON THE RP11E CONTROL PANEL@
 5812  030664  044124  020105  050122
 5813  030672  030461  020105  047503
 5814  030700  052116  047522  020114
 5815  030706  040520  042516  000114
 5816  030714  005015  042523  020124  MSG17H: .ASCIZ  <15><12>@SET THE FOLLOWING OCTAL VALUES IN THE CYLINDER LOA SWITCHES: @
 5817  030722  044124  020105  047506
 5818  030730  046114  053517  047111
 5819  030736  020107  041517  040524
 5820  030744  020114  040526  052514
 5821  030752  051505  044440  020116
 5822  030760  044124  020105  054503
 5823  030766  044514  042116  051105
 5824  030774  046040  040517  051440
 5825  031002  044527  041524  042510
 5826  031010  035123  000040
 5827  031014  005015  042523  020124  MSG17I: .ASCIZ  <15><12>@SET SWITCHES TO @
 5828  031022  053523  052111  044103
 5829  031030  051505  052040  020117
 5830  031036     000
 5831  031037     015  051012  051505  MSG17J: .ASCII  <15><12>@RESET THE CYLINDER LOA SWITCHES@
 5832  031044  052105  052040  042510
 5833  031052  041440  046131  047111
```

```
 5834  031060  042504  020122  047514
 5835  031066  020101  053523  052111
 5836  031074  044103  051505
 5837  031100  005015  051412  052105          .ASCIZ  <15><12><12>@SET THE FOLLOWING OCTAL VALUES IN THE DRIVE LOA SWITCHES: @
 5838  031106  052040  042510  043040
 5839  031114  046117  047514  044527
 5840  031122  043516  047440  052103
 5841  031130  046101  053040  046101
 5842  031136  042525  020123  047111
 5843  031144  052040  042510  042040
 5844  031152  044522  042526  046040
 5845  031160  040517  051440  044527
 5846  031166  041524  042510  035123
 5847  031174  000040
 5848  031176  005015  042412  042116  MSG17K: .ASCII  <15><12><12>@END OF CONROL PANEL SWITCH TEST@
 5849  031204  047440  020106  047503
 5850  031212  051116  046117  050040
 5851  031220  047101  046105  051440
 5852  031226  044527  041524  020110
 5853  031234  042524  052123
 5854  031240  005015  042522  052524          .ASCIZ  <15><12>@RETURN RP11E CONTROL PANEL SWITCHES TO NORMAL@<15><12>
 5855  031246  047122  051040  030520
 5856  031254  042461  041440  047117
 5857  031262  051124  046117  050040
 5858  031270  047101  046105  051440
 5859  031276  044527  041524  042510
 5860  031304  020123  047524  047040
 5861  031312  051117  040515  006514
 5862  031320  000012
 5863
 5864  031322  000057                      SLASH:  .ASCIZ  @/@
 5865  031324  005015  047105  042524      ENTNM:  .ASCIZ  <15><12>@ENTER TEST NUMBER: @
 5866  031332  020122  042524  052123
 5867  031340  047040  046525  042502
 5868  031346  035122  000040
 5869  031352  047105  042524  020122      DRVNM:  .ASCIZ  @ENTER DRIVE NUMBER: @
 5870  031360  051104  053111  020105
 5871  031366  052516  041115  051105
 5872  031374  020072     000
 5873  031377     104  044522  042526      DRIVE:  .ASCIZ  @DRIVE @
 5874  031404  000040
 5875  031406  044103  047101  042507      CHNGPM: .ASCIZ  @CHANGE PARAMETERS FOR TEST 14 ? @
 5876  031414  050040  051101  046501
 5877  031422  052105  051105  020123
 5878  031430  047506  020122  042524
 5879  031436  052123  030440  020064
 5880  031444  020077     000
 5881  031447     115  054101  046511      MXWRDS: .ASCIZ  @MAXIMUM WORD COUNT FOR TEST 14 (IN OCTAL): @
 5882  031454  046525  053440  051117
 5883  031462  020104  047503  047125
 5884  031470  020124  047506  020122
 5885  031476  042524  052123  030440
 5886  031504  020064  044450  020116
 5887  031512  041517  040524  024514
 5888  031520  020072     000
 5889  031523     105  052116  051105      ENTWC:  .ASCIZ  @ENTER NEW WORD COUNT (IN OCTAL): @
```

```
 5890  031530  047040  053505  053440
 5891  031536  051117  020104  047503
 5892  031544  047125  020124  044450
 5893  031552  020116  041517  040524
 5894  031560  024514  020072     000
 5895  031565     105  052116  051105  CHNGPT: .ASCIZ  @ENTER PATTERN SELECTION CODE FOR TEST 14: @
 5896  031572  050040  052101  042524
 5897  031600  047122  051440  046105
 5898  031606  041505  044524  047117
 5899  031614  041440  042117  020105
 5900  031622  047506  020122  042524
 5901  031630  052123  030440  035064
 5902  031636  000040
 5903  031640  051525  020105  020101  CHNGAD: .ASCIZ  @USE A SPECIFIC DISK ADDRESS FOR TEST 14 ? @
 5904  031646  050123  041505  043111
 5905  031654  041511  042040  051511
 5906  031662  020113  042101  051104
 5907  031670  051505  020123  047506
 5908  031676  020122  042524  052123
 5909  031704  030440  020064  020077
 5910  031712     000
 5911  031713     040  040450  051440  ONEPAT: .ASCII  @ (A SINGLE DATA PATTERN MUST HAVE BEEN SPECIFIED@<15><12>
 5912  031720  047111  046107  020105
 5913  031726  040504  040524  050040
 5914  031734  052101  042524  047122
 5915  031742  046440  051525  020124
 5916  031750  040510  042526  041040
 5917  031756  042505  020116  050123
 5918  031764  041505  043111  042511
 5919  031772  006504     012
 5920  031775     040  043111  047040          .ASCIZ  @ IF NOT, PATTERN 0 (ALL ZEROS) WILL BE USED)@<15><12>
 5921  032002  052117  020054  040520
 5922  032010  052124  051105  020116
 5923  032016  020060  040450  046114
 5924  032024  055040  051105  051517
 5925  032032  020051  044527  046114
 5926  032040  041040  020105  051525
 5927  032046  042105  006451  000012
 5928  032054  047105  042524  020122  ENTCYL: .ASCIZ  @ENTER CYLINDER ADDRESS: @
 5929  032062  054503  044514  042116
 5930  032070  051105  040440  042104
 5931  032076  042522  051523  020072
 5932  032104     000
 5933  032105     105  052116  051105  ENTRK:  .ASCIZ  @ENTER TRACK ADDRESS: @
 5934  032112  052040  040522  045503
 5935  032120  040440  042104  042522
 5936  032126  051523  020072     000
 5937  032133     105  052116  051105  ENTSEC: .ASCIZ  @ENTER SECTOR ADDRESS: @
 5938  032140  051440  041505  047524
 5939  032146  020122  042101  051104
 5940  032154  051505  035123  000040
 5941
 5942  032162  050122  042101  020122  MRPADR: .ASCIZ  @RPADR = @
 5943  032170  020075     000
 5944
 5945                                  .SBTTL  ERROR MESSAGES
```

```
 5946
 5947  032173     122  030520  020061  EM1:    .ASCIZ  @RP11 DIDN'T RESPOND TO ADDRESSING@
 5948  032200  044504  047104  052047
 5949  032206  051040  051505  047520
 5950  032214  042116  052040  020117
 5951  032222  042101  051104  051505
 5952  032230  044523  043516     000
 5953  032235     103  047101  052047  EM2:    .ASCIZ  @CAN'T START READ COMMAND@
 5954  032242  051440  040524  052122
 5955  032250  051040  040505  020104
 5956  032256  047503  046515  047101
 5957  032264  000104
 5958  032266  041447  042514  051101  EM3:    .ASCIZ  @'CLEAR' COMMAND DIDN'T TERMINATE DATA TRANSFER OPERATION@
 5959  032274  020047  047503  046515
 5960  032302  047101  020104  044504
 5961  032310  047104  052047  052040
 5962  032316  051105  044515  040516
 5963  032324  042524  042040  052101
 5964  032332  020101  051124  047101
 5965  032340  043123  051105  047440
 5966  032346  042520  040522  044524
 5967  032354  047117     000
 5968  032357     105  051122  051117  EM4:    .ASCIZ  @ERROR WRITING TEST SECTOR(S)@
 5969  032364  053440  044522  044524
 5970  032372  043516  052040  051505
 5971  032400  020124  042523  052103
 5972  032406  051117  051450  000051
 5973  032414  051127  052111  020105  EM5:    .ASCIZ  @WRITE CHECK ERROR CHECKING TEST SECTOR(S)@
 5974  032422  044103  041505  020113
 5975  032430  051105  047522  020122
 5976  032436  044103  041505  044513
 5977  032444  043516  052040  051505
 5978  032452  020124  042523  052103
 5979  032460  051117  051450  000051
 5980  032466  054105  042520  052103  EM6:    .ASCIZ  @EXPECTED 'WCE' DIDN'T OCCUR@
 5981  032474  042105  023440  041527
 5982  032502  023505  042040  042111
 5983  032510  023516  020124  041517
 5984  032516  052503  000122
 5985  032522  051105  047522  020122  EM7:    .ASCIZ  @ERROR WRITING LESS THAN A FULL SECTOR@
 5986  032530  051127  052111  047111
 5987  032536  020107  042514  051523
 5988  032544  052040  040510  020116
 5989  032552  020101  052506  046114
 5990  032560  051440  041505  047524
 5991  032566  000122
 5992  032570  047117  020105  043117  EM11:   .ASCIZ  @ONE OF 2 WORDS WRITTEN IN THE PARTIAL WRITE DIDN'T COMPARE@
 5993  032576  031040  053440  051117
 5994  032604  051504  053440  044522
 5995  032612  052124  047105  044440
 5996  032620  020116  044124  020105
 5997  032626  040520  052122  040511
 5998  032634  020114  051127  052111
 5999  032642  020105  044504  047104
 6000  032650  052047  041440  046517
 6001  032656  040520  042522     000
```

```
 6002  032663     116  047117  055055  EM12:   .ASCII  @NON-ZERO DATA IN PART OF SECTOR WHICH SHOULD HAVE@<15><12>
 6003  032670  051105  020117  040504
 6004  032676  040524  044440  020116
 6005  032704  040520  052122  047440
 6006  032712  020106  042523  052103
 6007  032720  051117  053440  044510
 6008  032726  044103  051440  047510
 6009  032734  046125  020104  040510
 6010  032742  042526  005015
 6011  032746  042502  047105  055040          .ASCIZ  @BEEN ZERO FILLED DURING PARTIAL WRITE@
 6012  032754  051105  020117  044506
 6013  032762  046114  042105  042040
 6014  032770  051125  047111  020107
 6015  032776  040520  052122  040511
 6016  033004  020114  051127  052111
 6017  033012  000105
 6018  033014  042447  050117  020047  EM13:   .ASCII  @'EOP' DIDN'T SET DURING A 2 SECTOR WRITE BEGINNING@
 6019  033022  044504  047104  052047
 6020  033030  051440  052105  042040
 6021  033036  051125  047111  020107
 6022  033044  020101  020062  042523
 6023  033052  052103  051117  053440
 6024  033060  044522  042524  041040
 6025  033066  043505  047111  044516
 6026  033074  043516
 6027  033076  005015  052101  052040          .ASCIZ  <15><12>@AT THE LAST SECTOR OF THE PACK@
 6028  033104  042510  046040  051501
 6029  033112  020124  042523  052103
 6030  033120  051117  047440  020106
 6031  033126  044124  020105  040520
 6032  033134  045503     000
 6033  033137     047  051105  023522  EM14:   .ASCIZ  @'ERR' DIDN'T SET WITH 'EOP'@
 6034  033144  042040  042111  023516
 6035  033152  020124  042523  020124
 6036  033160  044527  044124  023440
 6037  033166  047505  023520     000
 6038  033173     047  047505  023520  EM15:   .ASCIZ  @'EOP' DIDN'T CLEAR RPCA@
 6039  033200  042040  042111  023516
 6040  033206  020124  046103  040505
 6041  033214  020122  050122  040503
 6042  033222     000
 6043  033223     123  041525  020101  EM16:   .ASCIZ  @SUCA NOT CORRECT AFTER 'EOP'@
 6044  033230  047516  020124  047503
 6045  033236  051122  041505  020124
 6046  033244  043101  042524  020122
 6047  033252  042447  050117  000047
 6048  033260  051105  047522  020122  EM17:   .ASCIZ  @ERROR READING TEST SECTOR@
 6049  033266  042522  042101  047111
 6050  033274  020107  042524  052123
 6051  033302  051440  041505  047524
 6052  033310  000122
 6053  033312  047503  052116  047105  EM20:   .ASCII  @CONTENTS OF FIRST SECTOR ON PACK CHANGED AFTER@<15><12>
 6054  033320  051524  047440  020106
 6055  033326  044506  051522  020124
 6056  033334  042523  052103  051117
 6057  033342  047440  020116  040520
```

```
 6058  033350  045503  041440  040510
 6059  033356  043516  042105  040440
 6060  033364  052106  051105  005015
 6061  033372  047506  041522  047111          .ASCIZ  @FORCING 'EOP' DURING WRITE@
 6062  033400  020107  042447  050117
 6063  033406  020047  052504  044522
 6064  033414  043516  053440  044522
 6065  033422  042524     000
 6066  033425     047  051120  043517  EM21:   .ASCII  @'PROG' ERROR DIDN'T SET WHEN READ COMMAND ISSUED@
 6067  033432  020047  051105  047522
 6068  033440  020122  044504  047104
 6069  033446  052047  041440  052105
 6070  033454  053440  042510  020116
 6071  033462  042522  042101  041440
 6072  033470  046517  040515  042116
 6073  033476  044440  051523  042525
 6074  033504     104
 6075  033505     015  053412  044510          .ASCIZ  <15><12>@WHILE CONTROLLER BUSY@
 6076  033512  042514  041440  047117
 6077  033520  051124  046117  042514
 6078  033526  020122  052502  054523
 6079  033534     000
 6080  033535     047  051105  023522  EM22:   .ASCIZ  @'ERR' DIDN'T SET WITH 'PROG'@
 6081  033542  042040  042111  023516
 6082  033550  020124  042523  020124
 6083  033556  044527  044124  023440
 6084  033564  051120  043517  000047
 6085  033572  051105  047522  020122  EM23:   .ASCIZ  @ERROR ATTEMPTING TO MISFORMAT SECTOR 0@
 6086  033600  052101  042524  050115
 6087  033606  044524  043516  052040
 6088  033614  020117  044515  043123
 6089  033622  051117  040515  020124
 6090  033630  042523  052103  051117
 6091  033636  030040     000
 6092  033641     105  051122  051117  EM24:   .ASCIZ  @ERROR VERIFYING THE MISFORMATTED TEST HEADER@
 6093  033646  053040  051105  043111
 6094  033654  044531  043516  052040
 6095  033662  042510  046440  051511
 6096  033670  047506  046522  052101
 6097  033676  042524  020104  042524
 6098  033704  052123  044040  040505
 6099  033712  042504  000122
 6100  033716  044515  043123  051117  EM25:   .ASCIZ  @MISFORMATTED TEST HEADER IS NOT CORRECT@
 6101  033724  040515  052124  042105
 6102  033732  052040  051505  020124
 6103  033740  042510  042101  051105
 6104  033746  044440  020123  047516
 6105  033754  020124  047503  051122
 6106  033762  041505  000124
 6107  033766  044047  043116  020047  EM26:   .ASCIZ  @'HNF' DIDN'T SET WHEN SEARCHING FOR MISFORMATTED SECTOR@
 6108  033774  044504  047104  052047
 6109  034002  051440  052105  053440
 6110  034010  042510  020116  042523
 6111  034016  051101  044103  047111
 6112  034024  020107  047506  020122
 6113  034032  044515  043123  051117
```

```
 6114  034040  040515  052124  042105
 6115  034046  051440  041505  047524
 6116  034054  000122
 6117  034056  042047  045523  020047  EM27:   .ASCIZ  @'DSK' NOT SET WITH 'HNF'@
 6118  034064  047516  020124  042523
 6119  034072  020124  044527  044124
 6120  034100  023440  047110  023506
 6121  034106     000
 6122  034107     047  042510  020047  EM30:   .ASCIZ  @'HE' DIDN'T SET WITH 'HNF'@
 6123  034114  044504  047104  052047
 6124  034122  051440  052105  053440
 6125  034130  052111  020110  044047
 6126  034136  043116  000047
 6127  034142  042447  051122  020047  EM31:   .ASCIZ  @'ERR' NOT SET WITH 'HNF'@
 6128  034150  047516  020124  042523
 6129  034156  020124  044527  044124
 6130  034164  023440  047110  023506
 6131  034172     000
 6132  034173     105  051122  051117  EM32:   .ASCIZ  @ERROR WHILE RESTORING MISFORMATTED TEST HEADER@
 6133  034200  053440  044510  042514
 6134  034206  051040  051505  047524
 6135  034214  044522  043516  046440
 6136  034222  051511  047506  046522
 6137  034230  052101  042524  020104
 6138  034236  042524  052123  044040
 6139  034244  040505  042504  000122
 6140  034252  047503  052116  047522  EM33:   .ASCII  @CONTROLLER DIDN'T BECOME BUSY WHEN READ COMMAND@
 6141  034260  046114  051105  042040
 6142  034266  042111  023516  020124
 6143  034274  042502  047503  042515
 6144  034302  041040  051525  020131
 6145  034310  044127  047105  051040
 6146  034316  040505  020104  047503
 6147  034324  046515  047101     104
 6148  034331     015  044412  051523          .ASCIZ  <15><12>@ISSUED TO SEEKING DRIVE@
 6149  034336  042525  020104  047524
 6150  034344  051440  042505  044513
 6151  034352  043516  042040  044522
 6152  034360  042526     000
 6153  034363     105  051122  051117  EM34:   .ASCIZ  @ERROR DURING READ COMMAND WHICH WAS ISSUED TO SEEKING DRIVE@
 6154  034370  042040  051125  047111
 6155  034376  020107  042522  042101
 6156  034404  041440  046517  040515
 6157  034412  042116  053440  044510
 6158  034420  044103  053440  051501
 6159  034426  044440  051523  042525
 6160  034434  020104  047524  051440
 6161  034442  042505  044513  043516
 6162  034450  042040  044522  042526
 6163  034456     000
 6164  034457     104  052101  020101  EM35:   .ASCIZ  @DATA INCORRECT FROM HEADER READ COMMAND ISSUED TO A SEEKING DRIVE@
 6165  034464  047111  047503  051122
 6166  034472  041505  020124  051106
 6167  034500  046517  044040  040505
 6168  034506  042504  020122  042522
 6169  034514  042101  041440  046517
```

```
 6170  034522  040515  042116  044440
 6171  034530  051523  042525  020104
 6172  034536  047524  040440  051440
 6173  034544  042505  044513  043516
 6174  034552  042040  044522  042526
 6175  034560     000
 6176  034561     047  054116  042515  EM36:   .ASCIZ  @'NXME' DIDN'T SET WHEN LOCATION 760000 ADDRESSED@
 6177  034566  020047  044504  047104
 6178  034574  052047  051440  052105
 6179  034602  053440  042510  020116
 6180  034610  047514  040503  044524
 6181  034616  047117  033440  030066
 6182  034624  030060  020060  042101
 6183  034632  051104  051505  042523
 6184  034640  000104
 6185  034642  044047  023505  042040  EM37:   .ASCIZ  @'HE' DIDN'T SET WITH 'NXME'@
 6186  034650  042111  023516  020124
 6187  034656  042523  020124  044527
 6188  034664  044124  023440  054116
 6189  034672  042515  000047
 6190  034676  042447  051122  020047  EM40:   .ASCIZ  @'ERR' DIDN?T SET WITH 'NXME'@
 6191  034704  044504  047104  052067
 6192  034712  051440  052105  053440
 6193  034720  052111  020110  047047
 6194  034726  046530  023505     000
 6195  034733     105  051122  051117  EM41:   .ASCIZ  @ERROR FORMATTING TRACK 0@
 6196  034740  043040  051117  040515
 6197  034746  052124  047111  020107
 6198  034754  051124  041501  020113
 6199  034762  000060
 6200  034764  051105  047522  020122  EM42:   .ASCIZ  @ERROR READING THE HEADER FROM THE TEST SECTOR@
 6201  034772  042522  042101  047111
 6202  035000  020107  044124  020105
 6203  035006  042510  042101  051105
 6204  035014  043040  047522  020115
 6205  035022  044124  020105  042524
 6206  035030  052123  051440  041505
 6207  035036  047524  000122
 6208  035042  042523  052103  051117  EM43:   .ASCIZ  @SECTOR FIELD FROM HEADER IS NOT CORRECT@
 6209  035050  043040  042511  042114
 6210  035056  043040  047522  020115
 6211  035064  042510  042101  051105
 6212  035072  044440  020123  047516
 6213  035100  020124  047503  051122
 6214  035106  041505  000124
 6215  035112  051105  047522  020122  EM44:   .ASCIZ  @ERROR WRITING SECTOR ADDRESS IN DATA FIELD@
 6216  035120  051127  052111  047111
 6217  035126  020107  042523  052103
 6218  035134  051117  040440  042104
 6219  035142  042522  051523  044440
 6220  035150  020116  040504  040524
 6221  035156  043040  042511  042114
 6222  035164     000
 6223  035165     047  047523  023524  EM45:   .ASCIZ  @'SOT' OR SECTOR ADDRESS REIGSTER IS NOT CORRECT@
 6224  035172  047440  020122  042523
 6225  035200  052103  051117  040440
```

```
 6226  035206  042104  042522  051523
 6227  035214  051040  044505  051507
 6228  035222  042524  020122  051511
 6229  035230  047040  052117  041440
 6230  035236  051117  042522  052103
 6231  035244     000
 6232  035245     104  052101  020101  EM46:   .ASCIZ  @DATA READ FROM THE SECTOR IS NOT CORRECT@
 6233  035252  042522  042101  043040
 6234  035260  047522  020115  044124
 6235  035266  020105  042523  052103
 6236  035274  051117  044440  020123
 6237  035302  047516  020124  047503
 6238  035310  051122  041505  000124
 6239  035316  042523  052103  051117  EM47:   .ASCII  @SECTOR CONTENTS WRONG, DATA SHOULD BE THE@<15><12>
 6240  035324  041440  047117  042524
 6241  035332  052116  020123  051127
 6242  035340  047117  026107  042040
 6243  035346  052101  020101  044123
 6244  035354  052517  042114  041040
 6245  035362  020105  044124  006505
 6246  035370     012
 6247  035371     124  040522  045503          .ASCIZ  @TRACK NUMBER OF THE CURRENTLY SELECTED HEAD@
 6248  035376  047040  046525  042502
 6249  035404  020122  043117  052040
 6250  035412  042510  041440  051125
 6251  035420  042522  052116  054514
 6252  035426  051440  046105  041505
 6253  035434  042524  020104  042510
 6254  035442  042101     000
 6255  035445     105  051122  051117  EM50:   .ASCIZ  @ERROR AFTER 2 WORD READ STARTING AT LOC 177776@
 6256  035452  040440  052106  051105
 6257  035460  031040  053440  051117
 6258  035466  020104  042522  042101
 6259  035474  051440  040524  052122
 6260  035502  047111  020107  052101
 6261  035510  046040  041517  030440
 6262  035516  033467  033467  000066
 6263  035524  046447  054105  023460  EM51:   .ASCII  @'MEX0' DIDN'T SET AFTER 2 WORD READ STARTING@<15><12>
 6264  035532  042040  042111  023516
 6265  035540  020124  042523  020124
 6266  035546  043101  042524  020122
 6267  035554  020062  047527  042122
 6268  035562  051040  040505  020104
 6269  035570  052123  051101  044524
 6270  035576  043516  005015
 6271  035602  052101  046040  041517          .ASCIZ  @AT LOC 177776@
 6272  035610  030440  033467  033467
 6273  035616  000066
 6274  035620  046447  054105  023461  EM52:   .ASCIZ  @'MEX1' SET AFTER 2 WORD READ STARTING AT LOC 177776@
 6275  035626  051440  052105  040440
 6276  035634  052106  051105  031040
 6277  035642  053440  051117  020104
 6278  035650  042522  042101  051440
 6279  035656  040524  052122  047111
 6280  035664  020107  052101  046040
 6281  035672  041517  030440  033467
```

```
 6282  035700  033467  000066
 6283  035704  047503  052116  047105  EM53:   .ASCII  @CONTENTS OF LOC 200000 WRONG AFTER 2 WORD READ@<15><12>
 6284  035712  051524  047440  020106
 6285  035720  047514  020103  030062
 6286  035726  030060  030060  053440
 6287  035734  047522  043516  040440
 6288  035742  052106  051105  031040
 6289  035750  053440  051117  020104
 6290  035756  042522  042101  005015
 6291  035764  052123  051101  044524          .ASCIZ  @STARTING AT LOC 177776@
 6292  035772  043516  040440  020124
 6293  036000  047514  020103  033461
 6294  036006  033467  033067     000
 6295  036013     105  051122  051117  EM54:   .ASCIZ  @ERROR AFTER 2 WORD READ STARTING AT LOC 377776@
 6296  036020  040440  052106  051105
 6297  036026  031040  053440  051117
 6298  036034  020104  042522  042101
 6299  036042  051440  040524  052122
 6300  036050  047111  020107  052101
 6301  036056  046040  041517  031440
 6302  036064  033467  033467  000066
 6303  036072  046447  054105  023460  EM55:   .ASCIZ  @'MEX0' DIDN'T CLEAR AFTER 2 WORD STARTING AT LOC 377776@
 6304  036100  042040  042111  023516
 6305  036106  020124  046103  040505
 6306  036114  020122  043101  042524
 6307  036122  020122  020062  047527
 6308  036130  042122  051440  040524
 6309  036136  052122  047111  020107
 6310  036144  052101  046040  041517
 6311  036152  031440  033467  033467
 6312  036160  000066
 6313  036162  046447  054105  023461  EM56:   .ASCIZ  @'MEX1' DIDN'T SET AFTER 2 WORD READ STARTING AT LOC 377776@
 6314  036170  042040  042111  023516
 6315  036176  020124  042523  020124
 6316  036204  043101  042524  020122
 6317  036212  020062  047527  042122
 6318  036220  051040  040505  020104
 6319  036226  052123  051101  044524
 6320  036234  043516  040440  020124
 6321  036242  047514  020103  033463
 6322  036250  033467  033067     000
 6323  036255     103  047117  042524  EM57:   .ASCII  @CONTENTS OF LOC 400000 WRONG AFTER 2 WORD READ@<15><12>
 6324  036262  052116  020123  043117
 6325  036270  046040  041517  032040
 6326  036276  030060  030060  020060
 6327  036304  051127  047117  020107
 6328  036312  043101  042524  020122
 6329  036320  020062  047527  042122
 6330  036326  051040  040505  006504
 6331  036334     012
 6332  036335     123  040524  052122          .ASCIZ  @STARTING AT LOCATION 377776@
 6333  036342  047111  020107  052101
 6334  036350  046040  041517  052101
 6335  036356  047511  020116  033463
 6336  036364  033467  033067     000
 6337  036371     105  051122  051117  EM60:   .ASCII  @ERROR RETURNING SECTOR TO PDP-11 MODE USING A@<15><12>
```

```
 6338  036376  051040  052105  051125
 6339  036404  044516  043516  051440
 6340  036412  041505  047524  020122
 6341  036420  047524  050040  050104
 6342  036426  030455  020061  047515
 6343  036434  042504  052440  044523
 6344  036442  043516  040440  005015
 6345  036450  020062  047527  042122          .ASCIZ  @2 WORD WRITE@
 6346  036456  053440  044522  042524
 6347  036464     000
 6348  036465     105  051122  051117  EM61:   .ASCIZ  @ERROR WRITING TEST SECTOR(S)@
 6349  036472  053440  044522  044524
 6350  036500  043516  052040  051505
 6351  036506  020124  042523  052103
 6352  036514  051117  051450  000051
 6353  036522  051105  047522  020122  EM62:   .ASCIZ  @ERROR READING TEST SECTOR(S)@
 6354  036530  042522  042101  047111
 6355  036536  020107  042524  052123
 6356  036544  051440  041505  047524
 6357  036552  024122  024523     000
 6358  036557     104  044522  042526  EM63:   .ASCII  @DRIVE DIDN'T RETURN TO CYL 0 FROM THE MAXIMUM@<15><12>
 6359  036564  042040  042111  023516
 6360  036572  020124  042522  052524
 6361  036600  047122  052040  020117
 6362  036606  054503  020114  020060
 6363  036614  051106  046517  052040
 6364  036622  042510  046440  054101
 6365  036630  046511  046525  005015
 6366  036636  054503  020114  047117          .ASCIZ  @CYL ON CONTROLLER POWER FAIL@
 6367  036644  041440  047117  051124
 6368  036652  046117  042514  020122
 6369  036660  047520  042527  020122
 6370  036666  040506  046111     000
 6371  036673     103  047117  042524  EM64:   .ASCII  @CONTENTS OF MEMORY CHANGED DURING POWER@<15><12>
 6372  036700  052116  020123  043117
 6373  036706  046440  046505  051117
 6374  036714  020131  044103  047101
 6375  036722  042507  020104  052504
 6376  036730  044522  043516  050040
 6377  036736  053517  051105  005015
 6378  036744  040506  046111  053440          .ASCIZ  @FAIL WHILE READING THE DISK@
 6379  036752  044510  042514  051040
 6380  036760  040505  044504  043516
 6381  036766  052040  042510  042040
 6382  036774  051511  000113
 6383  037000  051447  047525  023514  EM65:   .ASCIZ  @'SUOL' SET WITH DRIVE DISABLED@
 6384  037006  051440  052105  053440
 6385  037014  052111  020110  051104
 6386  037022  053111  020105  044504
 6387  037030  040523  046102  042105
 6388  037036     000
 6389  037037     047  052523  04611/  EM66:   .ASCIZ  @'SUOL' NOT SET WITH DRIVE ENABLED@
 6390  037044  020047  047516  020124
 6391  037052  042523  020124  044527
 6392  037060  044124  042040  044522
 6393  037066  042526  042440  040516
```

```
 6394  037074  046102  042105     000
 6395  037101     047  052523  050127  EM67:   .ASCIZ  @'SUWP' NOT SET WITH DRIVE SET TO 'READ ONLY'@
 6396  037106  020047  047516  020124
 6397  037114  042523  020124  044527
 6398  037122  044124  042040  044522
 6399  037130  042526  051440  052105
 6400  037136  052040  020117  051047
 6401  037144  040505  020104  047117
 6402  037152  054514  000047
 6403  037156  051447  053525  023520  EM70:   .ASCIZ  @'SUWP' SET WITH DRIVE SET TO 'READ/WRITE'@
 6404  037164  051440  052105  053440
 6405  037172  052111  020110  051104
 6406  037200  053111  020105  042523
 6407  037206  020124  047524  023440
 6408  037214  042522  042101  053457
 6409  037222  044522  042524  000047
 6410  037230  051447  053525  023520  EM71:   .ASCII  @'SUWP' SET WITH RP11 'WRITE LOCKOUT' SET, CYLINDER@<15><12>
 6411  037236  051440  052105  053440
 6412  037244  052111  020110  050122
 6413  037252  030461  023440  051127
 6414  037260  052111  020105  047514
 6415  037266  045503  052517  023524
 6416  037274  051440  052105  020054
 6417  037302  054503  044514  042116
 6418  037310  051105  005015
 6419  037314  047514  023501  020123          .ASCIZ  @LOA'S = 0, AND RPCA = 0@
 6420  037322  020075  026060  040440
 6421  037330  042116  051040  041520
 6422  037336  020101  020075  000060
 6423  037344  051447  053525  023520  EM72:   .ASCIZ  @'SUWP' SET WITH RPCA = 2 & CYL LOA'S = 0@
 6424  037352  051440  052105  053440
 6425  037360  052111  020110  050122
 6426  037366  040503  036440  031040
 6427  037374  023040  041440  046131
 6428  037402  046040  040517  051447
 6429  037410  036440  030040     000
 6430  037415     047  052523  050127  EM73:   .ASCIZ  @'SUWP' NOT SET; RPCA = VALUE IN CYL LOA'S@
 6431  037422  020047  047516  020124
 6432  037430  042523  035524  051040
 6433  037436  041520  020101  020075
 6434  037444  040526  052514  020105
 6435  037452  047111  041440  046131
 6436  037460  046040  040517  051447
 6437  037466     000
 6438  037467     047  052523  050127  EM74:   .ASCIZ  @'SUWP' SET WITH RPCA ONE GREATER THAN CYL LOA VALUE@
 6439  037474  020047  042523  020124
 6440  037502  044527  044124  051040
 6441  037510  041520  020101  047117
 6442  037516  020105  051107  040505
 6443  037524  042524  020122  044124
 6444  037532  047101  041440  046131
 6445  037540  046040  040517  053040
 6446  037546  046101  042525     000
 6447  037553     047  052523  050127  EM75:   .ASCIZ  @'SUWP' SET WITH DRIVE LOA SWITCHES EQUAL TO SELECTED DRIVE@
 6448  037560  020047  042523  020124
 6449  037566  044527  044124  042040
```

```
 6450  037574  044522  042526  046040
 6451  037602  040517  051440  044527
 6452  037610  041524  042510  020123
 6453  037616  050505  040525  020114
 6454  037624  047524  051440  046105
 6455  037632  041505  042524  020104
 6456  037640  051104  053111  000105
 6457  037646  051104  053111  020105  EM76:   .ASCIZ  @DRIVE HAS GONE OFFLINE@
 6458  037654  040510  020123  047507
 6459  037662  042516  047440  043106
 6460  037670  044514  042516     000
 6461  037675     104  044522  042526  EM77:   .ASCIZ  @DRIVE IS UNSAFE@
 6462  037702  044440  020123  047125
 6463  037710  040523  042506     000
 6464  037715     104  044522  042526  EM100:  .ASCIZ  @DRIVE TIMED OUT@
 6465  037722  052040  046511  042105
 6466  037730  047440  052125     000
 6467  037735     103  047117  051124  EM101:  .ASCIZ  @CONTROLLER TIMED OUT@
 6468  037742  046117  042514  020122
 6469  037750  044524  042515  020104
 6470  037756  052517  000124
 6471  037762  040504  040524  041440  EM102:  .ASCIZ  @DATA COMPARE ERROR@
 6472  037770  046517  040520  042522
 6473  037776  042440  051122  051117
 6474  040004     000
 6475  040005     104  052101  020101  EM103:  .ASCIZ  @DATA COMPARE ERROR (MEMORY MANAGEMENT ENABLED)@
 6476  040012  047503  050115  051101
 6477  040020  020105  051105  047522
 6478  040026  020122  046450  046505
 6479  040034  051117  020131  040515
 6480  040042  040516  042507  042515
 6481  040050  052116  042440  040516
 6482  040056  046102  042105  000051
 6483  040064  042447  051122  020047  EM106:  .ASCIZ  @'ERR' DIDN'T SET WITH 'WCE'@
 6484  040072  044504  047104  052047
 6485  040100  051440  052105  053440
 6486  040106  052111  020110  053447
 6487  040114  042503  000047
 6488  040120  044047  023505  042040  EM107:  .ASCIZ  @'HE' DIDN'T SET WITH 'PROG'@
 6489  040126  042111  023516  020124
 6490  040134  042523  020124  044527
 6491  040142  044124  023440  051120
 6492  040150  043517  000047
 6493  040154  051104  053111  020105  EM112:  .ASCIZ  @DRIVE UNSAFE AFTER POWER FAIL@
 6494  040162  047125  040523  042506
 6495  040170  040440  052106  051105
 6496  040176  050040  053517  051105
 6497  040204  043040  044501  000114
 6498  040212  051447  053525  023520  EM113:  .ASCIZ  @'SUWP' NOT SET WITH DRIVE LOA SWITCHES GREATER THAN SELECTED DRIVE@
 6499  040220  047040  052117  051440
 6500  040226  052105  053440  052111
 6501  040234  020110  051104  053111
 6502  040242  020105  047514  020101
 6503  040250  053523  052111  044103
 6504  040256  051505  043440  042522
 6505  040264  052101  051105  052040
```

```
 6506  040272  040510  020116  042523
 6507  040300  042514  052103  042105
 6508  040306  042040  044522  042526
 6509  040314     000
 6510
 6511                          ;'DH' MESSAGE LINES
 6512
 6513  040315     122  040520  051104  DH1:    .ASCIZ  @RPADR@
 6514  040322     000
 6515  040323     124  051505  020124  DH2:    .ASCIZ  @TEST #  ERR PC  DRV #   RPDS    RPER    RPCS    RPWC@
 6516  040330  020043  042440  051122
 6517  040336  050040  020103  042040
 6518  040344  053122  021440  020040
 6519  040352  051040  042120  020123
 6520  040360  020040  051040  042520
 6521  040366  020122  020040  051040
 6522  040374  041520  020123  020040
 6523  040402  051040  053520  000103
 6524  040410  050122  040502  020040  DH2A:   .ASCIZ  @RPBA    RPCA    RPDA    RPM1    SUCA    SILO@
 6525  040416  020040  050122  040503
 6526  040424  020040  020040  050122
 6527  040432  040504  020040  020040
 6528  040440  050122  030515  020040
 6529  040446  020040  052523  040503
 6530  040454  020040  020040  044523
 6531  040462  047514     000
 6532  040465     124  051505  020124  DH5:    .ASCIZ  @TEST #  ERR PC  DRV #   PATRN # RPDS    RPER    RPCS@
 6533  040472  020043  042440  051122
 6534  040500  050040  020103  042040
 6535  040506  053122  021440  020040
 6536  040514  050040  052101  047122
 6537  040522  021440  051040  042120
 6538  040530  020123  020040  051040
 6539  040536  042520  020122  020040
 6540  040544  051040  041520  000123
 6541  040552  050122  041527  020040  DH5A:   .ASCIZ  @RPWC    RPBA    RPCA    RPDA    RPM1    SUCA    SILO@
 6542  040560  020040  050122  040502
 6543  040566  020040  020040  050122
 6544  040574  040503  020040  020040
 6545  040602  050122  040504  020040
 6546  040610  020040  050122  030515
 6547  040616  020040  020040  052523
 6548  040624  040503  020040  020040
 6549  040632  044523  047514     000
 6550  040637     124  051505  020124  DH11:   .ASCIZ  @TEST #  ERR PC  DRV #   EXPT'D 1ST WORD 2ND WORD@
 6551  040644  020043  042440  051122
 6552  040652  050040  020103  042040
 6553  040660  053122  021440  020040
 6554  040666  042440  050130  023524
 6555  040674  020104  051461  020124
 6556  040702  047527  042122  031040
 6557  040710  042116  053440  051117
 6558  040716  000104
 6559  040720  042524  052123  021440  DH12:   .ASCIZ  @TEST #  ERR PC  DRV #   EXPT'D  RECV'D  LOCN@
 6560  040726  020040  051105  020122
 6561  040734  041520  020040  051104
```

```
 6562  040742  020126  020043  020040
 6563  040750  054105  052120  042047
 6564  040756  020040  042522  053103
 6565  040764  042047  020040  047514
 6566  040772  047103     000
 6567  040775     124  051505  020124  DH14:   .ASCIZ  @TEST #  ERR PC  DRV #   RPDS    RPER    RPCS@
 6568  041002  020043  042440  051122
 6569  041010  050040  020103  042040
 6570  041016  053122  021440  020040
 6571  041024  051040  042120  020123
 6572  041032  020040  051040  042520
 6573  041040  020122  020040  051040
 6574  041046  041520  000123
 6575  041052  020040  020040  020040  DH16:   .ASCII  @                       EXPT'D  ACTUAL@<15><12>
 6576  041060  020040  020040  020040
 6577  041066  020040  020040  020040
 6578  041074  020040  020040  020040
 6579  041102  054105  052120  042047
 6580  041110  020040  041501  052524
 6581  041116  046101  005015
 6582  041122  042524  052123  021440          .ASCIZ  @TEST #  ERR PC  DRV #   SUCA    SUCA@
 6583  041130  020040  051105  020122
 6584  041136  041520  020040  051104
 6585  041144  020126  020043  020040
 6586  041152  052523  040503  020040
 6587  041160  020040  052523  040503
 6588  041166     000
 6589  041167     124  051505  020124  DH25:   .ASCIZ  @TEST #  ERR PC  DRV #   EXPT'D  RECV'D@
 6590  041174  020043  042440  051122
 6591  041202  050040  020103  042040
 6592  041210  053122  021440  020040
 6593  041216  042440  050130  023524
 6594  041224  020104  051040  041505
 6595  041232  023526  000104
 6596  041236  042510  042101  051105  DH25A:  .ASCIZ  @HEADER CONTENTS@
 6597  041244  041440  047117  042524
 6598  041252  052116  000123
 6599  041256  020040  020040  020040  DH43:   .ASCII  @                       EXPT'D  ACTUAL@<15><12>
 6600  041264  020040  020040  020040
 6601  041272  020040  020040  020040
 6602  041300  020040  020040  020040
 6603  041306  054105  052120  042047
 6604  041314  020040  041501  052524
 6605  041322  046101  005015
 6606  041326  042524  052123  021440          .ASCIZ  @TEST #  ERR PC  DRV #   SECTOR  SECTOR@
 6607  041334  020040  051105  020122
 6608  041342  041520  020040  051104
 6609  041350  020126  020043  020040
 6610  041356  042523  052103  051117
 6611  041364  020040  042523  052103
 6612  041372  051117     000
 6613  041375     122  042120  020123  DH43A:  .ASCIZ  @RPDS    RPER    RPCS    RPWC    RPBA    RPCA    RPDA@
 6614  041402  020040  051040  042520
 6615  041410  020122  020040  051040
 6616  041416  041520  020123  020040
 6617  041424  051040  053520  020103
```

```
 6618  041432  020040  051040  041120
 6619  041440  020101  020040  051040
 6620  041446  041520  020101  020040
 6621  041454  051040  042120  000101
 6622  041462  042510  042101  051105  DH43B:  .ASCIZ  @HEADER CONTENTS@
 6623  041470  041440  047117  042524
 6624  041476  052116  000123
 6625  041502  020040  020040  020040  DH45:   .ASCII  @                       EXPT'D  'SOT'   SECTOR@<15><12>
 6626  041510  020040  020040  020040
 6627  041516  020040  020040  020040
 6628  041524  020040  020040  020040
 6629  041532  054105  052120  042047
 6630  041540  020040  051447  052117
 6631  041546  020047  020040  042523
 6632  041554  052103  051117  005015
 6633  041562  042524  052123  021440          .ASCIZ  @TEST #  ERR PC  DRV #   VALUE  COUNTER  ADDR@
 6634  041570  020040  051105  020122
 6635  041576  041520  020040  051104
 6636  041604  020126  020043  020040
 6637  041612  040526  052514  020105
 6638  041620  041440  052517  052116
 6639  041626  051105  020040  042101
 6640  041634  051104     000
 6641  041637     122  042120  020123  DH45A:  .ASCIZ  @RPDS    RPER    RPCS    RPDA@
 6642  041644  020040  051040  042520
 6643  041652  020122  020040  051040
 6644  041660  041520  020123  020040
 6645  041666  051040  042120  000101
 6646  041674  020040  020040  020040  DH46:   .ASCII  @                       EXPT'D  RECV'D@<15><12>
 6647  041702  020040  020040  020040
 6648  041710  020040  020040  020040
 6649  041716  020040  020040  020040
 6650  041724  020040  020040  020040
 6651  041732  020040  054105  052120
 6652  041740  042047  020040  042522
 6653  041746  053103  042047  005015
 6654  041754  042524  052123  021440          .ASCIZ  @TEST #  ERR PC  DRV #   SECTOR  DATA    DATA@
 6655  041762  020040  051105  020122
 6656  041770  041520  020040  051104
 6657  041776  020126  020043  020040
 6658  042004  042523  052103  051117
 6659  042012  020040  040504  040524
 6660  042020  020040  020040  040504
 6661  042026  040524     000
 6662  042031     124  051505  020124  DH47:   .ASCIZ  @TEST #  ERR PC  DRV #   EXPT'D  RECV'D@
 6663  042036  020043  042440  051122
 6664  042044  050040  020103  042040
 6665  042052  053122  021440  020040
 6666  042060  042440  050130  023524
 6667  042066  020104  051040  041505
 6668  042074  023526  000104
 6669  042100  050122  051504  020040  DH47A:  .ASCIZ  @RPDS    RPER    RPCS    RPCA    RPDA@
 6670  042106  020040  050122  051105
 6671  042114  020040  020040  050122
 6672  042122  051503  020040  020040
 6673  042130  050122  040503  020040
```

```
 6674  042136  020040  050122  040504
 6675  042144     000
 6676  042145     124  051505  020124  DH51:   .ASCIZ  @TEST #  ERR PC  DRV #   RPDS    RPER    RPCS    RPWC    RPBA@
 6677  042152  020043  042440  051122
 6678  042160  050040  020103  042040
 6679  042166  053122  021440  020040
 6680  042174  051040  042120  020123
 6681  042202  020040  051040  042520
 6682  042210  020122  020040  051040
 6683  042216  041520  020123  020040
 6684  042224  051040  053520  020103
 6685  042232  020040  051040  041120
 6686  042240  000101
 6687  042242  042524  052123  021440  DH63:   .ASCIZ  @TEST #  ERR PC  DRV #   RPDS    RPER    RPCS    SUCA@
 6688  042250  020040  051105  020122
 6689  042256  041520  020040  051104
 6690  042264  020126  020043  020040
 6691  042272  050122  051504  020040
 6692  042300  020040  050122  051105
 6693  042306  020040  020040  050122
 6694  042314  051503  020040  020040
 6695  042322  052523  040503     000
 6696  042327     124  051505  020124  DH71:   .ASCIZ  @TEST #  ERR PC  DRV #   RPDS    RPER    RPCS    RPCA@
 6697  042334  020043  042440  051122
 6698  042342  050040  020103  042040
 6699  042350  053122  021440  020040
 6700  042356  051040  042120  020123
 6701  042364  020040  051040  042520
 6702  042372  020122  020040  051040
 6703  042400  041520  020123  020040
 6704  042406  051040  041520  000101
 6705  042414  042524  052123  021440  DH102:  .ASCIZ  @TEST #  ERR PC  DRV #  SEC STRT CYL     TRK     SEC     PATTERN@
 6706  042422  020040  051105  020122
 6707  042430  041520  020040  051104
 6708  042436  020126  020043  051440
 6709  042444  041505  051440  051124
 6710  042452  020124  054503  020114
 6711  042460  020040  020040  051124
 6712  042466  020113  020040  020040
 6713  042474  042523  020103  020040
 6714  042502  020040  040520  052124
 6715  042510  051105  000116
 6716  042514  020040  020040  020040  DH102A: .ASCII  @        GOOD    BAD@<15><12>
 6717  042522  020040  047507  042117
 6718  042530  020040  020040  040502
 6719  042536  006504     012
 6720  042541     101  042104  020122          .ASCIZ  @ADDR    DATA    DATA@
 6721  042546  020040  042040  052101
 6722  042554  020101  020040  042040
 6723  042562  052101  000101
 6724  042566  044513  040520  031522  DH103:  .ASCIZ  @KIPAR3  KIPAR4  KIPAR5  KIPDR3  KIPDR4  KIPDR5@
 6725  042574  020040  044513  040520
 6726  042602  032122  020040  044513
 6727  042610  040520  032522  020040
 6728  042616  044513  042120  031522
 6729  042624  020040  044513  042120
```

```
 6730  042632  032122  020040  044513
 6731  042640  042120  032522     000
 6732  042645     126  051111  052524  DH103A: .ASCII  @VIRTUAL GOOD    BAD@<15><12>
 6733  042652  046101  043440  047517
 6734  042660  020104  020040  041040
 6735  042666  042101  005015
 6736  042672  042101  051104  020040          .ASCIZ  @ADDR    DATA    DATA@
 6737  042700  020040  040504  040524
 6738  042706  020040  020040  040504
 6739  042714  040524     000
 6740  042717     124  052117  046101  DH105:  .ASCIZ  @TOTAL COMPARE ERRORS:@
 6741  042724  041440  046517  040520
 6742  042732  042522  042440  051122
 6743  042740  051117  035123     000
 6744  042745     125  051516  042503  DH110:  .ASCIZ  @UNSCESSFUL AFTER 3 RETRIES@
 6745  042752  051523  052506  020114
 6746  042760  043101  042524  020122
 6747  042766  020063  042522  051124
 6748  042774  042511  000123
 6749  043000  042522  051124  020131  DH111:  .ASCII  @RETRY SUCESSFUL@<15><12>
 6750  043006  052523  042503  051523
 6751  043014  052506  006514     012
 6752  043021     122  052105  054522          .ASCIZ  @RETRY COUNT:@
 6753  043026  041440  052517  052116
 6754  043034  000072
 6755
 6756                                  .EVEN
 6757
 6758                                  ;'DT' WORDS
 6759
 6760  043036  001324                  DT1:    .WORD   RPADR
 6761  043040  001160  001116  001176  DT2:    .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$RPWC
 6762  043046  001334  001336  001340
 6763  043054  001342
 6764  043056  001344  001346  001350          .WORD   $RPBA,$RPCA,$RPDA,$RPM1,$SUCA,$SILO
 6765  043064  001352  001354  001356
 6766  043072  001160  001116  001176  DT5:    .WORD   $TMPO,$ERRPC,CHKDRV,PATNUM,$RPDS,$RPER,$RPCS
 6767  043100  001204  001334  001336
 6768  043106  001340
 6769  043110  001342  001344  001346          .WORD   $RPWC,$RPBA,$RPCA,$RPDA,$RPM1,$SUCA,$SILO
 6770  043116  001350  001352  001354
 6771  043124  001356
 6772  043126  001160  001116  001176  DT11:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,BUFFER,BUFFER+2
 6773  043134  001124  043656  043660
 6774  043142  001160  001116  001176  DT12:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,$BDDAT,$BDADR
 6775  043150  001124  001126  001122
 6776  043156  001160  001116  001176  DT14:   .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS
 6777  043164  001334  001336  001340
 6778  043172  001160  001116  001176  DT16:   .WORD   $TMPO,$ERRPC,CHKDRV,MAXCYL,$SUCA
 6779  043200  001210  001354
 6780  043204  001160  001116  001176  DT25:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,$BDDAT
 6781  043212  001124  001126
 6782  043216  043656  043660  043662          .WORD   BUFFER,BUFFER+2,BUFFER+4
 6783  043224  001160  001116  001176  DT43:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,BUFFER+4
 6784  043232  001124  043662
 6785  043236  001334  001336  001340          .WORD   $RPDS,$RPER,$RPCS,$RPWC,$RPBA,$RPCA,$RPDA
```

```
 6786  043244  001342  001344  001346
 6787  043252  001350
 6788  043254  043656  043660  043662          .WORD   BUFFER,BUFFER+2,BUFFER+4
 6789  043262  001160  001116  001176  DT45:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,$BDDAT,$BDADR
 6790  043270  001124  001126  001122
 6791  043276  001334  001336  001340          .WORD   $RPDS,$RPER,$RPCS,$RPDA
 6792  043304  001350
 6793  043306  001160  001116  001176  DT46:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDADR,$GDDAT,$BDDAT
 6794  043314  001120  001124  001126
 6795  043322  001160  001116  001176  DT47:   .WORD   $TMPO,$ERRPC,CHKDRV,$GDDAT,BUFFER
 6796  043330  001124  043656
 6797  043334  001334  001336  001340          .WORD   $RPDS,$RPER,$RPCS,$RPCA,$RPDA
 6798  043342  001346  001350
 6799  043346  001160  001116  001176  DT51:   .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$RPWC,$RPBA
 6800  043354  001334  001336  001340
 6801  043362  001342  001344
 6802  043366  001160  001116  001176  DT63:   .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$SUCA
 6803  043374  001334  001336  001340
 6804  043402  001354
 6805  043404  001160  001116  001176  DT71:   .WORD   $TMPO,$ERRPC,CHKDRV,$RPDS,$RPER,$RPCS,$RPCA
 6806  043412  001334  001336  001340
 6807  043420  001346
 6808  043422  001160  001116  001176  DT102:  .WORD   $TMPO,$ERRPC,CHKDRV,CMBUF,CMCYL,CMTRK,CMSEC,PATNUM
 6809  043430  025430  025422  025426
 6810  043436  025424  001204
 6811  043442  001122  001124  001126          .WORD   $BDADR,$GDDAT,$BDDAT
 6812  043450  001160  001116  001176  DT103:  .WORD   $TMPO,$ERRPC,CHKDRV,CMBUF,CMCYL,CMTRK,CMSEC,PATNUM
 6813  043456  025430  025422  025426
 6814  043464  025424  001204
 6815  043470  172346  172350  172352          .WORD   KIPAR3,KIPAR4,KIPAR5,KIPDR3,KIPDR4,KIPDR5
 6816  043476  172306  172310  172312
 6817  043504  001122  001124  001126  DT104:  .WORD   $BDADR,$GDDAT,$BDDAT
 6818  043512  025414                  DT105:  .WORD   ERCTR
 6819  043514  001314                  DT111:  .WORD   RETRY
 6820
 6821                                  ;'DF' BLOCKS
 6822
 6823  043516  000001                  DF1:    .WORD   1
 6824  043520     001                          .BYTE   1
 6825  043521     000                          .BYTE   0
 6826
 6827  043522  000002                  DF2:    .WORD   2
 6828  043524     007                          .BYTE   7
 6829  043525     000                          .BYTE   0
 6830  043526  040410                          .WORD   DH2A
 6831  043530     006                          .BYTE   6
 6832  043531     000                          .BYTE   0
 6833
 6834  043532  000002                  DF5:    .WORD   2
 6835  043534     007                          .BYTE   7
 6836  043535     010                          .BYTE   10
 6837  043536  040552                          .WORD   DH5A
 6838  043540     007                          .BYTE   7
 6839  043541     000                          .BYTE   0
 6840
 6841  043542  000001                  DF11:   .WORD   1
```

```
 6842  043544  006                     .BYTE   6
 6843  043545  000                     .BYTE   0
 6844
 6845  043546  000002          DF25:   .WORD   2
 6846  043550  005                     .BYTE   5
 6847  043551  000                     .BYTE   0
 6848  043552  041236                  .WORD   DH25A
 6849  043554  003                     .BYTE   3
 6850  043555  000                     .BYTE   0
 6851
 6852  043556  000003          DF43:   .WORD   3
 6853  043560  006                     .BYTE   6
 6854  043561  000                     .BYTE   0
 6855  043562  041375                  .WORD   DH43A
 6856  043564  005                     .BYTE   5
 6857  043565  000                     .BYTE   0
 6858  043566  041462                  .WORD   DH43B
 6859  043570  003                     .BYTE   3
 6860  043571  000                     .BYTE   0
 6861
 6862  043572  000002          DF45:   .WORD   2
 6863  043574  006                     .BYTE   6
 6864  043575  000                     .BYTE   0
 6865  043576  041637                  .WORD   DH45A
 6866  043600  004                     .BYTE   4
 6867  043601  000                     .BYTE   0
 6868
 6869  043602  000002          DF47:   .WORD   2
 6870  043604  005                     .BYTE   5
 6871  043605  000                     .BYTE   0
 6872  043606  042100                  .WORD   DH47A
 6873  043610  005                     .BYTE   5
 6874  043611  000                     .BYTE   0
 6875
 6876  043612  000001          DF51:   .WORD   1
 6877  043614  010                     .BYTE   8.
 6878  043615  000                     .BYTE   0
 6879
 6880  043616  000001          DF53:   .WORD   1
 6881  043620  005                     .BYTE   5
 6882  043621  000                     .BYTE   0
 6883
 6884  043622  000001          DF63:   .WORD   1
 6885  043624  007                     .BYTE   7
 6886  043625  000                     .BYTE   0
 6887
 6888  043626  000002          DF102:  .WORD   2
 6889  043630  010                     .BYTE   8.
 6890  043631  200                     .BYTE   200
 6891  043632  042514                  .WORD   DH102A
 6892  043634  000                     .BYTE   0
 6893  043635  000                     .BYTE   0
 6894
 6895  043636  000003          DF103:  .WORD   3
 6896  043640  010                     .BYTE   8.
 6897  043641  200                     .BYTE   200
```

```
 6898  043642  042566                          .WORD   DH103
 6899  043644     006                          .BYTE   6
 6900  043645     000                          .BYTE   0
 6901  043646  042645                          .WORD   DH103A
 6902  043650     000                          .BYTE   0
 6903  043651     000                          .BYTE   0
 6904
 6905  043652  000001                DF104:    .WORD   1
 6906  043654     003                          .BYTE   3
 6907  043655     000                          .BYTE   0
 6908
 6909
 6910                                .EVEN
 6911
 6912          043656                BUFFER=.                ;BUFFER STARTS HERE
 6913
 6914  043656  005015  041412  051132  TITLE:  .ASCII  <15><12><12>@CZRPY-D@
 6915  043664  054520  042055
 6916  043670  005015  050122  030461          .ASCIZ  <15><12>@RP11-E FUNCTIONAL LOGIC & READ/WRITE TEST@<15><12><12>
 6917  043676  042455  043040  047125
 6918  043704  052103  047511  040516
 6919  043712  020114  047514  044507
 6920  043720  020103  020046  042522
 6921  043726  042101  053457  044522
 6922  043734  042524  052040  051505
 6923  043742  006524  005012     000
 6924          000001                          .END
```

```
ACTMEM  001224          1240#  3497*  3498*  5190   5191   5200*  5201*  5202   5205   5207*  5208*
AIE   = 020000          1178#
ATABIT  001360          1293#  2313   2334   3750   4879   4921   5068   5494   5512
BITS    001400          1319#  2404   2451   2530   2619   2703   2753   2848   2909   2976   3094   3170   3300
                        3469   3614   3721   3773   5084   5473
BIT0  = 000001          1083#
BIT00 = 000001          1073#  1083   1319   2814
BIT01 = 000002          1072#  1082   1320   2660
BIT02 = 000004          1071#  1081   1321   2933
BIT03 = 000010          1070#  1080   1322   2505   3934
BIT04 = 000020          1069#  1079   1323
BIT05 = 000040          1068#  1078   1324
BIT06 = 000100          1067#  1077   1325
BIT07 = 000200          1066#  1076   1326   5379
BIT08 = 000400          1065#  1075   1327
BIT09 = 001000          1064#  1074   1328   3905   4052   4645
BIT1  = 000002          1082#
BIT10 = 002000          1063#  1329   2728   4036
BIT11 = 004000          1062#  1330   4652
BIT12 = 010000          1061#  1331   2810
BIT13 = 020000          1060#  1332   4043
BIT14 = 040000          1059#  1333   4627
BIT15 = 100000          1058#  1334
BIT2  = 000004          1081#
BIT3  = 000010          1080#
BIT4  = 000020          1079#
BIT5  = 000040          1078#
BIT6  = 000100          1077#
BIT7  = 000200          1076#
BIT8  = 000400          1075#
BIT9  = 001000          1074#
BLKS14  001254          1247#  3499*  3500*  3501*  3502*  3503   3523   3563   5098   5105
BLNKS2  027430          4113   4124   4131   5689#
BPTVEC= 000014          1090#
BUFFER= 043656          2419   2470   2495*  2546   2583   2585   2588   2590   2636   2686   2720   2769*  2770*
                        2771*  2795   2801   2825*  2826*  2827*  2874   2890   2894   2992   3024   3074   3110
                        3137   3192*  3193*  3194   3314   3331   3343   3351   3360   3364   3395*  3396*  3397
                        3495   4968   4971   4995   4998   5159   5411   6772   6782   6783   6788   6795   6912#
BUSADR  001216          1237#  2203*  2205*  2208*  2210*  5427   5429*
BYPASS  001200          1230#  2281*  2412*  2459*  2538*  2627*  2711*  2761*  2856*  2917*  2984*  3102*  3178*
                        3308*  3477*  3622*  3729*  3781*  5031   5057
CHKDRV  001176          1229#  2368*  2372*  2378   2380   3663   3670   3792   3800   3807   3815   3825   3833
                        3840   3848   3910*  3915   3917   3929   3931*  3933*  3934   3937   3949   3974   6761
                        6766   6772   6774   6776   6778   6780   6783   6789   6793   6795   6799   6802   6805
                        6808   6812
CHNGAD  031640          5549   5903#
CHNGPM  031406          5518   5875#
CHNGPT  031565          5541   5895#
CIR     023120          4888   4910#  4920
CKSWR = 104407          4032   4051   4626   4834#
CK.CHR  027122          5615#  5636   5652
CK.NUM  027136          3743   5440   5466   5488   5532   5544   5558   5568   5578   5632#
CK.OCT  027072          5597#  5634   5647
CLEAR = 000001          1151#  2430   2431   4854   4855   5370   5374
CLRP    025654          2387   2418   2472   2481   2489   2501   2545   2561   2573   2634   2650   2674   2717
                        2768   2785   2805   2824   2862   2923   2991   3008   3045   3064   3113   3128   3191
```

```
DH25A   041236          6596#   6848
DH43    041256          1901    6599#
DH43A   041375          6613#   6855
DH43B   041462          6622#   6858
DH45    041502          1915    6625#
DH45A   041637          6641#   6865
DH46    041674          1922    6646#
DH47    042031          1930    6662#
DH47A   042100          6669#   6872
DH5     040465          1685    1692    2005    2012    6532#
DH5A    040552          6541#   6837
DH51    042145          1945    1952    1975    1982    6676#
DH63    042242          2020    6687#
DH71    042327          2064    2071    2078    2085    6696#
DISPLA  001142          1212#   2238*   2246*   2411*   2458*   2537*   2626*   2710*   2760*   2855*   2916*   2983*   3101*
                        3177*   3307*   3476*   3621*   3728*   3780*   4035*   4666*
DISPRE  000174           962#   2246
DPB     002500          1629#   2378*   2388*   2414*   2415*   2423*   2461*   2462*   2473*   2482*   2540*   2541*   2552*
                        2564*   2575*   2629*   2630*   2641*   2678*   2713*   2714*   2722*   2763*   2764*   2775*   2776*
                        2786*   2806*   2828*   2829*   2858*   2859*   2865*   2869*   2875*   2919*   2920*   2927*   2929*
                        2986*   2987*   2997*   2998*   3009*   3010*   3046*   3048*   3055*   3063*   3104*   3105*   3114*
                        3129*   3180*   3181*   3198*   3219*   3245*   3246*   3310*   3311*   3332*   3333*   3352*   3392*
                        3394*   3479*   3480*   3535*   3545*   3572*   3624*   3625*   3642*   3651*   3731*   3732*   3755*
                        3783*   3784*   4885    4887    4899    4900    4901    4902    5067    5197*   5225
DRIVE   031377          5497    5873#
DRSTAT  027236          2298    5664#
DRVBAD  001310          1263#   2278*   2375    4921*
DRVINT  022666          3664    4857    4865#   4912
DRVMSK  001306          1262#   2369*   2370    2373*
DRVNM   031352          5485    5869#
DRVOK   005004          2371    2378#
DRVRDY  023546          2390    2867    2877    3758    5012#   5030
DRVSEL  001312          1266#   2329*   2334*   2341    2370    2375*   3968    5479*   5494*   5512*
DRVSTA  001370          1308#   2307    2332    3665    4849*   4850*   4851*   4852*   4868*   4876*   4914    5495    5498
                        5510
DRVTST  027345          2338    5679#
DRVTYP  001504          1260#   2291    2313    3750    4848*   4879*   5068    5482
DSWR  = 177570          1004#   1211    2237
DT1     043036          1658    6760#
DT102   043422          2128    6808#
DT103   043450          2135    6812#
DT104   043504          2142    6817#
DT105   043512          2149    6818#
DT11    043126          1714    1766    6772#
DT111   043514          2177    6819#
DT12    043142          1722    2029    6774#
DT14    043156          1737    1774    1781    1795    1816    1823    1830    1874    1881    2036    2043    2050    2057
                        2093    2156    2163    2191    6776#
DT16    043172          1751    6778#
DT2     043040          1665    1672    1679    1700    1730    1744    1758    1788    1809    1837    1845    1852    1867
                        1888    1895    1909    1938    1968    1999    2100    2107    2114    2121    2184    6761#
DT25    043204          1802    1860    1961    1991    6780#
DT43    043224          1902    6783#
DT45    043262          1916    6789#
DT46    043306          1923    6793#
DT47    043322          1931    6795#
```

```
DT5     043072          1686    1693    2006    2013    6766#
DT51    043346          1946    1953    1976    1983    6799#
DT63    043366          2021    6802#
DT71    043404          2065    2072    2079    2086    6805#
EMTVEC= 000030          1093#   2223*   2224*
EM1     032173          1656    5947#
EM100   037715          2112    6464#
EM101   037735          2119    6467#
EM102   037762          2126    6471#
EM103   040005          2133    6475#
EM106   040064          2154    6483#
EM107   040120          2161    6488#
EM11    032570          1712    5992#
EM112   040154          2182    6493#
EM113   040212          2189    6498#
EM12    032663          1719    6002#
EM13    033014          1727    6018#
EM14    033137          1735    6033#
EM15    033173          1742    6038#
EM16    033223          1749    6043#
EM17    033260          1756    6048#
EM2     032235          1663    5953#
EM20    033312          1763    6053#
EM21    033425          1771    6066#
EM22    033535          1779    6080#
EM23    033572          1786    6085#
EM24    033641          1793    6092#
EM25    033716          1800    6100#
EM26    033766          1807    6107#
EM27    034056          1814    6117#
EM3     032266          1670    5958#
EM30    034107          1821    6122#
EM31    034142          1828    6127#
EM32    034173          1835    6132#
EM33    034252          1842    6140#
EM34    034363          1850    6153#
EM35    034457          1857    6164#
EM36    034561          1865    6176#
EM37    034642          1872    6185#
EM4     032357          1677    5968#
EM40    034676          1879    6190#
EM41    034733          1886    6195#
EM42    034764          1893    6200#
EM43    035042          1900    6208#
EM44    035112          1907    6215#
EM45    035165          1914    6223#
EM46    035245          1921    6232#
EM47    035316          1928    6239#
EM5     032414          1684    5973#
EM50    035445          1936    6255#
EM51    035524          1943    6263#
EM52    035620          1951    6274#
EM53    035704          1958    6283#
EM54    036013          1966    6295#
EM55    036072          1974    6303#
EM56    036162          1981    6313#
```

```
EM57    036255              1988    6323#
EM6     032466              1691    5980#
EM60    036371              1996    6337#
EM61    036465              2004    6348#
EM62    036522              2011    6353#
EM63    036557              2018    6358#
EM64    036673              2026    6371#
EM65    037000              2034    6383#
EM66    037037              2041    6389#
EM67    037101              2048    6395#
EM7     032522              1698    5985#
EM70    037156              2055    6403#
EM71    037230              2062    6410#
EM72    037344              2070    6423#
EM73    037415              2077    6430#
EM74    037467              2084    6438#
EM75    037553              2091    6447#
EM76    037646              2098    6457#
EM77    037675              2105    6461#
ENDCMP  025630              5292    5355#
ENTCYL  032054              5555    5928#
ENTNM   031324              5463    5865#
ENTPRM  026314              2327    5463#   5467    5471
ENTPRX  027070              5517    5521    5552    5586#
ENTPR1  026374              5476    5479#   5504
ENTPR2  026572              5518#
ENTPR3  026670              5535    5541#   5545
ENTPR4  026716              5547    5549#
ENTRK   032105              5565    5933#
ENTSEC  032133              5575    5937#
ENTWC   031523              5529    5889#
ERCTR   025414              5271*   5312#   5350*   5355    6818
ERR     = 100000            1180#   2477    2486    2509    2556    2568    2579    2645    2664    2682    2737    2780    2789
                            2820    2833    2885    2940    3002    3018    3052    3068    3119    3133    3202    3223    3250
                            3337    3356    3401    3539    3549    3576    3646
ERRVEC= 000004              1086#   2235    2236*   2247*   4632    4633*   4635*   4638*   4737    4738    4742*   4747*   4759*
                            4764*   4773*   4783*   4784*   5256*   5260*   5445    5446*   5448*   5450*
EXIT0   005306              2412    2429    2434    2439#
EXIT1   005722              2459    2513    2516#
EXIT10  011530              2984    3005    3079    3082#
EXIT11  012060              3102    3141    3145#
EXIT12  012712              3178    3205    3239    3262    3266#
EXIT13  013572              3308    3402    3404#
EXIT14  014616              3477    3571    3597#
EXIT15  015302              3622    3669    3691#
EXIT16  015620              3729    3760#
EXIT17  016604              3781    3935    3951#
EXIT2   006400              2538    2559    2571    2582    2596    2604#
EXIT3   007056              2627    2648    2685    2688    2692#
EXIT4   007306              2711    2733    2738    2740#
EXIT5   010032              2761    2836#
EXIT6   010366              2834    2856    2882    2896    2898#
EXIT7   010610              2917    2936    2941    2943#
FILSEC  024336              3047    5143#
FILTRK  024376              3115    5158#
FLAG    001202              1231#
```

```
FRSTER  025412          5270* 5311# 5322  5334*
GETADR  026150          2288  5427#
GETBUF  024502          3567  5190# 5209
GNS   = ****** U         961  3966  3972  3980  3997  4826  4827  4828  4829  4830  4832  4834  4835
                        4836  4837  4838
GTSWR = 104406          2260  4832#
HDR   = 004000          1176# 2775  2828  2875  2997  3009
HE    = 040000          1179# 2734  2817  2937
HIMEM   001234          1242# 3489* 3490* 3493* 3494* 5202  5205
HOMSEK= 000015          1157# 2388  2865
HT    = 000011           996# 4168  4209
ICNT    001214          1236#
IDE   = 000100          1174#
INCADR  024114          3552  3595  5092#
INCSEC  001250          1245# 3505* 3511* 5109
INCTRK  001252          1246# 3504* 3508* 5116
IOTVEC= 000020          1091# 2221* 2222*
KIPAR0= 172340          1124# 3209* 4723  4751  5218*
KIPAR1= 172342          1125# 3210* 5219*
KIPAR2= 172344          1126# 3211* 3240* 5220*
KIPAR3= 172346          1127# 5236* 6815
KIPAR4= 172350          1128# 5238* 6815
KIPAR5= 172352          1129# 5248* 6815
KIPAR6= 172354          1130#
KIPAR7= 172356          1131# 3208* 4767  5221*
KIPDR0= 172300          1113# 3212* 5239*
KIPDR1= 172302          1114# 3213* 5240*
KIPDR2= 172304          1115# 3214* 5241*
KIPDR3= 172306          1116# 5242* 6815
KIPDR4= 172310          1117# 5243* 6815
KIPDR5= 172312          1118# 5249* 6815
KIPDR6= 172314          1119#
KIPDR7= 172316          1120# 3215* 5244*
LDRFLG  001322          1270# 4979  4984* 4990  5006*
LF    = 000012           997# 4203  4209
LIMIT   025416          5267* 5268* 5313# 5338  5340*
LMT8K   001262          1250# 2272  2274  5538
LODSEC  024304          2996  5129#
LOMEM   001240          1243# 3488* 3495* 3496* 3497  3498  3518  3628* 3629* 3634  3641  3679  5207
                        5208
LRETRY  001316          1268# 3584
MAXCYL  001210          1234# 2290* 2293* 2654  2670  3638  5481* 5484* 5562  6778
MAXMEM  001230          1241# 3186  3238  3489  3490  5389* 5390* 5391* 5392* 5393* 5394* 5395* 5396*
                        5397* 5398* 5399* 5400* 5401* 5402* 5403* 5404* 5408* 5409* 5410
MAXWC   001264          1251# 2271  2272  5405* 5410* 5411* 5413* 5416* 5418* 5523  5536
MEMSIZ  001302          1258# 3494  3636  3682  5388* 5407*
MEX0  = 000020          1172# 2927  3226  3245  3253
MEX1  = 000040          1173# 2927  3230  3257
MMACTV  001222          1239# 2286* 3486* 3492* 5216  5251  5306  5324  5335
MMVEC = 000250          1102#
MODE  = 010000          1177# 2775  2828  2875  2997  3009  3332
MOVLDR  023346          3487  3627  4965#
MRPADR  032162          5431  5942#
MSG15A  027433          3626  5691#
MSG15B  027456          3649  5695#
MSG15C  027617          3690  5712#
```

```
MSG15D  027652            3659    5717#
MSG16A  027743            3734    5728#
MSG16B  030255            3740    5764#
MSG17A  030317            3785    5771#
MSG17B  030362            3790    5777#
MSG17C  030441            3791    5785#
MSG17D  030464            3806    5789#
MSG17E  030506            3824    5792#
MSG17F  030542            3839    5797#
MSG17G  030577            3858    5802#
MSG17H  030714            3877    5816#
MSG17I  031014            3879    3916    3936    5827#
MSG17J  031037            3911    5831#
MSG17K  031176            3952    5848#
MXWRDS  031447            5522    5881#
NONE    027372            2343    5683#
OFFLIN  027312            2319    5500    5672#
ONEPAT  031713            5553    5911#
ONLINE  027262            2310    5668#
OPRSEL  001260            1249#   2267*   3521    3561    5092    5585*
PAKSIZ  001244            1244#   3523*   3524*   3563*   3564*   5064*   5065*   5070*   5071*   5094    5096    5098    5100
                          5102*   5105*   5106*
PATNUM  001204            1232#   2468*   2492    2512    2514*   2550*   2637*   2651*   3520*   3560*   5078*   5079    5081*
                          5082    5173    5278    6766    6808    6812
PATSEL  024050            3525    3566    5077#
PATTRN  001300            1257#   2275*   5084    5548*
PAT.PT  001440            1336#   5177    5281
PAT0    001500            1336    1355#
PAT1    001540            1337    1372#
PAT10   002200            1346    1525#
PAT11   002240            1347    1542#
PAT12   002300            1348    1559#
PAT13   002340            1349    1576#
PAT14   002400            1350    1593#
PAT15   002440            1351    1610#
PAT2    001600            1338    1389#
PAT3    001640            1339    1406#
PAT4    001700            1340    1423#
PAT5    001740            1341    1440#
PAT6    002000            1342    1457#
PAT7    002040            1343    1474#
PAT8    002100            1344    1491#
PAT9    002140            1345    1508#
PIRQ   = 177772           1003#
PIRQVE= 000240            1097#
PR0    = 000000           1020#
PR1    = 000040           1021#
PR2    = 000100           1022#
PR3    = 000140           1023#
PR4    = 000200           1024#
PR5    = 000240           1025#
PR6    = 000300           1026#
PR7    = 000340           1027#   3633
PS     = 177776           1000#   1001    2287*   3705*
PSW    = 177776           1001#   2277*
PWRVEC= 000024            1092#
```

```
RDCHR = 104410          4556    4835#
RDERR   001274          1255#
RDLIN = 104411          3741    3797    3812    3830    3845    3859    3887    3922    3942    4836#   5438    5464    5486
                        5519    5530    5542    5550    5556    5566    5576
RDSEK = 000005          1153#   2423    2575    2678    2722    2726    2786    2929    3010    3055    3063    3129    3219
                        3246    3352    3572    3651
RDSW    016056          3818    3824#
RDY   = 000200          1175#   2425    2433    2879    5038    5045
READ  = 000017          1158#   2876
RESLDR  023452          2264    3598    3692    4989#
RESREG= 104413          4136    4838#   4860    4904    5007    5138    5153    5166    5185    5305
RESTAR  004762          2372#   4009
RESVEC= 000010          1087#
RETRY   001314          1267#   3565*   3578    3583*   3584    3588    6819
RPADR   001324          1274#   2386    2413    2460    2539    2628    2712    2762    2857    2918    2985    3103    3179
                        3309    3478    3623    3709    3730    3782    4853    4949    5432    5444*   5447    6760
RPBA  = 000010          1140#   2874*   4892*   4954
RPCA  = 000012          1141#   3861*   3867*   3889*   3896*   3914*   4893*   4955
RPCS  = 000004          1138#   2425    2430*   2431*   2726*   2876*   3670*   3800*   3815*   3833*   3848*   3915*   3949*
                        4854*   4855*   4865*   4885*   4899*   4900*   4903*   4952    5038    5045    5370*   5374*
RPDA  = 000014          1142#   3017    4894*   4956
RPDS  = 000000          1136#   4866    4870    4872    4874    4877    4950    5012    5019
RPER  = 000002          1137#   4951
RPINIT  022566          2289    3658    4844#   5480
RPM1  = 000016          1143#   4957
RPM2  = 000020          1144#
RPM3  = 000022          1145#
RPPRIO  001332          1276#   4847
RPVEC   001326          1275#   4845
RPWC  = 000006          1139#   2873*   4891*   4953
RP02    027327          2315    5675#
RP03    027336          2317    5677#
RP11    022770          2389    2424    2474    2483    2502    2553    2565    2576    2642    2657    2679    2723    2777
                        2787    2807    2830    2866    2870    2930    2999    3012    3049    3065    3116    3130    3199
                        3220    3247    3334    3353    3398    3536    3546    3573    3643    3652    3757    4884#
RSTRT1  004740          2368#   2377
RSTRT2  004752          2370#   2374
RSTRT3  004760          2371#
SAVREG= 104412          4078    4837#   4844    4884    4989    5129    5143    5158    5172    5215
SAVRP   023240          2427    2432    2476    2485    2504    2555    2567    2578    2644    2659    2681    2727    2779
                        2809    2832    2878    2884    2932    3001    3014    3051    3067    3118    3132    3201    3222
                        3249    3336    3355    3400    3538    3548    3575    3645    3671    3801    3816    3834    3849
                        3862    3868    3890    3898    3924    3944    4913    4948#   5024    5050
SCYL    001266          1252#   2268*   3517    3558    5564*
SECNBR  024002          3513    3554    5063#
SEEK  = 000011          1155#   2869    3755
SETBUF  024432          2469    2551    2562    2638    2652    3532    5172#
SILO  = 000026          1147#   4959
SIZMEM  025700          2265    5379#
SLASH   031322          5437    5864#
SR0   = 177572          1106#   3216*   3266*   4748    4763*   4770*   5261*   5308*   5330*   5337*
SR1   = 177574          1107#
SR2   = 177576          1108#
SR3   = 172516          1109#   4760*   5257*
SSEC    001272          1254#   2270*   3515    3556    5584*
STACK = 001100           991#   2219    2417    2464    2543    2632    2716    2766    2861    2922    2989    3111    3189
```

```
                    3313    3481    3512    3630    3704    3733    3786
STALL   023202      2725    2872    3016    3662    3707    4930#
STALLT  001220      1238#   2724*   2871*   3015*   3661*   3706*   4930
START   003710      2207    2212#
START1  003654       980    2205#
START2  003676       982    2210#   3693    3953    4397    4480
START3  003644       984    2203#
START4  003666       986    2208#
START5  004146      2250    2252    2254#
START6  004360      2287#
START7  004544      2297    2325#
STKLMT= 177774      1002#
STRK    001270      1253#   2269*   3516    3557    5574*
STRT1A  003660      2204    2206#
STRT2A  003702      2209    2211#
SUCA  = 000024      1146#   4958
SUFU  = 001000      1164#   4866
SUOL  = 040000      1167#   3802    3817    4870
SURDY = 100000      1168#   4874    5012    5019
SURPO3= 020000      1166#   4877
SUSI  = 004000      1165#   4872
SUWP  = 000400      1163#   3835    3850    3863    3869    3891    3899    3925    3945
SWR     001140      1211#   2217    2237*   2239    2245*   2258    2296    3366    3384    3482    3528    3533    3542
                    3570    3580    3592    3736    3738    4036    4043    4048    4052    4079    4400    4439    4494*
                    4627    4645    4652    4919    5029    5055    5284    5342
SWREG   000176       963#   2245    2258    4400    4439    4462
SW0   = 000001      1055#
SW00  = 000001      1045#   1055    3530    3533
SW01  = 000002      1044#   1054    3530    3542
SW02  = 000004      1043#   1053    3570
SW03  = 000010      1042#   1052    3366    3592    5284
SW04  = 000020      1041#   1051    3384    5342
SW05  = 000040      1040#   1050    3580
SW06  = 000100      1039#   1049    3482
SW07  = 000200      1038#   1048    2296    3736    3738
SW08  = 000400      1037#   1047
SW09  = 001000      1036#   1046    4919    5029    5055
SW1   = 000002      1054#
SW10  = 002000      1035#
SW11  = 004000      1034#
SW12  = 010000      1033#
SW13  = 020000      1032#   4079
SW14  = 040000      1031#
SW15  = 100000      1030#
SW2   = 000004      1053#
SW3   = 000010      1052#
SW4   = 000020      1051#
SW5   = 000040      1050#
SW6   = 000100      1049#
SW7   = 000200      1048#
SW8   = 000400      1047#
SW9   = 001000      1046#
TBITVE= 000014      1088#
TEST0   005160      2407    2417#
TEST1   005404      2454    2464#
TEST10  010706      2979    2989#   2990
```

```
TEST11  011652          3097    3111#
TEST12  012202          3173    3187    3189#   3190
TEST13  013014          3303    3313#
TEST14  014066          3472    3512#
TEST15  014736          3617    3630#
TEST16  015460          3724    3733#
TEST17  015716          3776    3786#
TEST2   006020          2533    2543#   2544
TEST3   006476          2622    2632#   2633
TEST4   007154          2706    2716#
TEST5   007404          2756    2766#   2767
TEST6   010130          2851    2861#
TEST7   010464          2912    2922#
TIMOUT  001212          1235#   5014    5040
TITLE   043656          2253    6914#
TKVEC =  000060         1095#   4377*   4378*
TPVEC =  000064         1096#
TRAPVE=  000034         1094#   2225*   2226*
TRTVEC=  000014         1089#
TSTING  027401          2379    5685#
TSTNMS  001276          1256#   2276*   2404    2451    2530    2619    2703    2753    2848    2909    2976    3094    3170
                        3300    3469    3614    3721    3773    5473*   5477*
TST0    005064          2391    2403#   2408
TST1    005310          2406    2450#   2455
TST10   010612          2911    2975#   2980
TST11   011532          2978    3093#   3098
TST12   012062          3096    3169#   3174
TST12B  012534          3234    3238#
TST13   012720          3172    3185    3188    3299#   3304
TST13A  013474          3340    3359    3367    3385    3388    3390#
TST14   013574          3302    3468#   3473
TST14A  014342          3531    3554#
TST15   014630          3471    3613#   3618
TST15A  015112          3658#   3710
TST15B  015314          3632    3697#   3698
TST15C  015324          3697    3702#
TST16   015372          3616    3720#   3725
TST17   015624          3723    3772#   3777
TST2    005724          2453    2529#   2534
TST3    006402          2532    2618#   2623
TST4    007060          2621    2702#   2707
TST5    007310          2705    2752#   2757
TST5A   007634          2800    2804#
TST5B   007742          2783    2813    2821    2823#
TST6    010034          2755    2847#   2852
TST7    010370          2850    2908#   2913
TYPDS =  104405         4121    4830#
TYPE  =  104401         2253    2298    2310    2312    2315    2317    2319    2320    2324    2338    2343    2363    2379
                        2385    3626    3649    3659    3690    3734    3740    3785    3790    3791    3806    3824    3839
                        3858    3877    3879    3911    3916    3936    3952    3964    3970    3978    3995    3999    4038
                        4046    4092    4093    4097    4098    4110    4113    4124    4128    4131    4133    4173    4270
                        4346    4394    4406    4460    4461    4464    4475    4485    4496    4515    4563    4569    4574
                        4578    4583    4584    4586    4589    4593    4826#   5430    5431    5437    5463    5485    5497
                        5500    5502    5503    5518    5522    5528    5529    5541    5549    5553    5555    5565    5575
TYPERR  017254          3378    3380    3591    4045    4078#   5328    5333    5349
TYPOC =  104402         3983    4117    4463    4827#
```