```
                .REM    \
```

IDENTIFICATION

PRODUCT CODE: AC-F9303-MC

PRODUCT NAME: ZRMPB0 RM05/3/2 DISKLESS TEST, PT 1

PRODUCT DATE: APRIL 1981

MAINTAINER: CX DIAGNOSTIC GROUP

AUTHOR: MIKE LEAVITT

CONTENTS

1.0 INTRODUCTION

1.1 ABSTRACT

THE RM05/3/2 DISKLESS DIAGNOSTIC IS A STAND ALONE PROGRAM WHICH USES FUNCTIONAL AND DIAGNOSTIC MEANS TO VERIFY THE OPERABILITY OF THE RM05/3/2 DISK SUBSYSTEM EXCLUDING AND INDEPENDENTLY OF THE STORAGE MODULE DRIVE. IN PARTICULAR, THE PROGRAM SERVES THE FOLLOWING PURPOSES:

TO DETECT ERRORS AND FAULTS IN THE RH MASSBUS CONTROLLER;

TO DETECT ERRORS AND FAULTS IN THE RM MASSBUS ADAPTER;

TO RESOLVE HARDWARE FAILURES IN THE RH/RM TO A FIELD REPLACEABLE MODULE OR MODULES.

1.2 UNIT UNDER TEST

THE UNIT UNDER TEST IS THE RM05/3/2 DISK SUBSYSTEM, EXCLUDING THE STORAGE MODULE DISK DRIVE AND THE RH11 OR RH70 MASSBUS CONTROLLER.

2.0 OPERATING REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

THE FOLLOWING MINIMUM HARDWARE CONFIGURATION, ASSUMED TO BE OPERATIONAL, IS REQUIRED TO LOAD AND EXECUTE THE RM05/3/2 DISKLESS DIAGNOSTIC:

PDP-11 PROCESSOR
20K MEMORY
KW11-L OR KW11-P CLOCK
PROGRAM LOADING DEVICE
TERMINAL
RH11 OR RH70 CONTROLLER
1 TO 8 DISK DRIVES (ANY COMBINATION OF RM05'S, RM03'S OR RM02'S)

2.2 MEDIA REQUIREMENTS

NONE

2.3 PREREQUISITE DIAGNOSTIC PROGRAMS

NONE

3.0 OPERATING PROCEDURE

3.1 LOADING

THE PROGRAM MAY BE LOADED BY EITHER PAPER TAPE, USING THE STANDARD PAPER TAPE LOADING PROCEDURE, OR XXDP MEDIA, USING THE APPROPRIATE LOADING DEVICE.

3.2 SWITCH OPTIONS

THE FOLLOWING SWITCH OPTIONS ARE INVOKED WHEN THE APPROPRIATE SWITCH IS ON.

SW15 HALT ON ERROR
SW14 LOOP ON TEST (CURRENTLY BEING EXECUTED)
SW13 INHIBIT ERROR TYPEOUTS
SW12 UNUSED
SW11 INHIBIT TEST ITERATIONS
SW10 BELL ON ERROR
SW09 LOOP ON ERROR
SW08 LOOP ON TEST IN SW07-00

THE LOW ORDER 8 SWITCHES ARE USED IN CONJUNCTION WITH SW08 TO SPECIFY A PARTICULAR TEST WHICH THE PROGRAM WILL LOOP ON.

3.3 STARTING

THE PROGRAM MAY BE STARTED AT LOCATION 200 OR 204. STARTING AT 200 WILL BE THE NORMAL STARTING ADDRESS. STARTING AT 204 WILL ENABLE THE RH/RM BASE ADDRESS TO BE CHANGED. IF RUNNING IN A STAND-ALONE ENVIRONMENT, THE PROGRAM USES CONSOLE DIALOGUE TO ALLOW THE OPERATOR TO CONTROL TEST CONDITIONS.

3.4 HALTING

THE PROGRAM CAN BE HALTED BY TYPING CONTROL C FROM THE CONSOLE OR BY PRESSING THE HALT SWITCH ON THE PROCESSOR FRONT PANEL.

3.5 RESTARTING

THE PROGRAM CAN BE RESTARTED AT ADDRESS 200 OR 204. (SEE SECTION 3.3)

4.0 OPERATOR INTERFACE

4.1 PROGRAM ID

THE PROGRAM TYPES ITS NAME AND MAINDEC NUMBER THE FIRST TIME IT IS STARTED AFTER BEING LOADED.

4.2 CONSOLE DIALOGUE

WHEN THE PROGRAM IS RUNNING IN STAND ALONE MODE, IT ENTERS A CONSOLE DIALOGUE SEQNENCE AFTER TYPING THE PROGRAM I.D..

THE FIRST QUESTION TYPED OUT IS: "TYPE HELP TEXT (L) N ?". IF THE OPERATOR RESPONDS WITH A "Y", THE PROGRAM WILL TYPE A BRIEF HELP MESSAGE WHICH WILL LIST SWITCH OPTIONS, ETC. ANY OTHER RESPONSE TO THE QUESTION IS CONSIDERED A "N" AND NO HELP TEXT IS TYPED. THIS QUESTION IS ONLY ASKED ON THE INITIAL PROGRAM START AND NOT ON SUBSEQUENT START-UP'S.

ON THE PROGRAM INITIAL START AND WHEN RESTARTING AT LOCATION 204, THE OPERATOR MAY CHANGE THE RH/RM BASE ADDRESSES WITH THE FOLLOWING DIALOGUE.

EXAMPLE 1

```
RMCS1=176700 <CR>        ;NO CHANGE IN ADDRESS
RMVEC=000254 <CR>        ;NO CHANGE IN ADDRESS
```

EXAMPLE 2

```
RMCS1=176700 177200<CR> ;CHANGE BASE ADDRESS TO 177200
RMVEC=000254 260<CR>    ;CHANGE VECTOR ADDRESS TO 260
```

ON THE INITIAL START, THE NEXT QUESTION TYPED IS, 'TYPE "A" TO TEST ALL DRIVES, OR TYPE DRIVE NUMBER(S) AND TERMINATE INPUT WITH A CARRIAGE RETURN'. THEN, 'DRIVE(S):' IS TYPED AND WAITS FOR THE OPERATOR TO TYPE AN "A", TO TEST ALL POSSIBLE DRIVES OR TYPE ANY STRING OF DRIVE NUMBER(S) TO BE TESTED AND TERMINATE THE INPUT WITH A "CARRIAGE RETURN". NO COMMAS OR ANY OTHER SEPARATORS ARE NEEDED WHEN ENTERING THE DRIVE NUMBERS AS A STRING. THE PROGRAM ENTERS THE COMMA SEPARTOR AUTOMATICALLY AFTER TYPING EACH NUMBER. ON ALL SUBSEQUENT STARTS, ONLY THE 'DRIVE(S):' PROMPT IS TYPED.

THE DIAGNOSTIC THEN INITIALIZES AND REPORTS THE STATUS OF THE DRIVES WHICH WHERE PREVIOUSLY SPECIFIED FOR TESTING. THE FOLLOWING IS AN EXAMPLE PRINTOUT:

```
'UNIT STATUS:
  0    ONLINE   RM03
  1    LOAD DEVICE
  2    OFFLINE  RM05
  3    NOT PRESENT
  4    NOT PRESENT
  5    NOT AN RM05/3/2
  6    NOT PRESENT
  7    NOT PRESENT'
```

THE ABOVE UNIT STATUS SHOWS THAT DRIVE 0 & 2 WILL BE TESTED, WHILE DRIVES 1, & 3 - 7 WILL NOT BE TESTED.

THE DIAGNOSTIC THEN TYPES THE FOLLOWING MESSAGE, BASED ON THE

STATUS OF THE DRIVE:

'DRIVE(S) TO BE TESTED, 0, 2'

IF NO DRIVES ARE AVAILABLE FOR TESTING, THE FOLLOWING MESSAGE WILL BE TYPED TO THE OPERATOR:

'DRIVE(S) TO BE TESTED, NONE'

THE PROGRAM WILL THEN, EITHER START TESTING THE DRIVES AVAILABLE FOR TESTING OR RETURN TO THE BEGINNING OF THE PROGRAM AND WAIT.

ONCE THE DRIVES START TESTING, THE FOLLOWING MESSAGE WILL OCCUR AS EACH DRIVE BEGINS TO BE TESTED:

'DRIVE 0
 DRIVE 2'

AFTER ALL THE DRIVES ARE COMPLETELY TESTED, THE END OF PASS MESSAGE WILL BE TYPED (SEE SECTION 4.3) AND THE PROGRAM WILL START TESTING ALL THE DRIVES AGAIN. THIS WILL CONTINUE UNTIL THE PROGRAM IS HALTED BY THE OPERATOR.

NOTE: THE LETTER LOCATED WITHIN THE BRACKETS () INDICATES THE TYPE OF RESPONSE REQUIRED BY THE USER, D DECIMAL, O-OCTAL AND L=LETTER.

4.3 PROGRESS REPORTS

AN END OF PASS REPORT OCCURS EACH TIME THE PROGRAM IS EXECUTED FOR ALL DEVICES IN THE TEST QUE. THE END OF PASS REPORT IS AS FOLLOWS.

'END OF PASS 1'

THE FOLLOWING MESSAGE WILL ALSO OCCUR IF THERE WERE ERRORS SINCE THE LAST END OF PASS REPORT.

'TOTAL ERRORS SINCE LAST REPORT 0'

4.4 PERFORMANCE REPORT

NO PERFORMANCE REPORTS ARE GIVEN DURING THE EXECUTION OF THE PROGRAM.

4.5 PROGRAM HALTS

THERE ARE NO SCHEDULED HALTS DURING THE EXECUTION OF THE PROGRAM. PROCESSOR HALTS ARE DUE TO THE TRAP CATCHER.

4.6 ERROR REPORTS

THE RM05/3/2 DISKLESS DIAGNOSTIC PROVIDES COMPREHENSIVE ERROR REPORTS INTENDED TO (1) AID IN FAULT RESOLUTION AND (2) MINIMIZE REFERENCES TO PROGRAM LISTINGS.

THE FIRST LINE OF THE ERROR REPORT CONTAINS THE NUMBER OF THE UNIT (DRIVE) BEING TESTED, DRIVE TYPE, THE TEST NUMBER, THE ERROR NUMBER AND THE VALUE OF THE PROGRAM COUNTER WHERE THE ERROR WAS CALLED. THIS LINE IS FOLLOWED BY THE ERROR MESSAGE: SEVERAL LINES OF TEXT WHICH GIVE A COMPREHENSIVE DESCRIPTION OF THE ERROR, AND A LIST OF FAILING MODULES IN ORDER OF DECREASING PROBABILITY. THE ERROR MESSAGE IS NORMALLY FOLLOWED BY ONE OR MORE PAIRS OF LINES CONTAINING DATA HEADERS AND DATA PERTININENT TO THE ERROR, INCLUDING EXPECTED AND ACTUAL TEST RESULTS.

THE FOLLOWING PRINTOUT IS AN ERROR MESSAGE IN THIS PROGRAM:

```
DRV# 0 - RM03, TEST# 25, ERR# 66, PC-017566
ILLEGAL REGISTER ERROR "ILR" (RMER1, BIT 01) SHOULD BE SET
DURING REGISTER TRANSFER
       PROBABLE FAULT(S):
       (NOT INCLUDING CABLES OR CONNECTORS)
       IF MODULE, M7686,

EXPCTD  RECEVD  TEST
STATUS  STATUS  REGSTR
000002  000000  176750
```

4.6 ERROR REPORTS

THE RM05/3/2 DISKLESS DIAGNOSTIC PROVIDES COMPREHENSIVE ERROR REPORTS INTENDED TO (1) AID IN FAULT RESOLUTION AND (2) MINIMIZE REFERENCES TO PROGRAM LISTINGS.

THE FIRST LINE OF THE ERROR REPORT CONTAINS THE UNIT NUMBER BEING TESTED, DRIVE TYPE, THE TEST NUMBER, THE ERROR NUMBER AND THE VALUE OF THE PROGRAM COUNTER WHERE THE ERROR WAS CALLED. THIS LINE IS FOLLOWED BY THE ERROR MESSAGE: SEVERAL LINES OF TEXT WHICH GIVE A COMPREHENSIVE DESCRIPTION OF THE ERROR, AND A LIST OF FAILING MODULES IN ORDER OF DECREASING PROBABILITY. THE ERROR MESSAGE IS NORMALLY FOLLOWED BY ONE OR MORE PAIRS OF LINES CONTAINING DATA HEADERS AND DATA PERTININENT TO THE ERROR, INCLUDING EXPECTED AND ACTUAL TEST RESULTS.

4.7 EXECUTION TIME

TIME FOR RM02/3:

PASS 1 OF THE PROGRAM TAKES ABOUT 20 SECONDS. PASS 2 AND SUBSEQUENT PASSES TAKE 1 MINUTE 35 SECONDS.

5.0 ENVIRONMENTAL SUPPORT

5.1 PROCESSOR COMPATIBILITY

THE RM05/3/2 DISKLESS DIAGNOSTIC IS EXECUTABLE ON ANY PDP-11 PROCESSOR, PROVIDING PREVIOUSLY MENTIONED HARDWARE REQUIREMENTS ARE MET.

5.2 DUAL PORT CONFIGURATIONS

THE RM05/3/2 DISKLESS DIAGNOSTIC IS NOT EXECUTABLE ON RM05/3/2 SUBSYSTEMS HAVING THE DUAL PORT OPTION UNLESS THE DUAL PORT SWITCH IS SET TO THE APPROPRIATE PORT (A OR B) AND NOT TO THE PROGRAMMABLE POSITION (A/B).

5.3 MEMORY PARITY HARDWARE

MEMORY PARITY HARDWARE WILL NOT BE USED DURING THE EXECUTION OF THE RM05/3/2 DISKLESS DIAGNGOSTIC.

5.4 MEMORY MANAGEMENT HARDWARE

MEMORY MANAGEMENT HARDWARE WILL NOT BE USED DURING THE RM05/3/2 DISKLESS DIAGNOSTIC.

5.5 ACT11, APT11 COMPATIBILITY

THE RM05/3/2 DISKLESS DIAGNOSTIC PROGRAM IS COMPATIBLE WITH ACT11 AND APT11 IN BOTH DUMP AND AUTOMATIC MODES. FURTHER, THE PROGRAM WILL EXECUTE A QUICK PASS DURING THE FIRST PASS IN SUPPORT OF QUICK VERIFY MODE.

5.6 XXDP COMPATIBILITY

THE RM05/3/2 DISKLESS DIAGNOSTIC PROGRAM IS COMPATIBLE WITH XXDP IN DUMP AND CHAIN MODES.

5.7 OPERATING SYSTEM COMPATIBILITY

THE PROGRAM IS NOT REQUIRED TO BE COMPATIBLE WITH ANY OPERATING SYSTEM.

6.0 TEST DESCRIPTION

THE PROGRAM IS DESIGNED IN A BOTTOM UP MANNER SUCH THAT EACH TEST

GENERALLY USES A MORE COMPLEX SUBSET OF HARDWARE THAN THE PREVIOUS TEST.

MODULE CALLOUT IS PREDICATED ON THE ASSUMPTION THAT EARLIER TESTS HAVE BEEN COMPLETED WITHOUT ERROR AND THAT ERRORS ARE DUE TO SINGLE, NONTRANSIENT HARDWARE FAILURES.

THE RM05/3/2 DISKLESS DIAGNOSTIC CAN BE EXECUTED USING AN RH70 OR AN RH11 MASSBUS CONTROLLER.

UNLESS SPECIFIED BY THE OPERATOR OR BY THE ENVIRONMENT TABLE THE TEST IS REPEATED FOR EACH POSSIBLE DEVICE STARTING WITH DEVICE 0.

THE MODULES WHICH MAY BE CALLED OUT DURING THE EXECUTION OF THE TEST ARE AS FOLLOWS:

IF
CS
DS
MASSBUS MODULE

THE RADIAL MODULE (RD) IS NOT TESTED BY THIS PROGRAM.

TEST 1 TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE RM05/3/2 CAN COMPLETE A REGISTER TRANSFER ON THE MASSBUS, AND, IN PARTICULAR, TO VERIFY THAT ''TRANSFER'' IS NOT STUCK IN AN INACTIVE STATE.

PROCEDURE:

THE PROGRAM WRITES AND READS REMOTE REGISTERS FOR THE SELECTED DEVICE. REGISTER CONTENTS AND PARITY ERRORS ARE IGNORED, AND THE TEST FAILS IF A 'NONEXISTENT DEVICE ERROR'' OR BUS TIMEOUT OCCURRS FOR EVERY REGISTER ACCESS. IF THE TEST FAILS THE PROGRAM JUMPS TO THE END OF PASS HANDLER WHICH SELECTS THE NEXT DEVICE TO BE TESTED.

PROBABLE FAULT:

THE TEST FAILS IF THE SELECTED DEVICE IS NONEXISTENT OR IS SWITCHED TO THE PROGRAMMABLE POSITION OR TO THE ALTERNATE PORT. THE FOLLOWING FAULTS ARE APPLICABLE ONLY WHEN THE DEVICE IS PRESENT AND IS SWITCHED TO THE APPROPRIATE PORT.

1. IF MODULE

2. ASYNCHRONOUS MASSBUS MODULE

3. CS MODULE

TEST 2 CTOD TEST

PURPOSE:

TO VERIFY THAT DATA CAN BE TRANSFERRED TO AND FROM THE RM05/3/2 USING THE CONTROL BUS AND, IN PARTICULAR, TO VERIFY THAT ''CONTROLLER TO DEVICE'' HAS NOT FAILED.

PROCEDURE:

THE TEST WRITES ONES IN REMOTE REGISTERS THEN READS EACH REGISTER WHICH WILL WRITE ZEROS IN THE REGISTER IF ''IF3 CTOD HOLD H'' IS STUCK AT ONE. THE TEST THEN READS AS MANY REMOTE REGISTERS AS ARE NECESSARY TO OBTAIN ONE OR MORE ONE BITS.

PROBABLE FAULT:

1. IF MODULE

2. ASYNCHRONOUS MASSBUS MODULE

TEST 3 MASSBUS INITIALIZE TEST

PURPOSE:

TO VERIFY THAT THE MASSBUS ADAPTER IS BEING INITIALIZED BY THE MASS BUS.

PROCEDURE:

USING CONTROLLER CLEAR TO INITIALIZE THE SELECTED UNIT, THIS TEST THEN READS MASSBUS ADAPTER REGISTERS TO VERIFY THAT AT LEAST ONE BIT IS CLEARED. MASSBUS ADAPTER REGISTERS ARE PRESET TO A NON ZERO VALUE PRIOR TO CONTROLLER CLEAR.

PROBABLE FAULT:

1. ASYNCHRONOUS MASSBUS MODULE

2. IF MODULE

3. CS MODULE

TEST 4 CLEAR STUCK ACTIVE TEST

PURPOSE:

TO VERIFY THAT 'MBA CLR L'' ON THE CS MODULE IS NOT STUCK IN AN ACTIVE STATE.

PROCEDURE:

CONTROLLER CLEAR IS USED TO INITIALIZE THE SELECTED UNIT, AFTER WHICH 1'S ARE WRITTEN IN ERROR REGISTERS 1 AND 2 AND MAINTENANCE REGISTER 1. IF ANY 1 BITS CAN BE READ BACK THE TEST IS OK, ELSE, 'MBA CLR L'' IS PROBABLY STUCK ACTIVE.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

3. ASYNCHRONOUS MASSBUS MODULE

TEST 5 TRISTATE TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE PATH TO AND FROM THE MASSBUS ADAPTER TRI-STATE REGISTER BUS IS NOT STUCK AT ONE OR ZERO AND THAT EACH BIT POSITION IS INDEPENDENT.

PROCEDURE:

THIS TEST PRESETS MASSBUS ADAPTER REGISTERS TO A NONZERO VALUE, THEN, ASSUMING THE REGISTERS ARE PRESET, IT CLEARS THEM USING A MOVE INSTRUCTION. THE TEST THEN READS AS MANY REGISTERS AS IS NECESSARY TO OBTAIN ONE OR MORE ZEROS FROM EACH BIT POSITION.

THE TEST CLEARS MASSBUS ADAPTER REGISTERS, THEN, ASSUMING THE REGISTERS ARE CLEARED, IT LOADS THEM WITH ONES AND READS AS MANY REGISTERS AS IS NECESSARY TO OBTAIN ONE OR MORE ONE BITS IN EACH BIT POSITION.

FINALLY, THE TEST WRITES A SINGLE ONE BIT PATTERN IN BIT 0 OF SELECTED REMOTE REGISTERS AND VERIFIES THAT THE PATTERN CAN BE READ BACK. THE ONE BIT IS SHIFTED AND THE TEST REPEATED FOR ALL BIT POSITIONS.

PROBABLE FAULT:

1. ASYNCHRONOUS MASSBUS MODULE

2. IF MODULE

3. CS MODULE

4. DS MODULE

TEST 6 REGISTER SELECT TEST

PURPOSE:

TO VERIFY THAT THE REGISTER SELECT LINES APE NOT IN A STUCK POSITION.

PROCEDURE:

EACH REGISTER SELECT LINE IS TESTED BY WRITING ZEROS IN THOSE DEVICE REGISTERS FOR WHICH THE LINE MUST BE ZERO, THEN WRITING ONES IN THOSE DEVICE REGISTERS FOR WHICH THE LINE MUST BE ONE. THE ZERO REGISTER IS READ BACK AND IF THE SELECT LINE IS STUCK AT ZERO, THE ZERO REGISTER WILL CONTAIN ONES. THE PROCESS IS REPEATED TO DETECT A STUCK AT ONE FAULT, EXCEPT IN THIS CASE, THE ONES REGISTER IS WRITTEN FIRST.

REGISTER SELECT LINES 1, 2, 4 AND 8 ARE TESTED IN THIS MANNER; SELECT LINE 16 IS EXPLICITLY TESTED IN THE ''ILR TEST''.

PROBABLE FAULT:

1. IF MODULE

2. ASYNCHRONOUS MASSBUS MODULE

TEST 7 DRIVE TYPE TEST

PURPOSE:

TO TEST THE 'DRIVE TYPE'' REGISTER, RMDT.

PROCEDURE:

THE PROGRAM READS RMDT AND VERIFIES THAT THE RESULT CORRESPONDS TO A SINGLE PORT OR DUAL PORT RM05, RM[illegible] OR RM02 DRIVE.

PROBABLE FAULT:

1. IF MODULE

TEST 10 DEVICE AVAILABLE TEST

PURPOSE:

TO VERIFY THAT DEVICE AVAILABLE STATUS IS SET.

PROCEDURE:

THE PROGRAM TESTS 'DVA", BIT 11 OF RMCS1.

PROBABLE FAULT:

1. IF MODULE

TEST 11 HOLDING REGISTER TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE HOLDING REGISTER IS NOT STUCK AT ONE, STUCK AT ZERO, AND THAT THERE IS NO BIT INTERFERENCE.

PROCEDURE:

THE PROGRAM TRANSFERS ONES, THEN ZEROS TO THE HOLDING REGISTER AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ONE.

THE PROGRAM TRANSFERS ZEROS, THEN ONES TO THE HOLDING REGISTER AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ZERO.

FINALLY, THE TEST TRANSFERS A SHIFTING ONE BIT PATTERN AND VERIFIES THAT EACH BIT IS INDEPENDENT.

PROBABLE FAULT:

1. IF MODULE

TEST 12 CONTROL STATUS #1 TRANSFER TEST

PURPOSE:

TO VERIFY THAT BITS 01 THROUGH 05 OF CONTROL STATUS REGISTER 1 ARE NOT STUCK AT ONE OR ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

PROCEDURE:

THIS TEST WRITES ONES IN CONTROL STATUS REGISTER 1, RMCS1, THEN WRITES ZEROS AND VERIFIES THAT THE BITS ARE NOT STUCK AT ONE. THE GO BIT IS NOT TESTED IN THIS TEST.

NEXT, THE TEST CLEARS THE CONTROL STATUS REGISTER, RMCS1, WRITES ONES IN BITS 01 THROUGH 05 AND VERIFIES THAT THE BITS ARE NOT STUCK AT ZERO. THE GO BIT IS NOT TESTED.

THE TEST TRANSFERS A SHIFTING ONE BIT DATA PATTERN TO AND FROM RMCS1 AND CHECKS FOR ADJACENT BIT INTERFERENCE.

PROBABLE FAULT:

1. IF MODULE

TEST 13 ERROR REGISTER #1 TRANSFER TEST

PURPOSE:

TO VERIFY THAT ERROR REGISTER 1 IS NOT STUCK AT ONE OR ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

PROCEDURE:

THIS TEST WRITES ONES IN ERROR REGISTER 1, RMER1, THEN WRITES ZEROS AND VERIFIES THAT THE REGISTER IS NOT STUCK AT ONE. 'UNSAFE'' IS NOT TESTED DURING THIS TEST. IN ORDER TO LIMIT THE PROBABLE FAULTS TO ONE OR TWO MODULES, THE TEST IS EXECUTED IN 3

PARTS WITH EACH PART TESTING THOSE BITS WHOSE PRESET FUNCTIONS ARE DERIVED FROM THE SAME MODULE.

THE TEST WRITES ZEROS IN ERROR REGISTER 1, RMER1, THEN WRITES ONES AND VERIFIES THAT THE REGISTER IS NOT STUCK AT ZERO.

FINALLY, THE TEST WRITES A SHIFTING ONE BIT PATTERN IN RMER1 AND CHECKS FOR ADJACENT BIT INTERFERENCE.

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

3. DS MODULE

TEST 14 CLEAR OFFSET STUCK ACTIVE TEST

PURPOSE:

TO VERIFY THAT THE SIGNAL WHICH CLEARS OFFSET MODE IS NOT STUCK IN ACTIVE STATE.

PROCEDURE:

THE TEST WRITES A ONE IN THE OFFSET DIRECTION BIT WHICH IS CLEARED BY THE SIGNAL AND VERIFIES THAT A ONE CAN BE READ BACK.

PROBABLE FAULT:

1. IF MODULE

2. DS MODULE

TEST 15 OFFSET REGISTER TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE OFFSET REGISTER IS NOT STUCK AT ONE, STUCK AT ZERO, AND THAT THERE IS NO ADJACENT BIT INTERFERENCE.

PROCEDURE:

THE OFFSET REGISTER, RMOF, IS WRITTEN WITH ONES, THEN WRITTEN WITH ZEROS AND READ TO VERIFY THAT NONE OF THE BITS ARE STUCK AT ONE.

THEN THE OFFSET REGISTER IS WRITTEN WITH ZEROS AND WRITTEN WITH ONES TO VERIFY THAT THE REGISTER IS NOT STUCK AT ZERO.

FINALLY, THE OFFSET REGISTER IS TESTED WITH A SHIFTING ONE BIT PATTERN.

PROBABLE FAULT:

1. IF MODULE

2. DS MODULE

TEST 16 ERROR REGISTER #2 TRANSFER TEST

PURPOSE:

TO VERIFY THAT ERROR REGISTER 2, RMER2, IS NOT STUCK AT ONE, STUCK AT ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

PROCEDURE:

THE TEST WRITES ONES THEN WRITES ZEROS IN RMER2 AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ONE. "SKI" AND "DVC" ARE NOT TESTED. IN ORDER TO LIMIT THE NUMBER OF PROBABLE FAULTS TO ONE OR TWO MODULES, THE TEST IS EXECUTED IN 3 PARTS WITH EACH PART

TESTING THOSE BITS WHOSE PRESET FUNCTIONS ARE DERIVED FROM THE SAME MODULE.

THEN THE TEST WRITES ZEROS IN ERROR REGISTER 2, AND WRITES ONES VERIFYING THAT THE REGISTER IS NOT STUCK AT ZERO.

FINALLY, THE TEST WRITES A SHIFTING ONE BIT PATTERN IN THE REGISTER AND VERIFIES THAT ALL BIT POSITIONS ARE INDEPENDENT.

PROBABLE FAULT:

1. IF MODULE
2. CS MODULE
3. DS MODULE

TEST 17 SERIAL NUMBER TEST

PURPOSE:

TO VERIFY THAT THE SERIAL NUMBER CAN BE READ.

PROCEDURE:

THE TEST READS THE SERIAL NUMBER REGISTER SEVERAL TIMES AND VERIFIES THAT THE NUMBER IS THE SAME EACH TIME.

PROBABLE FAULT:

1. CS MODULE

TEST 20 CONTROL BUS PARITY DETECTION TEST

PURPOSE:

TO TEST THE RM05/3/2'S PARITY CHECKING LOGIC FOR THE MASSBUS ASYNCHRONOUS CONTROL BUS.

PROCEDURE:

THIS TEST WRITES A SHIFTING ONE BIT DATA PATTERN IN THE DISK ADDRESS REGISTER USING "PAT" TO CONTROL THE STATE OF THE PARITY BIT. "PAR" STATUS, BIT 03 OF RMER1, IS CHECKED AFTER EACH PATTERN IS TRANSFERRED..NOTE THE FOLLOWING TABLE SHOWS A SET OF TEST PATTERNS THAT COULD BE USED INSTEAD OF A SHIFTING ONE BIT PATTERN.

DATA PATTERN	PAT	PAR

000000	1	1
075266	0	0
163753	0	0
116535	1	1

PROBABLE FAULT:

1. IF MODULE

2. ASSYNCHRONOUS MASSBUS MODULE

TEST 21 CONTROL BUS PARITY GENERATION TEST

PURPOSE:

TO TEST THE RM05/3/2'S PARITY GENERATING LOGIC FOR THE MASSBUS ASYNCHRONOUS CONTROL BUS.

PROCEDURE:

THE TEST TRANSFERS A SHIFTING ONE BIT DATA PATTERN TO THE DISK ADDRESS REGISTER. AFTER EACH PATTERN IS READ BACK, 'MASSBUS CONTROL BUS PARITY ERROR'' IS TESTED AND SHOULD BE ZERO..NOTE THE FOLLOWING SET OF TEST PATTERNS COULD BE USED INSTEAD OF THE SHIFTING ONE BIT PATTERN.

DATA PATTERN	MCPE
000000	0
056747	0
135672	0
163135	0

PROBABLE FAULT:

1. IF MODULE

2. ASYNCHRONOUS MASSBUS MODULE

TEST 22 RMDA, RMDC FAULT TEST

PURPOSE:

TO VERIFY THAT THERE ARE NOT FAULTS WHICH INHIBIT THE PROGRAM FROM WRITING RMDC AND RMDA. SPECIFICALLY, THESE FAULTS INCLUDE:

.''GO H'' STUCK HIGH, WHICH WOULD INHIBIT THE REGISTER LOAD

FUNCTION,

.'RIP '' STUCK LOW, WHICH WOULD CONSTANTLY CLEAR THE REGISTER

.'EBL'' STUCK, WHICH WOULD INHIBIT THE CLOCK FUNCTION.

PROCEDURE:

THE TEST WRITES AND READS BOTH RMDC, AND RMDA. WITH ZEROS, THEN ONES. THE TEST PASSES IF EITHER REGISTER CAN BE WRITTEN WITH ONES.

PROBABLE FAULT:

1. DS MODULE

2. IF MODULE

3. CS MODULE

TEST 23 DISK ADDRESS TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE DISK ADDRESS REGISTER IS NOT STUCK AT ONE OR ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

PROCEDURE:

THIS TEST PRESETS THE DISK ADDRESS TO A NONZERO VALUE, THEN USES A MOVE TO CLEAR THE REGISTER.THE TEST THEN READS RMDA AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ONE.

THEN THE TEST PRECLEARS THE MASSBUS ADAPTER DISK ADDRESS REGISTER (RMDA), LOADS IT TO ALL ONES, AND VERIFIES THAT NONE OF THE BITS ARE STUCK AT ZERO.

A SHIFTING ONE BIT PATTERN IS TRANSFERRED TO AND FROM THE DISK ADDRESS REGISTER, RMDA, AND THE TEST VERIFIES THAT EACH BIT IS INDEPENDENT.

PROBABLE FAULT:

1. DS MODULE

2. IF MODULE

TEST 24 DESIRED CYLINDER TRANSFER TEST

PURPOSE:

TO VERIFY THAT THE DESIRED CYLINDER ADDRESS REGISTER, RMDC, IS NOT STUCK AT ONE OR ZERO, AND THAT THERE IS NOT BIT INTERFERENCE.

PROCEDURE:

THIS TEST WRITES ONES IN THE DESIRED CYLINDER REGISTER RMDC, THEN WRITES ZEROS AND VERIFIES THAT THE REGISTER IS NOT STUCK AT ONE.

THEN THE TEST WRITES ZEROS IN THE DESIRED CYLINDER REGISTER, RMDC, WRITES ONES AND VERIFIES THAT THE REGISTER IS NOT STUCK AT ZERO.

FINALLY, A SHIFTING 1 BIT PATTERN IS TRANSFERPED TO AND FROM RMDC AND THE PROGRAM CHECKS FOR BIT INTERFERENCE.

PROBABLE FAULT:

1. DS MODULE

2. IF MODULE

TEST 25 ILLEGAL REGISTER TEST

PURPOSE:

TO TEST ILLEGAL REGISTER ERROR DETFCTION IN THE RM05/3/2.

PROCEDURE:

THIS TEST READS ALL LEGAL REGISTERS AND VERIFIES THAT ''ILR'', BIT 2 OF RMER1 DOES NOT SET. THEN, TO THE EXTENT ALLOWED BY THE MASSBUS CONTROLLER, IT READS ILLEGAL REGISTERS AND VERIFIES THAT ''ILR'' IS SET.

PROBABLE FAULT:

1. IF MODULE

2. ASSYNCHRONOUS MASSBUS MODULE

TEST 26 RESET GO BY INIT TEST

PURPOSE:

TO VERIFY THAT GO CAN BE RESET BY INITIALIZE.

PROCEDURE:

THE TEST SETS GO THEN CLEARS GO USING MASSBUS INITIALIZE, I.E., CONTROLLER CLEAR.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 27 DIAGNOSTIC MODE TEST

PURPOSE:

TO VERIFY THAT 'DIAGNOSTIC MODE', BIT 0 OF RMMR1, IS NOT STUCK AT ONE OR ZERO.

PROCEDURE:

THE RM05/3/2 IS INITIALIZED AND 'DMD' IS CHECKED FOR ZERO. 'DMD' IS WRITTEN WITH ONE AND READ TO VERIFY THAT IT IS NOT STUCK AT ZERO, THEN WRITTEN WITH ZERO AND READ TO VERIFY THAT IT IS NOT STUCK AT ONE.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 30 MOL TEST

PURPOSE:

TO VERIFY THAT 'MEDIUM ON LINE' STATUS CAN BE SET AND RESET USING MAINTENANCE UNIT READY.

PROCEDURE:

AFTER INITIALIZING THE SUBSYSTEM, THE TEST SETS 'DIAGNOSTIC MODE: AND READS THE DRIVE STATUS REGISTER, RMDS, EXPECTING MOL, BIT 12 TO BE ZERO. 'MAINTENANCE UNIT READY', BIT 9 OF RMMR1, IS SET AND MOL SHOULD BE ONE. THE TEST THEN WRITES A ZERO IN MUR AND READS RMDS, VERIFYING THAT 'MEDIUM ON LINE' IS ZERO.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 31 WRITE LOCK TEST

PURPOSE:

TO VERIFY THAT "WRITE LOCK" STATUS, WRL, CAN BE SET AND RESET USING "MAINTENANCE WRITE PROTECT", MWP.

PROCEDURE:

WITH DIAGNOSTIC MODE SET, THE PROGRAM SETS MWP, BIT 03 OF RMMR1, AND READS RMDS TO VERIFY THAT WRL, BIT 11 IS SET. THEN MWP IS RESET AND WRL SHOULD BE ZERO.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 32 DRIVE FAULT TEST

PURPOSE:

TO VERIFY THAT "DEVICE CHECK", DVC, AND "UNSAFE", UNS, CAN BE SET AND RESET USING "MAINTENANCE DRIVE FAULT", MDF.

PROCEDURE:

WITH DIAGNOSTIC MODE SET, THE PROGRAM SETS MDF, BIT 06 OF RMMR1, AND READS RMER3 TO VERIFY THAT DVC, BIT 07 IS SET RMER1 IS ALSO READ AND UNS, BIT 14 SHOULD ALSO BE SET. THEN MDF IS RESET AND DVC AND UNS SHOULD BE RESET.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 33 SEEK ERROR TEST

PURPOSE:

TO VERIFY THAT "SEEK ERROR", SKI, CAN BE SET AND RESET USING "MAINTENANCE SEEK ERROR", MSER.

PROCEDURE:

WITH DIAGNOSTIC MODE SET, THE TEST SETS MSER, BIT 07 OF RMMR1 AND READS RMER3 TO VERIFY THAT SKI, BIT 14 IS SET. MSER IS RESET AND SKI SHOULD RESET.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 34 PIP TEST

PURPOSE:

TO VERIFY THAT "POSITIONING IN PROGRESS", PIP, CAN BE SET AND RESET USING "MAINTENANCE ON CYLINDER", MOC.

PROCEDURE:

DIAGNOSTIC MODE IS SET THEN MOC, BIT 08 OF RMMR1 IS SET AND PIP, BIT 13 OF RMDS, SHOULD BE ZERO. MOC IS THEN RESET AND PIP SHOULD BE ONE.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 35 EBL TEST

PURPOSE:

TO VERIFY THAT END OF BLOCK STATUS "EBL" CAN BE SET AND RESET USING DIAGNOSTIC END OF BLOCK "DEBL".

PROCEDURE:

THE PROGRAM SETS DIAGNOSTIC MODE AND VERIFIES THAT EBL IS RESET. THEN IT SETS DEBL AND VERIFIES THAT EBL IS SET. FINALLY, THE TEST TRANSFERS A SHIFTING ONE BIT TO RMMR1, AND CHECKS FOR

DEBL BEING SET BY AN ADJACENT BIT.

PROBABLE FAULT:

1. CS MODULE

TEST 36 LAST SECTOR, LAST TRACK TEST

PURPOSE:

TO VERIFY THE DESIRED TRACK/SECTOR PLA ON THE DS MODULE USING RMMR1, BITS 01 AND 02.

PROCEDURE:

THE TEST WRITES ALL POSSIBLE PATTERNS IN THE DISK ADDRESS REGISTER, RMDA, AND VERIFIES "LS" AND "LST" STATUS FOR EACH PATTERN. THE PROCEDURE IS DONE ONCE FOR 18 BIT FORMAT AND ONCE FOR 16 BIT FORMAT.

PROBABLE FAULT:

1. DS MODULE

2. CS MODULE

TEST 37 RMDA COUNT TEST

PURPOSE:

TO VERIFY THAT THE DISK ADDRESS REGISTER (RMDA) INCREMENTS PROPERLY.

PROCEDURE:

THE TEST INCREMENTS RMDA USING DIAGNOSTIC END OF BLOCK "DEBL" AND VERIFIES THE RESULT IN 18 BIT FORMAT AND ONCE FOR 16 BIT FORMAT.

PROBABLE FAULT:

1. DS MODULE

TEST 40 RMDC COUNT TEST

```
1198                    PURPOSE:
1199
1200                        TO VERIFY THAT THE DESIRED CYLINDER REGISTER, RMDC,
1201                    INCREMENTS PROPERLY.
1202
1203                    PROCEDURE:
1204
1205                        THE PROGRAM INCREMENTS RMDC USING DIAGNOSTIC END OF BLOCK,
1206                    "DEBL", AND VERIFIES THE RESULT IN 18 BIT FORMAT AND 16 BIT FORMAT.
1207
1208                    PROBABLE FAULT:
1209
1210                    1.  DS MODULE
1211
1212
1213
1214
1215
1216                TEST 41 LBT TEST
1217
1218                    PURPOSE:
1219
1220                        TO INSURE THAT LAST BLOCK TRANSFERRED, "LBT", CLEARS WHEN
1221                    RMDA IS WRITTEN, AND SETS WHEN THE LAST SECTOR IS TRANSFERRED.
1222
1223                    PROCEDURE:
1224
1225                        THE TEST USES DIAGNOSTIC EBL TO SET LBT, AND TRANSFERS TO
1226                    RMDA TO RESET LBT.  THE RESULTS ARE VERIFIED IN 18 BIT FORMAT AND
1227                    16 BIT FORMAT.
1228
1229
1230                    PROBABLE FAULT:
1231
1232                    1.  DS MODULE
1233
1234                    2.  IF MODULE
1235
1236
1237
1238
1239
1240                TEST 42 COMPOSITE ERROR TEST
1241
1242                    PURPOSE:
1243
1244                        TO TEST "COMPOSITE ERROR", BIT 14 OF RMDS.
1245
1246                    PROCEDURE:
1247
1248                        THE TEST USES INITIALIZE AND DIAGNOSTIC MODE TO FORCE ALL
1249                    ERRORS TO ZERO THEN VERIFIES THAT "ERR" IS ZERO.  EACH ERROR IS
1250                    INDIVIDUALLY SET AND "ERR" SHOULD BE ONE FOR EVERY ERROR TESTED.
1251                    ADDRESSES #2 AND #17 OF THE COMPOSITE ERROR PLA ARE NOT TESTED.
1252                    "ABORT" AND "EXCEPTION" OUTPUTS OF THE PIA ARE NOT TESTED.  THE
1253                    TEST FAILS IF ERR IS NOT ZERO WITH ALL SET ARGUMENTS ZERO OR IF
1254                    ERR IS NOT ONE WITH ANY SET ARGUMENT ONE.
```

```
1255
1256                                    PROBABLE FAULT:
1257
1258                                    1.  IF MODULE
1259
1260
1261
1262
1263
1264                            TEST 43 WRITE GO TEST
1265
1266                                    PURPOSE:
1267
1268                                        TO VERIFY THAT GO CAN BE SET.
1269
1270                                    PROCEDURE:
1271
1272                                        THE TEST ENABLES THE  DEBUG  CLOCK,  THEN  TRANSFERS  A  NOP
1273                                    FUNCTION  CODE  AND GO BIT TO RMCS1, VERIFYING THAT GO SETS.  ALL
1274                                    FUNCTION CODES ARE TESTED.
1275
1276                                    PROBABLE FAULT:
1277
1278                                    1.  IF MODULE
1279
1280                                    2.  CS MODULE
1281
1282
1283
1284
1285
1286                            TEST 44 BRANCH MULTIPLEXOR TEST
1287
1288                                    PURPOSE:
1289
1290                                        TO VERIFY THAT THE OUTPUT OF THE  COMMAND  SEQUENCER  BRANCH
1291                                    MULTIPLEXOR DOES NOT HAVE A FAULT.
1292
1293                                    PROCEDURE:
1294
1295                                        WITH DEBUG CLOCK ENABLED, THE  TEST  USES  VARIOUS  FUNCTION
1296                                    CODES AND REGISTER CONDITIONS TO ADDRESS THE TEST BIT MULTIPLEXOR
1297                                    SUCH THAT THE TEST BIT, BIT12 OF RMMR2,  CAN  BE  CHECKED  FOR  A
1298                                    STUCK FAULT.
1299
1300                                    PROBABLE FAULT:
1301
1302                                    1.  CS MODULE
1303
1304
1305
1306
1307
1308                            TEST 45 SET/RESET GO TEST
1309
1310                                    PURPOSE:
1311
```

TO VERIFY THAT GO CAN BE SET AND RESET.

PROCEDURE:

THE SUBSYSTEM IS INITIALIZED AND PUT IN DIAGNOSTIC MODE WITH "DEBUG CLOCK ENABLE", BIT 14 OF RMMR1 SET. CERTAIN FUNCTION CODES ARE WRITTEN IN RMCS1 AND THE PROGRAM READS RMCS1 TO VERIFY THAT GO IS SET. RMDS IS ALSO READ TO VERIFY THAT "DRY" IS RESET. THEN THE PROGRAM STEPS THE DEBUG CLOCK USING BIT 15 OF RMMR1 AND VERIFIES THAT "GO" RESETS AND "DRY" SETS. USING A FUNCTION CODE THAT RESETS GO AT A DIFFERENT PROM ADDRESS. THE TEST FAILS IF GO DOES NOT SET OR CANNOT BE RESET BY THE COMMAND SEQUENCER. THE TEST ALSO FAILS IF "DRIVE READY" IS NOT THE COMPLIMENT OF GO.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 46 END 1 RESET GO TEST

PURPOSE:

TO VERIFY THAT THE COMMAND SEQUENCER CAN RESET GO AT THE END1 LOCATION.

PROCEDURE:

THE TEST EXECUTES RELEASE, SEARCH AND ILLEGAL FUNCTION CODE 32 IN DIAGNOSTIC MODE AND VERIFIES THAT GO RESETS ON THE SPECIFIED CLOCK CYCLE.

PROBABLE FAULT:

1. CS MODULE

TEST 47 SET PULSE TEST

PURPOSE:

TO VERIFY THAT THE COMMAND SEQUENCER CAN GENERATE SET PULSE.

PROCEDURE:

WITH DEBUG CLOCK ENABLED, THE TEST STEPS THE COMMAND SEQUENCER THROUGH PARTS OF VARIOUS FUNCTION CODES AND CHECKS CONTINUE, BIT 06 OF RMMR1 TO DETERMINE IF SET PULSE IS BEING

```
1369                                    GENERATED.
1370
1371                                    PROBABLE FAULT:
1372
1373                                    1.  CS MODULE
1374
1375
1376
1377
1378
1379                                TEST 50 SET/RESET IVC TEST
1380
1381                                    PURPOSE:
1382
1383                                        TO  TEST  ''INVALID COMMAND''  STATUS  FOR EACH FUNCTION CODE.
1384
1385                                    PROCEDURE:
1386
1387                                        THE PROGRAM RESETS  VOLUME  VALID  USING  'MAINTENANCE  UNIT
1388                                READY'',  BIT09  OF  RMMR1, THEN LOADS THE FUNCTION CODE AND GO IN
1389                                RMCS1.  EACH FUNCTION CODE IS TESTED AND ''IVC'', BIT 12  OF  RMER2
1390                                IS CHECKED.
1391
1392                                    PROBABLE FAULT.
1393
1394                                    1.  CS MODULE
1395
1396                                    2.  IF MODULE
1397
1398
1399
1400
1401
1402                                TEST 51 SET LSC TEST
1403
1404                                    PURPOSE:
1405
1406                                        TO  VERIFY  THAT  'LOSS OF SYSTEM CLOCK''  CAN SET AND RESET.
1407
1408                                    PROCEDURE:
1409
1410                                        THE TEST ENABLES THE DEBUG CLOCK AND SETS THE GO BIT.  AFTER
1411                                WAITING  ENOUGH  TIME  FOR THE ONE SHOT TO SET, THE TEST DISABLES
1412                                THE DEBUG CLOCK AND VERIFIES THAT LSC SETS.
1413
1414                                    PROBABLE FAULT:
1415
1416                                    1.  CS MODULE
1417
1418                                    2.  IF MODULE
1419
1420
1421
1422
1423
1424                                TEST 52 DECODE TEST
1425
```

PURPOSE:

TO VERIFY THAT THE "DECODE" FLOP ON THE IF MODULE SETS WITH THE LEADING EDGE OF "SET PULSE" EXCEPT WHEN "COMPOSITE ERROR" IS ACTIVE.

PROCEDURE:

THE TEST USES "VOLUME VALID" AND "OCCUPPIED" TO DETERMINE IF THE DECODE FLOP IS SET OR RESET. INITIALLY, VV AND OCCUPPIED ARE RESET AND THE TEST EXECUTES THOSE COMMANDS WHICH SET VV OR OCC AND VERIFIES THAT ONE OR BOTH BITS SET. THE SAME COMMANDS ARE EXECUTED AGAIN WITH COMPOSITE ERROR SET, AND THE TEST VERIFIES THAT NEITHER BIT SETS.

PROBABLE FAULT.

1. IF MODULE

TEST 53 SET/RESET VOLUME VALID TEST

PURPOSE:

TO VERIFY THAT "VOLUME VALID" RESETS WITH THE LEADING EDGE OF UNIT READY, AND SETS WITH PACK ACKNOWLEDGE AND READ IN PRESET COMMANDS.

PROCEDURE:

USING "MAINTENANCE UNIT READY", BIT 9 OF RMMR1, THIS TEST FORCES A ZERO TO ONE TRANSITION OF UNIT READY AND VERIFIES THAT VOLUME VALID,BIT 6 OF RMDS IS ZERO. THEN THE TEST EXECUTES A PACK ACKNOWLEDGE COMMAND, VERIFYING THAT VV SETS. THE PROCEDURE IS REPEATED WITH A READ IN PRESET COMMAND.

PROBABLE FAULT:

1. IF MODULE

TEST 54 ILLEGAL FUNCTION TEST

PURPOSE:

TO TEST ILLEGAL FUNCTION ERROR IN THE RM05/3/2.

PROCEDURE:

WITH DIAGNOSTIC CLOCK ENABLED TO INHIBIT THE COMMAND SEQUENCER, THIS TEST VERIFIES THAT "ILF", BIT 0 OF RMER1, IS OFF

FOR LEGAL FUNCTION CODES AND ON FOR ILLEGAL FUNCTIONCODES. THE STATUS OF THE "GO" BIT IS IGNORED.

PROBABLE FAULT:

1. IF MODULE

TEST 55 OCCUPIED TEST

PURPOSE:

TO VERIFY THAT "OCCUPIED" IS SET DURING DATA TRANSFERS AND IS RESET FOR ALL OTHER COMMANDS.

PROCEDURE:

FOR EACH DATA TRANSFER COMMAND, "OCC", BIT 15 OF RMMR1 SHOULD BE ONE, DEBUG CLOCK IS ENABLED TO PREVENT GO FROM RESETTING BEFORE STATUS IS SAMPLED.

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

TEST 56 READ IN PRESET TEST

PURPOSE:

TO VERIFY THAT "READ IN PRESET" COMMAND IS DECODED, AND IN PARTICULAR, TO VERIFY THAT "IFS READ IN CMD L" IS NOT STUCK AT ONE.

PROCEDURE:

EACH VISIBLE STATUS OR REGISTER BIT WHICH IS CLEARED BY "READ IN PRESET" IS SET. THEN THE RIP COMMAND IS EXECUTED AND THE TEST VERIFIES THAT ONE OR MORE BITS ARE CLEARED. THE FOLLOWING ARE USED DURING THE TEST.

. ALL BITS OF RMOF ARE SET BY A MOVE INSTRUCTION AND THE TEST PASSES IF USED BITS ARE ZERO AFTER THE RIP COMMAND.

. THE DESIRED CYLINDER REGISTER, RMDC, IS SET WITH A MOVE INSTRUCTION AND THE TEST PASSES IF BITS 00-09 ARE ZERO AFTER THE RIP COMMAND.

. THE DISK ADDRESS REGISTER, RMDA, IS SET WITH A MOVE INSTRUCTION

AND THE TEST PASSES IF BITS 00-07, AND BITS 08-15 ARE ZERO AFTER THE RIP COMMAND.

THE TEST FAILS IF NONE OF THE PRESET TERMS ARE ZERO AFTER THE RIP COMMAND.

PROBABLE FAULT:

1. IF MODULE

2. DS MODULE

TEST 57 RIP/RMOF TEST

PURPOSE:

TO VERIFY THAT "READ IN PRESET" RESETS FMT16, ECI AND HCI BITS 10, 11 AND 12 OF RMOF.

PROCEDURE:

FMT16, ECI AND HCI ARE SET, THEN A RIP COMMAND IS EXECUTED AND EACH BIT SHOULD BE ZERO.

PROBABLE FAULT:

1. IF MODULE

2. DS MODULE

TEST 60 RMDA/RMDC/RIP TEST

PURPOSE:

TO VERIFY THAT "READ IN PRESET" RESETS THE DESIRED CYLINDER ADDRESS, RMDC, AND THE DISK ADDRESS, RMDA.

PROCEDURE:

RMDA AND RMDC ARE PRESET THEN TESTED FOR ZERO AFTER THE RIP COMMAND.

PROBABLE FAULT:

1. DS MODULE

TEST 61 OFFSET COMMAND TEST

PURPOSE:

TO VERIFY THAT 'OFFSET MODE'' SETS WITH OFFSET COMMAND.

PROCEDURE:

THE TEST EXECUTES OFFSET COMMAND AND VERIFIES THAT 'OM'', BIT 00 OF RMDS IS ONE.

PROBABLE FAULT:

1. IF MODULE

TEST 62 RETURN TO CENTER TEST

PURPOSE:

TO VFRIFY THAT 'RETURN TO CENTER'' RESETS OFFSET MODE.

PROCEDURE:

OFFSET MODE, BIT 00 OF RMDS, IS SET WITH OFFSET COMMAND, THEN THE TEST EXECUTES A RETURN TO CENTER COMMAND AND VERIFIES THAT OFFSET MODE RESETS. OFFSET DIRECTION IS ALSO SET AND CHECKED FOR ZERO AFTER THE COMMAND.

PROBABLE FAULT:

1. IF MODULE

TEST 63 RMDC CLEAR OFFSET TEST

PURPOSE:

TO VERIFY THAT CLEAR OFFSET IS ACTIVE WHEN THE DESIRED CYLINDER ADDRESS IS WRITTEN.

PROCEDURE:

THE TEST EXECUTES AN OFFSET COMMAND, WRITES RMDC, AND VERIFIES THAT OM, BIT 00 OF RMDS IS ZERO.

PROBABLE FAULT:

1. DS MODULE

2. IF MODULE

TEST 64 EBL CLEAR OFFSET TEST

PURPOSE:

TO VERIFY THAT OFFSET MODE CLEARS WHEN HEAD SWITCHING OCCURS.

PROCEDURE:

THE TEST EXECUTES AN OFFSET COMMAND TO SET OFFSET MODE. AFTER SETTING THE FORMAT BIT AND LOADING THE LAST SECTOR/TRACK ADDRESS IN RMDA, THE TEST FORCES AN EBL AND VERIFIES THAT OFFSET MODE RESETS.

PROBABLE FAULT:

1. DS MODULE

TEST 65 RUN AND GO TEST

PURPOSE:

TO VERIFY THAT 'RUN AND GO'' FLOP SETS DURING READ AND WRITE COMMANDS.

PROCEDURE:

THE RM05/3/2 IS INITIALIZED AND A DATA TRANSFER COMMAND WITH GO SET IS WRITTEN IN RMCS1. 'RUN AND GO'', BIT 14 OF RMMR1 SHOULD BE ONE FOR EACH DATA COMMAND. THE DEBUG CLOCK IS ENABLED SO THAT GO DOES NOT RESET BEFORE STATUS IS TESTED.

PROBABLE FAULT:

1. CS MODULE

2. SYNCHRONOUS MASSBUS MODULE

TEST 66 SET IAE TEST

PURPOSE:

TO VERIFY THAT INVALID ADDRESS ERROR CAN SET.

PROCEDURE:

THE TEST LOADS INVALID SECTOR, TRACK AND CYLINDER ADDRESSES AND EXECUTES A SEARCH COMMAND, VERIFYING THAT ''IAE'' SETS. THE PROCESS IS REPEATED WITH A DIFFERENT COMMAND IF THE IAE DOES NOT SET, AND THE TEST FAILS IF IAE CANNOT BE SET.

PROBABLE FAULT:

1. DS MODULE

2. IF MODULE

TEST 67 SEARCH, SEEK, READ, WRITE TEST

PURPOSE:

TO VERIFY THAT THE ''SCH SK R OR W'' DECODE ON THE IF MODULE IS CORRECT FOR ALL FUNCTION CODES.

PROCEDURE:

THE TEST LOADS INVALID SECTOR, TRACK AND CYLINDER ADDRESSES AND EXECUTES EACH COMMAND TO WHERE SET PULSE IS ACTIVE AND VERIFIES THE DECODE BY CHECKING ''IAE''.

PROBABLE FAULT:

1. IF MODULE

TEST 70 INVALID TRACK/SECTOR TEST

PURPOSE:

TO VERIFY THAT INVALID TRACK AND SECTOR ADDRESSES ARE DETECTED.

PROCEDURE:

THE TEST LOADS THE TEST PATTERN IN RMDA AND EXECUTES A SEARCH COMMAND, VERIFYING THAT ''IAE'' SETS.

PROBABLE FAULT:

1. DS MODULE

2. TRACK ADDRESS OPTION JUMPER

TEST 71 INVALID CYLINDER TEST

PURPOSE:

TO VERIFY THAT INVALID CYLINDER ADDRESSES ARE DETECTED.

PROCEDURE:

THE TEST LOADS THE TEST PATTERN IN RMDC AND EXECUTES A SEARCH COMMAND, VERIFYING THAT ''IAE'' SETS.

PROBABLE FAULTS:

1. DS MODULE

2. CYLINDER ADDRESS OPTION JUMPER

TEST 72 SET AOE TEST

PURPOSE:

TO VERIFY THAT ADDRESS OVERFLOW ERROR IS DETECTED.

PROCEDURE:

THE TEST LOADS THE ADDRESS OF THE LAST SECTOR IN RMDA AND RMDC, THEN INITIATES A DATA COMMAND WITH DEBUG CLOCK ENABLED. END OF BLOCK IS FORCED TO INCREMENT THE SECTOR ADDRESS, AND THE TEST VERIFIES THAT ''AOE'' IS SET, IN 18 BIT FORMAT AND 16 BIT FORMAT.

PROBABLE FAULT:

1. DS MODULE

TEST 73 SET RMR TEST

PURPOSE:

TO VERIFY THAT ''REGISTER MODIFICATION REFUSED'' SETS WHEN A REGISTER IS WRITTEN WHILE GO IS SET, EXCEPT WHEN THE ATTENTION OR MAINTENANCE REGISTER IS WRITTEN.

PROCEDURE:

''DEBUG CLOCK ENABLE'' IS SET TO INHIBIT THE COMMAND

SEQUENCER, THEN A NOP COMMAND AND GO BIT IS WRITTEN IN RMCS1. WITHOUT STEPPING THE DEBUG CLOCK, THE TEST WRITES RMMR AND RMAS, WHICH SHOULD NOT SET RMR STATUS. THEN RMDA IS WRITTEN AND RMR STATUS, BIT 02 OF RMER", SHOULD BE ONE.

PROBABLE FAULT:

1. IF MODULE

TEST 74 PGM STATUS CHECK

PURPOSE:

TO VERIFY THAT THE PROGRAMMABLE STATUS BIT AND THE DRIVE REQUEST STATUS BIT ARE COMPATABLE.

PROCEDURE:

THE TEST REPORTS AN ERROR IF PGM IS ON AND DRQ IS OFF. PGM IS NOT PREDICTABLE IN THE CASE WHERE DRQ IS ON BECAUSE OF THE PORT SELECT SWITCH.

PROBABLE FAULT:

1. IF MODULE

TEST 75 DVA/DPR STATUS CHECK

PURPOSE:

TO VERIFY THAT DEVICE AVAILABLE STATUS AND DRIVE PRESENT STATUS ARE SET.

PROCEDURE:

DVA AND DPR ARE TESTED AND BOTH SHOULD BE ON.

PROBABLE FAULT:

1. IF MODULE

TEST 76 PORT REQUEST TEST, PART 1

PURPOSE:

TO VERIFY THAT THE PORT REQUEST FLOPS ON THE IF MODULE SET WHEN THE PROGRAM READS RMCS1.

PROCEDURE:

THE TEST EXECUTES A RELEASE COMMAND, THEN, ASSUMING THE PORT IS RELEASED, IT READS RMCS1, THEN READS RMMR2 AND VERIFIES THAT ONE OF THE PORT REQUEST FLOPS IS SET.

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

TEST 77 PORT REQUEST TEST, PART 2

PURPOSE:

TO VERIFY THAT THE PORT REQUEST FLOPS ON THE IF MODULE SET WHEN THE PROGRAM WRITES RMAS.

PROCEDURE:

THE TEST EXECUTES A RELEASE COMMAND THEN WRITES RMAS AND READS RMMR2, VERIFYING THAT ONE OF THE REQUEST FLOPS IS SET.

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

TEST 100 PORT REQUEST TEST, PART 3

PURPOSE:

TO VERIFY THAT PORT REQUEST SETS WHEN ANY REGISTER EXCEPT RMAS IS WRITTEN.

PROCEDURE:

THE TEST WRITES THE DISK ADDRESS REGISTER AND VERIFIES THAT THE PORT REQUEST FLOP IS ON.

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

TEST 101 RELEASE TEST

PURPOSE:

TO VERIFY THAT A RELEASE COMMAND CAN RESET THE REQUEST FLOPS RQA AND RQB IN MAINTANCE REGISTER #2.

PROCEDURE:

THE PROGRAM SETS REQUEST FLOP BY WRITTING THE RMCS1 REGISTER THEN, EXECUTES A RELEASE COMMAND TO RESET THE REQUEST FLOP.

PROBABLE FAULT:

1. IF MODULE

TEST 102 WRITE ATA TEST

PURPOSE:

TO VERIFY THAT ATTENTION CAN BE CLEARED BY WRITING THE ATTENTION SUMMARY REGISTER.

PROCEDURE:

THE PROGRAM RESETS AND SETS UNIT READY WHICH SHOULD CAUSE AN ATTENTION, THEN WRITES THE ATTENTION SUMMARY REGISTER AND VERIFIES THAT ATTENTION IS RESET.

PROBABLE FAULT:

1. IF MODULE

2. CS MODULE

TEST 103 RESLT ATA BY GO TEST

PURPOSE:

TO VERIFY THAT ATA RESETS WHEN GO IS ON AND COMPOSITE ERROR IS OFF.

PROCEDURE:

1996
1997 THE PROGRAM SETS MAINTENANCE UNIT READY WHICH SHOULD CAUSE
1998 AN ATTENTION. THEN, WITH DEBUG CLOCK ENABLED, GO IS SET, AND ATA
1999 SHOULD BE ZERO.
2000
2001 PROBABLE FAULT:
2002
2003 1. IF MODULE
2004
2005
2006
2007
2008
2009 TEST 104 UNIT READY ATA TEST
2010
2011 PURPOSE:
2012
2013 TO VERIFY THAT ONE-ZERO AND ZERO-ONE TRANSITIONS OF UNIT
2014 READY SET ATTENTION.
2015
2016 PROCEDURE:
2017
2018 THE TEST USES DIAGNOSTIC MODE TO FORCE BOTH TRANSITIONS OF
2019 UNIT READY AND VERIFIES THAT ATA SETS WITH EACH TRANSITION.
2020
2021 PROBABLE FAULT:
2022
2023 1. IF MODULE
2024
2025
2026
2027
2028
2029 TEST 105 ERROR ATA TEST
2030
2031 PURPOSE:
2032
2033 TO VERIFY THAT ATTENTION SETS WHEN COMPOSITE ERROR OCCURS
2034 WHILE GO IS OFF.
2035
2036 PROCEDURE:
2037
2038 THE PROGRAM CLEARS THE DEVICE AND SETS AN ERROR, THEN
2039 VERIFIES ATA IS ON.
2040
2041 PROBABLE FAULT:
2042
2043 1. IF MODULE
2044
2045
2046
2047
2048
2049 TEST 106 REGISTER TRANSFER ATA TEST
2050
2051 PURPOSE:
2052

```
2053                        TO VERIFY THAT ATTENTION SETS WHEN ANY REGISTER, EXCEPT  FOR
2054                    RMAS AND RMCS, IS WRITTEN WHILE COMP ERROR IS SET.
2055
2056                    PROCEDURE:
2057
2058                        THE PROGRAM FORCES AN ERROR THEN RESETS ATTENTION  FROM  THE
2059                    ERROR.   THE  PROGRAM THEN WRITES RMAS AND RMCS AND VERIFIES THAT
2060                    NO ATTENTION OCCURS, AND WRITES RMDC AND VERIFIES THAT  ATTENTION
2061                    DOES OCCUR.
2062
2063                    PROBABLE FAULT:
2064
2065                    1.  IF MODULE
2066
2067
2068
2069
2070
2071                TEST 107 P SET ATA TEST
2072
2073                    PURPOSE:
2074
2075                        TO VERIFY THAT ATA IS SET AT THE COMPLETETION OF AN OFFSET AND
2076                    RETURN TO CENTER I INE COMMAND.
2077
2078                    PROCEDURE:
2079
2080                        THE  PROGRAM  EXECUTES  THE COMMANDS USING THE MAINTANCE DEBUG
2081                    CLOCK AND EXPECTS ATA TO BE SET ON COMPLETETION.
2082
2083                    PROBABLE FAULT:
2084
2085                    1.  IF MODULE
2086
2087
2088
2089
2090
2091                TEST 110 SET WLE TEST
2092
2093                    PURPOSE:
2094
2095                        TO VERIFY THAT 'WLE'' IS SET OR RESET WHEN IT SHOULD BE.
2096
2097                    PROCEDURE:
2098
2099                        THE  PROGRAM  EXECUTES  THE  FOLLOWING  COMMANDS  USING  THE
2100                    MAINTANCE DEBUG CLOCK AND EXPECTS WLE SET OR RESET.
2101
2102                        EXECUTE WRITE DATA COMMAND WITH MAINTANCE WRITE PROTECT SET,
2103                    SHOULD EXPECT WLE TO BE SET.
2104
2105                        EXECUTE  WRITE  DATA COMMAND WITHOUT MAINTANCE WRITE PROTECT
2106                    SET, SHOULD EXPECT WLE TO BE RESET.
2107
2108                        EXECUTE  READ DATA COMMAND WITH MAINTANCE WRITE PROTECT SET,
2109                    SHOULD EXPECT WLE TO BE RESET.
```

EXECUTE READ IN PRESET COMMAND WITH MAINTANCE WRITE PROTECT SET, SHOULD EXPECT WLE TO BE RESET.

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 111 EXCEPTION TEST

PURPOSE:

TO VERIFY THAT "REX" OF RMMR1 IS RESET AFTER THE CONTROLLER IS INTIALIALIZED AND SET WHEN AN ERROR IS DETECTED DURING A DATA TRANSFER COMMAND.

PROCEDURE:

THE PROGRAM WILL INITIALIZE THE MASSBUS ("REX" SHOULD BE CLEAR) AND THEN EXECUTE THE WRITE DATA COMMAND USING THE MAINTANCE DEBUG CLOCK. WHILE THE COMMAND IS BEING EXECUTED (RUN AND GO SET), THE PROGRAM CAUSES A "RMR" ERROR, BY TRYING TO WRITE THE RMER1 REGISTER ("REX" SHOULD BE SET).

PROBABLE FAULT:

1. CS MODULE

2. IF MODULE

TEST 112 RECALIBRATE TEST

PURPOSE:

TO VERIFY THAT "OPI" SETS, IF UNIT READY DROPS DURING RECALIBRATE COMMAND EXECUTION.

TO VERIFY THAT THE RECALIBRATE COMMAND ABORTS DURING COMMAND EXECUTION.

TO VERIFY THAT "OPI" SETS, IF ON CYLINDER LATCH DOES NOT CLEAR.

TO VERIFY THAT "ATA" SETS, IF THE DRIVE COMPLETES THE RECALIBRATE COMMAND.

TO VERIFY THAT THE RECALIBRATE COMMAND ABORTS AFTER EXECUTION DURING A WAIT LOOP.

2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223

TO VERIFY THE TAG BUS DURING A RECALIBRATE COMMAND.

PROCEDURE:

THE PROGRAM EXECUTES THE FOLLOWING COMMANDS USING THE MAINTANCE DEBUG CLOCK AND EXPECTS THE RESULTS FOLLOWING EACH COMMAND.

EXECUTE RECALIBRATE COMMAND, DROP UNIT READY AND VERIFY THAT 'DPI' IS SET.

EXECUTE RECALIBRATE COMMAND, SET DRIVE FAULT ('MDF' IN RMMR1) TO CAUSE COMMAND ABORT AND VERIFY THAT 'GO' IS RESET.

EXECUTE RECALIBRATE COMMAND, VERIFY THAT 'DPI' IS SET WHEN ON CYLINDER LATCH IS NOT CLEARED.

EXECUTE RECALIBRATE COMMAND, DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER AGAIN AND VERIFY THAT 'ATA' IS SET.

EXECUTE RECALIBRATE COMMAND, DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER RESET AND VERIFY THAT 'GO' IS STILL SET.

EXECUTE RECALIBRATE COMMAND AND VERIFY THAT THE TAG BUS IS CORRECT ACCORDING A PRE-DETERMINED TABLE.

PROBABLE FAULT:

1. CS MODULE

TEST 113 SEEK TEST

PURPOSE:

TO VERIFY THAT 'DPI' SETS, IF UNIT READY DROPS DURING SEEK COMMAND EXECUTION.

TO VERIFY THAT THE SEEK COMMAND ABORTS DURING COMMAND EXECUTION.

TO VERIFY THAT 'DPI' SETS, IF ON CYLINDER LATCH DOES NOT CLEAR.

TO VERIFY THAT 'ATA' SETS, IF THE DRIVE COMPLETES THE SEEK COMMAND.

TO VERIFY THAT THE SEEK COMMAND ABORTS AFTER EXECUTION DURING A WAIT LOOP.

TO VERIFY THE TAG BUS DURING A SEEK COMMAND.

PROCEDURE:

THE PROGRAM EXECUTES THE FOLLOWING COMMANDS USING THE MAINTANCE DEBUG CLOCK AND EXPECTS THE RESULTS FOLLOWING EACH COMMAND.

```
2224                    EXECUTE SEEK COMMAND, DROP UNIT READY AND VERIFY THAT 'DPI' IS SET.
2225
2226                    EXECUTE SEEK COMMAND, SET DRIVE FAULT ('MDF' IN RMMR1) TO CAUSE
2227                COMMAND ABORT AND VERIFY THAT 'GO' IS RESET.
2228
2229                    EXECUTE SEEK COMMAND, VERIFY THAT 'DPI' IS SET WHEN ON CYLINDER
2230                LATCH IS NOT CLEARED.
2231
2232                    EXECUTE SEEK COMMAND, DROP ON CYLINDER TO RESET LATCH, THEN SET
2233                ON CYLINDER AGAIN AND VERIFY THAT 'ATA' IS SET.
2234
2235                    EXECUTE SEEK COMMAND, DROP ON CYLINDER TO RESET LATCH, LEAVE ON
2236                CYLINDER RESET AND VERIFY THAT 'GO' IS STILL SET.
2237
2238                    EXECUTE SEEK COMMAND AND VERIFY THAT THE TAG BUS IS CORRECT
2239                ACCORDING A PRE-DETERMINED TABLE.
2240
2241                PROBABLE FAULT:
2242
2243                1.  CS MODULE
2244
2245
2246
2247
2248
2249            TEST 114 SEARCH TEST
2250
2251                    TO VERIFY THAT 'DPI' SETS, IF UNIT READY DROPS DURING SEARCH COMMAND
2252                EXECUTION.
2253
2254                    TO VERIFY THAT THE SEARCH COMMAND ABORTS DURING COMMAND EXECUTION.
2255
2256                    TO VERIFY THAT 'DPI' SETS, IF ON CYLINDER LATCH DOES NOT CLEAR.
2257
2258                    TO VERIFY THAT 'ATA' SETS, IF THE DRIVE COMPLETES THE SEARCH COMMAND.
2259
2260                    TO VERIFY THAT THE SEARCH COMMAND ABORTS AFTER EXECUTION DURING A
2261                 WAIT LOOP.
2262
2263                    TO VERIFY THAT SEARCH COMMAND ABORTS DURING SECTOR COMPARE LOOP
2264
2265                    TO VERIFY THE TAG BUS DURING A SEARCH COMMAND.
2266
2267                PROCEDURE:
2268
2269                    THE PROGRAM EXECUTES THE FOLLOWING COMMANDS USING THE
2270                MAINTANCE DEBUG CLOCK AND EXPECTS THE RESULTS FOLLOWING EACH COMMAND.
2271
2272                    EXECUTE SEARCH COMMAND, DROP UNIT READY AND VERIFY THAT 'DPI' IS SET.
2273
2274                    EXECUTE SEARCH COMMAND, SET DRIVE FAULT ('MDF' IN RMMR1) TO CAUSE
2275                COMMAND ABORT AND VERIFY THAT 'GO' IS RESET.
2276
2277                    EXECUTE SEARCH COMMAND, VERIFY THAT 'DPI' IS SET WHEN ON CYLINDER
2278                LATCH IS NOT CLEARED.
2279
2280                    EXECUTE SEARCH COMMAND, DROP ON CYLINDER TO RESET LATCH, THEN SET
```

```
2281                    ON CYLINDER AGAIN AND VERIFY THAT "ATA" IS SET.
2282
2283                         EXECUTE SEARCH COMMAND, DROP ON CYLINDER TO RESET LATCH, LEAVE ON
2284                    CYLINDER RESET AND VERIFY THAT "GO" IS STILL SET.
2285
2286                         EXECUTE SEARCH COMMAND, WHILE IN SECTOR COMPARE LOOP, SET DRIVE
2287                    FAULT ("MDF" IN RMMR1) TO CAUSE COMMAND ABORT AND VERIFY THAT "ATA" IS
2288                    SET.
2289
2290                         EXECUTE SEARCH COMMAND AND VERIFY THAT THE TAG BUS IS CORRECT
2291                    ACCORDING A PRE-DETERMINED TABLE.
2292
2293                    PROBABLE FAULT:
2294
2295                    1.  CS MODULE
2296
2297
2298
2299
2300
2301               TEST 115 SEARCH TIMEOUT TEST
2302
2303                    PURPOSE:
2304
2305                         TO VERIFY THAT "DPI" SETS, IF "MSEN" (SEARCH TIMEOUT ENABLE) IS
2306                    DROPPED DURING SEARCH COMMAND EXECUTION.
2307
2308                    PROCEDURE:
2309
2310                         EXECUTE SEARCH COMMAND, VERIFY THAT "DPI" IS SET WHEN "MSEN" IS
2311                    CLEARED.
2312
2313                    PROBABLE FAULT:
2314
2315                    1.  CS MODULE
2316
2317
2318
2319
2320
2321               TEST 116 - 120 DATA COMMAND TESTS (1, 2, 3)
2322
2323                    PURPOSE:
2324
2325                         TO VERIFY THE COMMAND SEQUENCER DURING DATA COMMANDS.
2326
2327                    PROCEDURE:
2328
2329                         THIS TEST, LIKE RECALIBRATE, SEEK, AND SEARCH TESTS, USES
2330                    THE MAINTENANCE REGISTER TO SIMULATE DRIVE CONDITIONS AND FORCE
2331                    THE COMMAND SEQUENCER THROUGH EACH BRANCH PATH.  ADDITIONAL ITEMS
2332                    WHICH ARE TESTED INCLUDE OFFSET PLUS AND MINUS ON THE TAG BUS AND
2333                    "ENABLE SEARCH", BIT 11 OF RMMR1.
2334
2335                    PROBABLE FAULT:
2336
2337                    1.  CS MODULE
```

```
2338
2339
2340
2341
2342
2343
```

```
                        ;*LAST REVISION 04-APR-81
677
678                     .TITLE  CZRMPB0 RM05/3/2 DSKLS TST 1
                        ;*COPYRIGHT (C) 1981
                        ;*DIGITAL EQUIPMENT CORPORATION
                        ;*COLORADO SPGS., CO. 80919
                        ;*
                        ;*PROGRAM BY MIKE LEAVITT
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C5), 18-MAR-81
                        ;*
679                     .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
                        ;*         8            LOOP ON TEST IN SWR<7:0>
680                     ;*         7            TN128
                        ;*         6            TN64
                        ;*         5            TN32
                        ;*         4            TN16
                        ;*         3            TN8
                        ;*         2            TN4
                        ;*         1            TN2
                        ;*         0            TN1
68
682                     .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                  STACK   = 1100
104000                  ERROR   = EMT           ;;BASIC DEFINITION OF ERROR CALL
000004                  SCOPE   = IOT           ;;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
177776                  PSW = PS
177774                  STKLMT  = 177774        ;;STACK LIMIT REGISTER
177772                  PIRQ    = 177772        ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR    = 177570        ;;HARDWARE SWITCH REGISTER
177570                  DDISP   = 177570        ;;HARDWARE DISPLAY REGISTER

                        ;*GENERAL PURPOSE REGISTER DEFINITIONS
000000                  R0      = %0            ;;GENERAL REGISTER
000001                  R1      = %1            ;;GENERAL REGISTER
000002                  R2      = %2            ;;GENERAL REGISTER
000003                  R3      = %3            ;;GENERAL REGISTER
000004                  R4      = %4            ;;GENERAL REGISTER
```

```
000005                  R5      = %5            ;;GENERAL REGISTER
000006                  R6      = %6            ;;GENERAL REGISTER
000007                  R7      = %7            ;;GENERAL REGISTER
000006                  SP      = %6            ;;STACK POINTER
000007                  PC      = %7            ;;PROGRAM COUNTER

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

                        ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
100000                  SW15    = 100000
040000                  SW14    = 40000
020000                  SW13    = 20000
010000                  SW12    = 10000
004000                  SW11    = 4000
002000                  SW10    = 2000
001000                  SW09    = 1000
000400                  SW08    = 400
000200                  SW07    = 200
000100                  SW06    = 100
000040                  SW05    = 40
000020                  SW04    = 20
000010                  SW03    = 10
000004                  SW02    = 4
000002                  SW01    = 2
000001                  SW00    = 1
001000                  SW9=SW09
000400                  SW8=SW08
000200                  SW7=SW07
000100                  SW6=SW06
000040                  SW5=SW05
000020                  SW4=SW04
000010                  SW3=SW03
000004                  SW2=SW02
000002                  SW1=SW01
000001                  SW0=SW00

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15   = 100000
040000                  BIT14   = 40000
020000                  BIT13   = 20000
010000                  BIT12   = 10000
004000                  BIT11   = 4000
002000                  BIT10   = 2000
001000                  BIT09   = 1000
000400                  BIT08   = 400
000200                  BIT07   = 200
000100                  BIT06   = 100
000040                  BIT05   = 40
000020                  BIT04   = 20
```

```
                000010          BIT03   = 10
                000004          BIT02   = 4
                000002          BIT01   = 2
                000001          BIT00   = 1
                001000          BIT9=BIT09
                000400          BIT8=BIT08
                000200          BIT7=BIT07
                000100          BIT6=BIT06
                000040          BIT5=BIT05
                000020          BIT4=BIT04
                000010          BIT3=BIT03
                000004          BIT2=BIT02
                000002          BIT1=BIT01
                000001          BIT0=BIT00

                                ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
                000004          ERRVEC  = 4             ;;TIME OUT AND OTHER ERRORS
                000010          RESVEC  = 10            ;;RESERVED AND ILLEGAL INSTRUCTIONS
                000014          TBITVEC = 14            ;;''T'' BIT
                000014          TRTVEC  = 14            ;;TRACE TRAP
                000014          BPTVEC  = 14            ;;BREAKPOINT TRAP (BPT)
                000020          IOTVEC  = 20            ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                000024          PWRVEC  = 24            ;;POWER FAIL
                000030          EMTVEC  = 30            ;;EMULATOR TRAP (EMT) **ERROR**
                000034          TRAPVEC = 34            ;;''TRAP'' TRAP
                000060          TKVEC   = 60            ;;TTY KEYBOARD VECTOR
                000064          TPVEC   = 64            ;;TTY PRINTER VECTOR
                000240          PIRQVEC = 240           ;;PROGRAM INTERRUPT REQUEST VECTOR
    683
    684                         .SBTTL  RM REGISTER BIT DEFINITIONS
    685
    686                         ;*RMCS1 CONTROL STATUS REGISTER
    687
    688         004000          DVA     = BIT11                 ;DEVICE AVAILABLE-READ ONLY
    689         000040          F4      = BIT05                 ;FUNCTION CODE
    690         000020          F3      = BIT04                 ;FUNCTION CODE
    691         000010          F2      = BIT03                 ;FUNCTION CODE
    692         000004          F1      = BIT02                 ;FUNCTION CODE
    693         000002          F0      = BIT01                 ;FUNCTION CODE
    694         000001          GO      = BIT00                 ;GO BIT
    695         000077          FNCMSK  = 000077                ;FUNCTION CODE MASK
    696
    697                         ;FUNCTION CODES (BITS 01-05 OF RMCS1)
    698         000000          NOP     = 000000                ;NOP COMMAND
    699         000002          ILF02   = 000002                ;ILLEGAL COMMAND
    700         000004          SEEK    = 000004                ;SEEK COMMAND
    701         000006          RECAL   = 000006                ;RECALIBRATE COMMAND
    702         000010          DRVCLR  = 000010                ;DRIVE CLEAR COMMAND
    703         000012          RLEASE  = 000012                ;RELEASE COMMAND
    704         000014          OFFSET  = 000014                ;OFFSET COMMAND
    705         000016          RTC     = 000016                ;RETURN TO CENTERLINE COMMAND
    706         000020          RIP     = 000020                ;READ IN PRESET COMMAND
    707         000022          PAKACK  = 000022                ;PACK ACKNOWLEDGE COMMAND
    708         000022          PACACK  = PAKACK
    709         000024          ILF24   = 000024                ;ILLEGAL COMMAND
    710         000026          ILF26   = 000026                ;ILLEGAL COMMAND
    711         000030          SEARCH  = 000030                ;SEARCH COMMAND
```

```
714  000030          ILF30   - 000030          ;ILLEGAL COMMAND
     000032          ILF32   = 000032          ;ILLEGAL COMMAND
     000034          ILF34   - 000034          ;ILLEGAL COMMAND
     000036          ILF36   = 000036          ;ILLEGAL COMMAND
     000040          ILF40   = 000040          ;ILLEGAL COMMAND
     000042          ILF42   - 000042          ;ILLEGAL COMMAND
     000044          ILF44   = 000044          ;ILLEGAL COMMAND
     000046          ILF46   = 000046          ;ILLEGAL COMMAND
715  000050          WCD     - 000050          ;WRITE CHECK DATA COMMAND
716  000052          WCH     = 000052          ;WRITE CHECK HEADER AND DATA
717  000054          ILF54   - 000054          ;ILLEGAL COMMAND
718  000056          ILF56     000056          ;ILLEGAL COMMAND
719  000060          WD      = 000060          ;WRITE DATA COMMAND
720  000062          WH      = 000062          ;WRITE HEADER AND DATA COMMAND
721  000064          ILF64   = 000064          ;ILLEGAL COMMAND
722  000066          ILF66   = 000066          ;ILLEGAL COMMAND
723  000070          RD      = 000070          ;READ DATA COMMAND
724  000072          RH      - 000072          ;READ HEADER AND DATA COMMAND
725  000074          ILF74   = 000074          ;ILLEGAL COMMAND
726  000076          ILF76   = 000076          ;ILLEGAL COMMAND
727
728                  ;*RMDA  DISK ADDRESS REGISTER
729
730                  ;TRACK ADDRESS DEFINITIONS
731  010000          TA16    = BIT12           ;TRACK ADDRESS 16.
732  004000          TA8     - BIT11           ;TRACK ADDRESS 8.
733  002000          TA4     - BIT10           ;TRACK ADDRESS 4
734  001000          TA2     = BIT09           ;TRACK ADDRESS 2
735  000400          TA1     = BIT08           ;TRACK ADDRESS 1
736
737                  ;SECTOR ADDRESS DEFINITIONS
738  000020          SA16    = BIT04           ;SECTOR ADDRESS 16.
739  000010          SA8     = BIT03           ;SECTOR ADDRESS 8.
740  000004          SA4     = BIT02           ;SECTOR ADDRESS 4
741  000002          SA2     - BIT01           ;SECTOR ADDRESS 2
742  000001          SA1     = BIT00           ;SECTOR ADDRESS 1
743
744                  ;TRACK & SECTOR MASKS
745  177400          TADMSK  = 177400          ;TRACK ADDRESS MASK
746  000377          SADMSK  - 000377          ;SECTOR ADDRFSS MASK
747
748                  ;*RMDS  DRIVE STATUS REGISTER
749
750  100000          ATA       BIT15           ;ATTENTION ACTIVE
751  040000          ERR     - BIT14           ;COMPOSITE ERROR
752  020000          PIP       BIT13           ;POSITIONING IN PROGRESS
753  010000          MOL       BIT12           ;MEDIUM ON LINE
754  004000          WRL     = BIT11           ;WRITE LOCK
755  002000          LBT     = BIT10           ;LAST BLOCK TRANSFERRED
756  001000          PGM     - BIT09           ;PROGRAMMABLE
757  000400          DPR     = BIT08           ;DRIVE PRESENT
758  000200          DRY       BIT07           ;DRIVE READY
759  000100          VV      - BIT06           ;VOLUME VALID
760  000001          OM      - BIT00           ;OFFSET MODE ACTIVE
761
762                  ;*RMER1 ERROR REGISTER #1
763
```

```
        100000                  DCK     = BIT15                 ;DATA CHECK ERROR
        040000                  UNS     = BIT14                 ;DRIVE UNSAFE
        020000                  OPI     = BIT13                 ;OPERATION INCOMPLETE
        010000                  DTE     = BIT12                 ;DRIVE TIMING ERROR
        004000                  WLE     = BIT11                 ;WRITE LOCK ERROR
        002000                  IAE     = BIT10                 ;INVALID ADDRESS ERROR
        001000                  AOE     = BIT09                 ;ADDRESS OVERFLOW ERROR
        000400                  HCRC    = BIT08                 ;HEADER CRC ERROR
        000200                  HCE     = BIT07                 ;HEADER COMPARE ERROR
        000100                  ECH     = BIT06                 ;ECC ''HARD'' ERROR
        000040                  WCF     = BIT05                 ;WRITE CLOCK FAILURE
        000020                  FER     = BIT04                 ;FORMAT ERROR
        000010                  PAR     = BIT03                 ;PARITY ERROR
        000004                  RMR     = BIT02                 ;REGISTER MODIFICATION REFUSED
        000002                  ILR     = BIT01                 ;ILLEGAL REGISTER
        000001                  ILF     = BIT00                 ;ILLEGAL FUNCTION

        115760                  NDTMSK  = DCK!DTE!WLE!AOE!HCRC!HCE!ECH.WCF!FER
                                ;''NDTMSK'' IS USED TO MASK ERROR REGISTER 1 DURING NON - DATA
                                ;COMMANDS, I.E., HOUSEKEEPING AND POSITIONING COMMANDS

                                ;*RMAS  ATTENTION SUMMARY REGISTER

        000377                  ATNMSK  = 377                   ;MASK FOR ATTENTION BITS

                                ;*RMLA  LOOK AHEAD REGISTER

        002000                  SC4     = BIT10                 ;SECTOR COUNT = 16
        001000                  SC3     = BIT09                 ;SECTOR COUNT = 8
        000400                  SC2     = BIT08                 ;SECTOR COUNT = 4
        000200                  SC1     = BIT07                 ;SECTOR COUNT = 2
        000100                  SC0     = BIT06                 ;SECTOR COUNT = 1

        003700                  SCTMSK  = 003700                ;SECTOR COUNT MASK

                                ;*RMMR1 MAINTENANCE REGISTER #1

                                ;WRITE ONLY BITS
        100000                  DBCK    = BIT15                 ;DEBUG CLOCK
        040000                  DBEN    = BIT14                 ;DEBUG CLOCK ENABLE
        020000                  DEBL    = BIT13                 ;DIAGNOSTIC END OF BLOCK
        010000                  MSEN    = BIT12                 ;SEARCH TIMEOUT ENABLE
        004000                  MCLK    = BIT11                 ;MAINTENANCE CLOCK
        002000                  MRD     = BIT10                 ;READ DATA
        001000                  MUR     = BIT09                 ;UNIT READY
        000400                  MOC     = BIT08                 ;ON CYLINDER
        000200                  MSER    = BIT07                 ;SEEK ERROR
        000100                  MDF     = BIT06                 ;DRIVE FAULT
        000040                  MS      = BIT05                 ;SECTOR PULSE
        000010                  MWP     = BIT03                 ;WRITE PROTECT
        000004                  MI      = BIT02                 ;INDEX PULSE
        000002                  MSC     = BIT01                 ;SECTOR COMPARE
        000001                  DMD     = BIT00                 ;DIAGNOSTIC MODE

                                ;READ ONLY BITS
        100000                  OCC     = BIT15                 ;OCCUPIED
        040000                  RG      = BIT14                 ;RUN AND GO
```

```
821     020000          EBL     = BIT13         ;END OF BLOCK
822     010000          REX     - BIT12         ;EXCEPTION
823     004000          ESRC    - BIT11         ;ENABLE SEARCH
824     002000          PLFS    = BIT10         ;LOOKING FOR SYNC
825     001000          ECRC    = BIT09         ;ENABLE CRC OUT
826     000400          PDA     - BIT08         ;DATA AREA
827     000200          PHA     - BIT07         ;HEADER AREA
828     000100          CONT    - BIT06         ;CONTINUE
829     000040          WC      - BIT05         ;WORD CLOCK
830     000020          EECC    = BIT04         ;ENABLE ECC OUT
831     000010          MWD     = BIT03         ;WRITE DATA BIT
832     000004          LS      = BIT02         ;LAST SECTOR
833     000002          LST     = BIT01         ;LAST SECTOR AND TRACK
834     000001          DMD     = BIT00         ;DIAGNOSTIC MODE
835     051401          MR1AAA  - DMD!MUR!DBEN!MOC.MSEN
836
837                     ;*RMDT  DRIVE TYPE REGISTER
838
839     100000          NSA     - BIT15         ;NOT SECTOR ADDRESSED = 0
840     040000          TAP     = BIT14         ;TAPE DRIVE = 0
841     020000          MOH     - BIT13         ;MOVING HEAD = 1
842     004000          DRQ     = BIT11         ;DRIVE REQUEST REQUIRED
843
844     020024          SNGPRT  = 020024        ;SINGLE PORT DRIVE TYPE
845     024024          DULPRT  = 024024        ;DUAL PORT DRIVE TYPE
846
847                     ;*RMOF  OFFSET REGISTER
848
849     010000          FMT16   = BIT12         ;16 BIT WORD FORMAT
850     004000          ECI     = BIT11         ;ECC INHIBIT
851     002000          HCI     = BIT10         ;HEADER COMPARE INHIBIT
852     000200          OFD     = BIT07         ;OFFSET FORWARD
853     161577          XNUOF   - 161577        ;UNSED BITS OF RMOF
854
855                     ;*RMDC  DESIRED CYLINDER ADDRESS REGISTER
856
857     001777          CYLMSK  = 001777        ;MASK FOR CYLINDER ADDRESS
858     176000          XNUDC   = 176000        ;UNSED BITS OF RMDC
859
860                     ;*RMMR2 MAINTENANCE REGISTER #2
861
862                     ;READ ONLY BITS
863     100000          RQA     = BIT15         ;PORT A REQUEST
864     040000          RQB     - BIT14         ;PORT B REQUEST
865     020000          TAG     = BIT13         ;TAG CONTROL
866     010000          TST     = BIT12         ;COMMAND SEQUENCE TEST BIT
867     004000          CC      - BIT11         ;CONTROL OR CYLINDER TAG
868     002000          CH      = BIT10         ;CONTROL OR HEAD TAG
871     001000          BB09    = BIT09         ;TAG BUS
        000400          BB08    - BIT08         ;TAG BUS
        000200          BB07    - BIT07         ;TAG BUS
        000100          BB06    = BIT06         ;TAG BUS
        000040          BB05    = BIT05         ;TAG BUS
        000020          BB04    - BIT04         ;TAG BUS
        000010          BB03    = BIT03         ;TAG BUS
        000004          BB02    = BIT02         ;TAG BUS
        000002          BB01    = BIT01         ;TAG BUS
```

```
                000001          BB00    = BIT00                 ;TAG BUS
872
873                             ;*RMER2 ERROR REGISTER 2
874
875             100000          BSE     = BIT15                 ;BAD SECTOR ERROR
876             040000          SKI     = BIT14                 ;SEEK INCOMPLETE
877             020000          OPE     = BIT13                 ;OPERATOR PLUG ERROR
878             010000          IVC     = BIT12                 ;INVALID COMMAND ERROR
879             004000          LSC     = BIT11                 ;LOSS OF SYSTEM CLOCK
880             002000          LBC     = BIT10                 ;LOSS OF BIT CLOCK
881             000200          DVC     = BIT07                 ;DEVICE CHECK
882             000010          DPE     = BIT03                 ;DATA PARITY ERROR
883             001567          XNUER2  = 001567                ;UNSED BITS OF RMER2
884
885                             .SBTTL  PROGRAM MNEMONICS
886
887             100000          MSE     = BIT15                 ;MANUFACTURING DETECTED SECTOR ERROR
888             040000          USE     = BIT14                 ;USER DETECTED SECTOR ERROR
889
890                             .SBTTL  RM REGISTER INDEX VALUES
891
892             000000          RMCS1   = 00                    ;CONTROL STATUS REGISTER #1
893             000006          RMDA    = 06                    ;DISK ADDRESS REGISTER
894             000012          RMDS    = 12                    ;DRIVE STATUS REGISTER
895             000014          RMER1   = 14                    ;ERROR REGISTER #1
896             000016          RMAS    = 16                    ;ATTENTION SUMMARY REGISTER
897             000020          RMLA    = 20                    ;LOOK AHEAD REGISTER
898             000024          RMMR1   = 24                    ;MAINTENANCE REGISTER
899             000026          RMDT    = 26                    ;DRIVE TYPE REGISTER
900             000030          RMSN    = 30                    ;SERIAL NUMBER REGISTER
901             000032          RMOF    = 32                    ;OFFSET REGISTER
902             000034          RMDC    = 34                    ;DESIRED CYLINDER REGISTER
903             000036          RMHR    = 36                    ;HOLDING REGISTER
904             000040          RMMR2   = 40                    ;MAINTENANCE REGISTER #2
905             000042          RMER2   = 42                    ;ERROR REGISTER #2
906             000044          RMEC1   = 44                    ;ECC POSITION REGISTER
907             000046          RMEC2   = 46                    ;ECC PATTERN REGISTER
910             000050          ILRG50  = 50                    ;ILLEGAL REGISTER 50
                000052          ILRG52  = 52                    ;ILLEGAL REGISTER 52
                000054          ILRG54  = 54                    ;ILLEGAL REGISTER 54
                000056          ILRG56  = 56                    ;ILLEGAL REGISTER 56
                000060          ILRG60  = 60                    ;ILLEGAL REGISTER 60
                000062          ILRG62  = 62                    ;ILLEGAL REGISTER 62
                000064          ILRG64  = 64                    ;ILLEGAL REGISTER 64
                000066          ILRG66  = 66                    ;ILLEGAL REGISTER 66
                000070          ILRG70  = 70                    ;ILLEGAL REGISTER 70
                000072          ILRG72  = 72                    ;ILLEGAL REGISTER 72
                000074          ILRG74  = 74                    ;ILLEGAL REGISTER 74
                000076          ILRG76  = 76                    ;ILLEGAL REGISTER 76
911
912
913             000077          IDXMSK  = 77                    ;MASK FOR REGISTER INDEX NUMBER
914
915                             .SBTTL  RH CONTROLLER REGISTER BIT DEFINITIONS
916
917                             ;*RMCS1 CONTROL STATUS REGISTER #1
918
```

```
                100000          SC      = BIT15                 ;SPECIAL CONDITION-READ ONLY
                040000          TRE     = BIT14                 ;TRANSFER ERROR
                020000          MCPE    = BIT13                 ;MASSBUS CONTROL BUS PARITY ERROR-READ ONLY
                002000          PSEL    = BIT10                 ;PORT B SELECT
                001000          A17     = BIT09                 ;ADDRESS EXTENSION
                000400          A16     = BIT08                 ;ADDRESS EXTENSION
                000200          RDY     = BIT07                 ;READY-READ ONLY
                000100          IE      = BIT06                 ;INTERRUPT ENABLE

                                ;*RMCS2 RH CONTROL STATUS REGISTER #2

                100000          DLT     = BIT15                 ;DATA LATE-READ ONLY
                040000          WCE     = BIT14                 ;WRITE CHECK ERROR-READ ONLY
                020000          UPE     = BIT13                 ;UNIBUS PARITY ERROR
                010000          NED     = BIT12                 ;NONEXISTANT DRIVE-READ ONLY
                004000          NEM     = BIT11                 ;NONEXISTANT MEMORY-READ ONLY
                002000          PGE     = BIT10                 ;PROGRAM ERROR-READ ONLY
                001000          MXF     = BIT09                 ;MISSED TRANSFER
                000400          MDPE    = BIT08                 ;MASSBUS DATA BUS PARITY ERROR-READ ONLY
                000200          OR      = BIT07                 ;OUTPUT READY-READ ONLY
                000100          IR      = BIT06                 ;INPUT READY-READ ONLY
                000040          CLR     = BIT05                 ;CONTROLLER CLEAR
                000020          PAT     = BIT04                 ;PARITY TEST
                000010          BAI     = BIT03                 ;UNIBUS ADDRESS INCREMENT INHIBIT
                000004          U2      = BIT02                 ;UNIT SELECT
                000002          U1      = BIT01                 ;UNIT SELECT
                000001          U0      = BIT00                 ;UNIT SELECT

                                ;UNIT SELECT MASK

                000007          UNTMSK  = 7                     ;UNIT SELECT MASK

                                ;*RMCS3 RH70 CONTROL STATUS REGISTER #3

                100000          APE     = BIT15                 ;ADDRESS PARITY ERROR
                040000          DPEHI   = BIT14                 ;DATA PARITY ERROR HIGH WORD
                020000          DPELO   = BIT13                 ;DATA PARITY ERROR LOW WORD
                010000          WCEHI   = BIT12                 ;WHITE CHECK ERROR HIGH WORD
                004000          WCELO   = BIT11                 ;WRITE CHECK ERROR LOW WORD
                002000          DBL     = BIT10                 ;DOUBLE WORD TRANSFER
                000100          IE      = BIT06                 ;INTERRUPT ENABLE
                000010          IPCK3   = BIT03                 ;INVERT PARITY CHECK
                000004          IPCK2   = BIT02                 ;INVERT PARITY CHECK
                000002          IPCK1   = BIT01                 ;INVERT PARITY CHECK
                000001          IPCK0   = BIT00                 ;INVERT PARITY CHECK

                                .SBTTL  RH CONTROLLER REGISTER INDEX VALUES

                000000          RMCS1   = 00                    ;CONTROL, STATUS REGISTER #1
                000002          RMWC    = 02                    ;WORD COUNT REGISTER
                000004          RMBA    = 04                    ;BUS ADDRESS REGISTER
                000010          RMCS2   = 10                    ;CONTROL, STATUS REGISTER #2
                000022          RMDB    = 22                    ;DATA BUFFER
                000050          RMBAE   = 50                    ;BUS ADDRESS EXTENSION
                000052          RMCS3   = 52                    ;CONTROL, STATUS REGISTER #3

                176700          ABASE   = 176700                ;UNIBUS ADDRESS
```

```
976          120254                  AVECT1  = 120254              ;UNIBUS VECTOR ADDRESS AND PRIORITY
977
```

```
                        .SBTTL  TRAP CATCHER

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

                        .SBTTL  STARTING ADDRESS(ES)

000200  000137  004652          JMP     @#START         ;;JUMP TO STARTING ADDRESS OF PROGRAM
000204  000137  004642          JMP     @#START1        ;CHANGE RH/RM BUS ADDRESS

                        .SBTTL  ACT11 HOOKS

                        ;;*****************************************************************
                        ;HOOKS REQUIRED BY ACT11
                                $SVPC=.                 ;SAVE PC
                                .-46
000046  054020                  $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
                                .-52
000052  000000                  .WORD   0               ;;2)SET LOC.52 TO ZERO
                                .-$SVPC                 ;; RESTORE PC

                                .-1100
                        .SBTTL  APT PARAMETER BLOCK

                        ;;*****************************************************************
                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                        ;;*****************************************************************
                                .$X=.   ;;SAVE CURRENT LOCATION
                                .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                  200     ;;FOR APT START UP
                                .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001100                  $APTHDR ;;POINT TO APT HEADER BLOCK
                                .=.$X   ;;RESET LOCATION COUNTER
                        ;;*****************************************************************
                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                        ;INTERFACE SPEC.

001100                  $APTHD:
001100  000000          $HIBTS: .WORD   0       ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001102  001222          $MBADR: .WORD   $MAIL   ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001104  000024          $TSTM:  .WORD   20.     ;;RUN TIM OF LONGEST TEST
001106  000024          $PASTM: .WORD   20.     ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001110  000024          $UNITM: .WORD   20.     ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDED UNIT
001112  000042                  .WORD   $ETEND-$MAIL/2  ;;LENGTH MAILBOX-ETABLE(WORDS)
                        TAGADR=.
```

```
      0                         .SBTTL  COMMON TAGS

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

            001114                      .-TAGADR
      001114                    $CMTAG:                         ;;START OF COMMON TAGS
      001114  000000                    .WORD   0
      001116     000            $TSTNM: .BYTE   0               ;;CONTAINS THE TEST NUMBER
      001117     000            $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
      001120  000000            $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
      001122  000000            $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
      001124  000000            $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
      001126  000000            $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
      001130     000            $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
      001131     001            $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
      001132  000000            $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
      001134  000000            $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
      001136  000000            $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
      001140  000000            $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
      001142  000000            $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
      001144  000000                    .WORD   0               ;;RESERVED--NOT TO BE USED
      001146  000000                    .WORD   0
      001150     000            $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
      001151     000            $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
      001152  000000                    .WORD   0
      001154  177570            SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
      001156  177570            DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
      001160  177560            $TKS:   177560                  ;;TTY KBD STATUS
      001162  177562            $TKB:   177562                  ;;TTY KBD BUFFER
      001164  177564            $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
      001166  177566            $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
      001170     000            $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
      001171     002            $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
      001172     012            $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED'
      001173     000            $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE' FLAG (BIT<07> 0-YES)
      001174  000000            $TMP0:  .WORD   0               ;;USER DEFINED
      001176  000000            $TMP1:  .WORD   0               ;;USER DEFINED
      001200  000000            $TMP2:  .WORD   0               ;;USER DEFINED
      001202  000000            $TMP3:  .WORD   0               ;;USER DEFINED
      001204  000000            $TMP4:  .WORD   0               ;;USER DEFINED
      001206  000000            $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
      001210  000000            $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
      001212     207     377 377 $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
      001216     077            $QUES:  .ASCII  /?/             ;;QUESTION MARK
      001217     015            $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
      001220     012     000    $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .EVEN
      001222                    $MAIL:                          ;;APT MAILBOX
      001222  000000            $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
      001224  000000            $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
      001226  000000            $TESTN: .WORD   ATESTN          ;;TEST NUMBER
```

```
001230  000000                  $PASS:  .WORD   APASS   ;;PASS COUNT
001232  000000                  $DEVCT: .WORD   ADEVCT  ;;DEVICE COUNT
001234  000000                  $UNIT:  .WORD   AUNIT   ;;I/O UNIT NUMBER
001236  000000                  $MSGAD: .WORD   AMSGAD  ;;MESSAGE ADDRESS
001240  000000                  $MSGLG: .WORD   AMSGLG  ;;MESSAGE LENGTH
001242                          $ETABLE:                ;;APT ENVIRONMENT TABLE
001242     000                  $ENV:   .BYTE   AENV    ;;ENVIRONMENT BYTE
001243     000                  $ENVM:  .BYTE   AENVM   ;;ENVIRONMENT MODE BITS
001244  000000                  $SWREG: .WORD   ASWREG  ;;APT SWITCH REGISTER
001246  000000                  $USWR:  .WORD   AUSWR   ;;USER SWITCHES
001250  000000                  $CPUOP: .WORD   ACPUOP  ;;CPU TYPE,OPTIONS
                                ;*                      BITS 15-11=CPU TYPE
                                ;*                              11/04=01,11/05-02,11/20=03,11/40=04,11/45=05
                                ;*                              11/70=06,PDQ=07,Q=10
                                ;*                      BIT 10=REAL TIME CLOCK
                                ;*                      BIT  9=FLOATING POINT PROCESSOR
                                ;*                      BIT  8=MEMORY MANAGEMENT
001252     000                  $MAMS1: .BYTE   AMAMS1  ;;HIGH ADDRESS,M.S. BYTE
001253     000                  $MTYP1: .BYTE   AMTYP1  ;;MEM. TYPE,BLK#1
                                ;*                      MEM.TYPE BYTE   --  (HIGH BYTE)
                                ;*                              900 NSEC CORE=001
                                ;*                              300 NSEC BIPOLAR=002
                                ;*                              500 NSEC MOS=003
001254  000000                  $MADR1: .WORD   AMADR1  ;;HIGH ADDRESS,BLK#1
                                ;*                        MEM.LAST ADDR.=3 BYTES,THIS WORD AND LOW OF ''TYPE'' ABOVE
001256     000                  $MAMS2: .BYTE   AMAMS2  ;;HIGH ADDRESS,M.S. BYTE
001257     000                  $MTYP2: .BYTE   AMTYP2  ;;MEM.TYPE,BLK#2
001260  000000                  $MADR2: .WORD   AMADR2  ;;MEM.LAST ADDRESS,BLK#2
001262     000                  $MAMS3: .BYTE   AMAMS3  ;;HIGH ADDRESS,M.S.BYTE
001263     000                  $MTYP3: .BYTE   AMTYP3  ;;MEM.TYPE,BLK#3
001264  000000                  $MADR3: .WORD   AMADR3  ;;MEM.LAST ADDRESS,BLK#3
001266     000                  $MAMS4: .BYTE   AMAMS4  ;;HIGH ADDRESS,M.S.BYTE
001267     000                  $MTYP4: .BYTE   AMTYP4  ;;MEM.TYPE,BLK#4
001270  000000                  $MADR4: .WORD   AMADR4  ;;MEM.LAST ADDRESS,BLK#4
001272  120254                  $VECT1: .WORD   AVECT1  ;;INTERRUPT VECTOR#1,BUS PRIORITY#1
001274  000000                  $VECT2: .WORD   AVECT2  ;;INTERRUPT VECTOR#2BUS PRIORITY#2
001276  176700                  $BASE:  .WORD   ABASE   ;;BASE ADDRESS OF EQUIPMENT UNDER TEST
001300  000000                  $DEVM:  .WORD   ADEVM   ;;DEVICE MAP
001302  000000                  $CDW1:  .WORD   ACDW1   ;;CONTROLLER DESCRIPTION WORD#1
001304  000000                  $CDW2:  .WORD   ACDW2   ;;CONTROLLER DESCRIPTION WORD#2
001306  000000                  $DDW0:  .WORD   ADDW0   ;;DEVICE DESCRIPTOR WORD#0
001310  000000                  $DDW1:  .WORD   ADDW1   ;;DEVICE DESCRIPTOR WORD#1
001312  000000                  $DDW2:  .WORD   ADDW2   ;;DEVICE DESCRIPTOR WORD#2
001314  000000                  $DDW3:  .WORD   ADDW3   ;;DEVICE DESCRIPTOR WORD#3
001316  000000                  $DDW4:  .WORD   ADDW4   ;;DEVICE DESCRIPTOR WORD#4
001320  000000                  $DDW5:  .WORD   ADDW5   ;;DEVICE DESCRIPTOR WORD#5
001322  000000                  $DDW6:  .WORD   ADDW6   ;;DEVICE DESCRIPTOR WORD#6
001324  000000                  $DDW7:  .WORD   ADDW7   ;;DEVICE DESCRIPTOR WORD#7
001326                          $ETEND:
                                .MEXIT
```

```
                                .SBTTL  USER DEFINED TAGS

001326  000000  AUTSIZ: .WORD   0       ;ALLOW AUTO DRIVE SIZING - 0, USE MANUALLY INPUT DRIVES   1
001330  000000  CHGADR: .WORD   0       ;CHANGE RH/RM BUS ADDRESS = -1, NO CHANGE = 0
001332  000000  XXDP:   .WORD   0       ;THE LOW BYTE CONTAINS THE DRIVE NUMBER FROM WHICH
                                        ;THE PROGRAM WAS LOADED. THE HIGH BYTE CONTAINS THE
                                        ;'XXDP' DEVICE CODE FOR THE RM05/3/2.
001334      000 LSTRK:  .BYTE   0       ;LO BYTE - 0
001335      000         .BYTE   0       ;HI BYTE, CONTAINS LAST TRACK ADDRESS OF UNIT
                                        ;UNDER TEST. RM02/3 = 4., RM05 = 18.

                                ;THE REGISTER INPUT BUFFER IS USED FOR
                                ;STORING DRIVE STATUS

001336                          GETBUF:

                                ;REGISTER INPUT BUFFER
001336  000000  RMCS1I: .WORD   0       ;CONTROL, STATUS REGISTER #1
001340  000000  RMWCI:  .WORD   0       ;WORD COUNT REGISTER
001342  000000  RMBAI:  .WORD   0       ;BUS ADDRESS REGISTER
001344  000000  RMDAI:  .WORD   0       ;DISK ADDRESS REGISTER
001346  000000  RMCS2I: .WORD   0       ;CONTROL, STATUS REGISTER #2
001350  000000  RMDSI:  .WORD   0       ;DRIVE STATUS REGISTER
001352  000000  RMER1I: .WORD   0       ;ERROR REGISTER #1
001354  000000  RMASI:  .WORD   0       ;ATTENTION SUMMARY REGISTER
001356  000000  RMLAI:  .WORD   0       ;LOOK AHEAD REGISTER
001360  000000  RMDBI:  .WORD   0       ;DATA BUFFER
001362  000000  RMMR1I: .WORD   0       ;MAINTENANCE REGISTER #1
001364  000000  RMDTI:  .WORD   0       ;DRIVE TYPE REGISTER
001366  000000  RMSNI:  .WORD   0       ;SERIAL NUMBER REGISTER
001370  000000  RMOFI:  .WORD   0       ;OFFSET REGISTER
001372  000000  RMDCI:  .WORD   0       ;DESIRED CYLINDER REGISTER
001374  000000  RMHRI:  .WORD   0       ;HOLDING REGISTER
001376  000000  RMMR2I: .WORD   0       ;MAINTENANCE REGISTER #2
001400  000000  RMER2I: .WORD   0       ;ERROR REGISTER #2
001402  000000  RMEC1I: .WORD   0       ;ECC POSITION REGISTER
001404  000000  RMEC2I: .WORD   0       ;ECC PATTERN REGISTER
001406  000000  RMBAEI: .WORD   0       ;BUS ADDRESS EXTENSION REGISTER
001410  000000  RMCS3I: .WORD   0       ;CONTROL, STATUS REGISTER #3

                                ;THE REGISTER OUTPUT BUFFER IS USED FOR
                                ;ASSEMBLING DATA GOING TO REGISTER

001412                          PUTBUF:

                                ;REGISTER OUTPUT BUFFER
001412  000000  RMCS1O: .WORD   0       ;CONTROL, STATUS REGISTER #1
001414  000000  RMWCO:  .WORD   0       ;WORD COUNT REGISTER
001416  000000  RMBAO:  .WORD   0       ;BUS ADDRESS REGISTER
001420  000000  RMDAO:  .WORD   0       ;DISK ADDRESS REGISTER
001422  000000  RMCS2O: .WORD   0       ;CONTROL, STATUS REGISTER #2
001424  000000  RMDSO:  .WORD   0       ;DRIVE STATUS REGISTER
001426  000000  RMER1O: .WORD   0       ;ERROR REGISTER #1
001430  000000  RMASO:  .WORD   0       ;ATTENTION SUMMARY REGISTER
001432  000000  RMLAO:  .WORD   0       ;LOOK AHEAD REGISTER
001434  000000  RMDBO:  .WORD   0       ;DATA BUFFER
001436  000000  RMMR1O: .WORD   0       ;MAINTENANCE REGISTER #1
```

```
001440  000000                  RMDTO:  .WORD   0       ;DRIVE TYPE REGISTER
001442  000000                  RMSNO:  .WORD   0       ;SERIAL NUMBER REGISTER
001444  000000                  RMOFO:  .WORD   0       ;OFFSET REGISTER
001446  000000                  RMDCO:  .WORD   0       ;DESIRED CYLINDER REGISTER
001450  000000                  RMHRO:  .WORD   0       ;HOLDING REGISTER
001452  000000                  RMMR2O: .WORD   0       ;MAINTENANCE REGISTER #2
001454  000000                  RMER2O: .WORD   0       ;ERROR REGISTER #2
001456  000000                  RMEC1O: .WORD   0       ;ECC POSITION REGISTER
001460  000000                  RMEC2O: .WORD   0       ;ECC PATTERN REGISTER
001462  000000                  RMBAEO: .WORD   0       ;BUS ADDRESS EXTENSION REGISTER
001464  000000                  RMCS3O: .WORD   0       ;CONTROL, STATUS REGISTER #3

                                ;EACH WORD OF THE TEST QUE CONTAINS THE DEVICE NUMBER IN
                                ;THE LOW BYTE AND THE ATTENTION BIT IN THE HIGH BYTE. THE
                                ;FIRST WORD CONTAINS THE ADDRESS OF THE DEVICE UNDER TEST
                                ;IN THE TABLE. A ZERO WORD IS A BLANK AND REPRESENTS THE
                                ;END OF THE QUE.

001466  000000                  TSTQUE: .WORD   0               ;CONTAINS DEVICE POINTER
001470                                  .BLKW   8.              ;TEST QUE FOR DEVICES UNDER TEST
001510  000000                          .WORD   0               ;TABLE TERMINATOR GOES HERE WHEN
                                                                ;ALL 8. DEVICES ARE UNDER TEST.

001512  172540                  $LPCSR: .WORD   172540          ;KW11-P CONTROL + STATUS REGISTER
001514  172542                  $LPCSB: .WORD   172542          ;KW11-P COUNT SET BUFFER
001516  000104                  $LPVEC: .WORD   104             ;KW11-P INTERRUPT VECTOR
001520  000106                          .WORD   106
001522  177546                  $LLCSR: .WORD   177546          ;KW11-L CONTROL + STATUS REGISTER
001524  000100                  $LLVEC: .WORD   100             ;KW11-L INTERRUPT VECTOR
001526  000102                          .WORD   102
001530  000000                  $PSW:   .WORD                   ;STORAGE FOR PRIORITY
001532  000000                  TIME:   .WORD                   ;STORAGE FOR ELAPSED TIME
001534  000000                  WATCH:  .WORD                   ;STORAGE FOR REMAINING TIME
001536  000000                  CLOCK:  .WORD                   ;ADDRESS OF START CLOCK SUB
001540  000000                  STOPCL: .WORD                   ;ADDRESS OF STOP CLOCK SUB

                                ;PUT TAGS HERE
```

```
                                .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

001542                          $ERRTB:

                                ;ERROR  1       CANNOT CLEAR NED STATUS

001542  064154                          EMT1
001544  072032                          EHT1
001546  072132                          EDT1
001550  072160                          EFT1

                                ;ERROR  2       CANNOT READ OR WRITE ANY DEVICE REG WITHOUT NED

001552  064162                          EMT2
001554  072036                          EHT2
001556  072134                          EDT2
001560  072162                          EFT2

                                ;ERROR  3       CANNOT WRITE/READ ONES TO ANY DEVICE REGISTER

001562  064210                          EMT3
001564  000000                          0
001566  000000                          0
001570  000000                          0

                                ;ERROR  4       CANNOT CLEAR ANY DEVICE REGISTER BITS W/MASSBUS INIT

001572  064230                          EMT4
001574  000000                          0
001576  000000                          0
001600  000000                          0

                                ;ERROR  5       CANNOT WRITE/READ ZEROS TO ALL BIT POSITIONS

001602  064252                          EMT5
001604  072042                          EHT5
001606  072136                          EDT5
001610  072164                          EFT5

                                ;ERROR  6       CANNOT WRITE/READ ONES TO ALL BIT POSITIONS
```

ERROR POINTER TABLE

```
18
001612  064276                          EMT6
001614  072042                          EHT5
001616  072136                          EDT5
001620  072164                          EFT5

19
20                              ;ERROR  7       CANNOT WRITE/READ SHIFTING ONE BIT TO ALL BIT POSITIONS
21                              ;               OF DEVICE REGISTERS
22
001622  064320                          EMT7
001624  072046                          EHT7
001626  072136                          EDT5
001630  072164                          EFT5

23
24                              ;ERROR  10      REGISTER SELECT 1 APPEARS S-A-0
25
001632  064342                          EMT10
001634  000000                          0
001636  000000                          0
001640  000000                          0

26
27                              ;ERROR  11      REGISTER SELECT 1 APPEARS S-A-1
28
001642  064360                          EMT11
001644  000000                          0
001646  000000                          0
001650  000000                          0

29
30                              ;ERROR  12      REGISTER SELECT 2 APPEARS S-A-0
31
001652  064376                          EMT12
001654  000000                          0
001656  000000                          0
001660  000000                          0

32
33                              ;ERROR  13      REGISTER SELECT 2 APPEARS S-A-1
34
001662  064414                          EMT13
001664  000000                          0
001666  000000                          0
001670  000000                          0

35
36                              ;ERROR  14      REGISTER SELECT 4 APPEARS S-A-0
37
001672  064432                          EMT14
001674  000000                          0
001676  000000                          0
001700  000000                          0

38
39                              ;ERROR  15      REGISTER SELECT 4 APPEARS S-A-1
```

ERROR POINTER TABLE

```
 40
001702  064450                          EMT15
001704  000000                          0
001706  000000                          0
001710  000000                          0

 41
 42                             ;ERROR  16      REGISTER SELECT 8 APPEARS S-A-0
 43
001712  064466                          EMT16
001714  000000                          0
001716  000000                          0
001720  000000                          0

 44
 45                             ;ERROR  17      REGISTER SELECT 8 APPEARS S-A-1
 46
001722  064504                          EMT17
001724  000000                          0
001726  000000                          0
001730  000000                          0

 47
 48                             ;ERROR  20      CANT WRITE ZEROS RMDA
 49
001732  064522                          EMT20
001734  072032                          EHT1
001736  072132                          EDT1
001740  072160                          EFT1

 50
  5                             ;ERROR  21      CANT WRITE ONES RMDA
 52
001742  064542                          EMT21
001744  072032                          EHT1
001746  072132                          EDT1
001750  072160                          EFT1

 53
 54                             ;ERROR  22      BIT INTERFERENCE IN WRITING/READING RMDA
 55
001752  064562                          EMT22
001754  072032                          EHT1
001756  072132                          EDT1
001760  072160                          EFT1

 56
 57                             ;ERROR  23      CANT WRITE ZEROS RMCS1
 58
001762  064576                          EMT23
001764  072032                          EHT1
001766  072132                          EDT1
001770  072160                          EFT1

 59
 60                             ;ERROR  24      CANT WRITE ONES RMCS1
 61
```

ERROR POINTER TABLE

```
001772  064616                  EMT24
001774  072032                  EHT1
001776  072132                  EDT1
002000  072160                  EFT1

62
63                      ;ERROR  25      BIT INTERFERENCE IN WRITING/READING RMCS1
64
002002  064636                  EMT25
002004  072032                  EHT1
002006  072132                  EDT1
002010  072160                  EFT1

65
66                      ;ERROR  26      MBA CLR L IS STUCK ACTIVE
67
002012  064652                  EMT26
002014  000000                  0
002016  000000                  0
002020  000000                  0

68
69                      ;ERROR  27      CANNOT CLEAR RMER1-PAR,RMR,ILF,ILR
70
002022  064704                  EMT27
002024  072032                  EHT1
002026  072132                  EDT1
002030  072160                  EFT1

71
72                      ;ERROR  30      CANNOT CLEAR RMER1-DCK,IAE,AOE,HCRC,HCE,ECH,WCF,FER
73
002032  064716                  EMT30
002034  072032                  EHT1
002036  072132                  EDT1
002040  072160                  EFT1

74
75                      ;ERROR  31      CANNOT CLEAR RMER1-OPI,DTE
76
002042  064732                  EMT31
002044  072032                  EHT1
002046  072132                  EDT1
002050  072160                  EFT1

77
78                      ;ERROR  32      CANNOT WRITE 0 IN RMER1-PAR,RMR,ILF,ILR
79
002052  064746                  EMT32
002054  072032                  EHT1
002056  072132                  EDT1
002060  072160                  EFT1

80
81                      ;ERROR  33      CANNOT WRITE 0 IN RMER1-DCK,IAE,AOE,HCRC,HCE,ECH,WCF,FER
82
002062  064762                  EMT33
```

ERROR POINTER TABLE

```
002064  072032                          EHT1
002066  072132                          EDT1
002070  072160                          EFT1

 83
 84                             ;ERROR  34      CANNOT WRITE 0 IN RMER1-OPI,DTE
 85
002072  065000                          EMT34
002074  072032                          EHT1
002076  072132                          EDT1
002100  072160                          EFT1

 86
 87                             ;ERROR  35      CANNOT WRITE 1 IN RMER1
 88
002102  065016                          EMT35
002104  072032                          EHT1
002106  072132                          EDT1
002110  072160                          EFT1

 89
 90                             ;ERROR  36      CANNOT WRITE SHIFTING 1 IN RMER1
 91
002112  065032                          EMT36
002114  072032                          EHT1
002116  072132                          EDT1
002120  072160                          EFT1

 92
 93                             ;ERROR  37      CANNOT WRITE ZEROS IN RMDC
 94
002122  065046                          EMT37
002124  072032                          EHT1
002126  072132                          EDT1
002130  072160                          EFT1

 95
 96                             ;ERROR  40      CANNOT WRITE ONES IN RMDC
 97
002132  065062                          EMT40
002134  072032                          EHT1
002136  072132                          EDT1
002140  072160                          EFT1

 98
 99                             ;ERROR  41      BIT INTERFERENCE IN WRITING/READING RMDC
100
002142  065102                          EMT41
002144  072032                          EHT1
002146  072132                          EDT1
002150  072160                          EFT1

101
102                            ;ERROR  42      CANNOT WRITE 1'S IN RMDC OR RMDA
103
002152  065116                          EMT42
002154  000000                          0
```

ERROR POINTER TABLE

```
002156  000000                          0
002160  000000                          0

                                ;ERROR  43      CANNOT CLEAR RMCS1-FUNCTION CODE

002162  065142                          EMT43
002164  072032                          EHT1
002166  072132                          EDT1
002170  072160                          EFT1

                                ;ERROR  44      UNUSED BITS OF RMER2 NOT ZERO

002172  065154                          EMT44
002174  072032                          EHT1
002176  072132                          EDT1
002200  072160                          EFT1

                                ;ERROR  45      CANNOT CLEAR RMER2-OPE,IVC,LSC

002202  065170                          EMT45
002204  072032                          EHT1
002206  072132                          EDT1
002210  072160                          EFT1

                                ;ERROR  46      CANNOT CLEAR RMER2-LBC,DPE

002212  065204                          EMT46
002214  072032                          EHT1
002216  072132                          EDT1
002220  072160                          EFT1

                                ;ERROR  47      CANNOT WRITE ZEROS RMER2-OPE,IVC LSC

002222  065220                          EMT47
002224  072032                          EHT1
002226  072132                          EDT1
002230  072160                          EFT1

                                ;ERROR  50      CANNOT WRITE ZEROS RMER2-LBC,DPE

002232  065242                          EMT50
002234  072032                          EHT1
002236  072132                          EDT1
002240  072160                          EFT1

                                ;ERROR  51      CANNOT WRITE ONES RMER2

002242  065264                          EMT51
002244  072032                          EHT1
002246  072132                          EDT1
```

ERROR POINTER TABLE

```
        002250  072160                          EFT1

    125
    126                             ;ERROR  52      CANNOT WRITE SHIFTING ONES RMER2
    127
        002252  065304                          EMT52
        002254  072032                          EHT1
        002256  072132                          EDT1
        002260  072160                          EFT1

    128
    129                             ;ERROR  53      UNUSED BITS OF RMOF ARE NOT ZERO
    130
        002262  065320                          EMT53
        002264  072032                          EHT1
        002266  072132                          EDT1
        002270  072160                          EFT1

    131
    132                             ;ERROR  54      CANNOT WRITE ZEROS RMOF-FMT,ECI,HCI,OFD
    133
        002272  065334                          EMT54
        002274  072032                          EHT1
        002276  072132                          EDT1
        002300  072160                          EFT1

    134
    135                             ;ERROR  55      CANNOT WRITE ONES RMOF-FMT,ECI,HCI,OFD
    136
        002302  065354                          EMT55
        002304  072032                          EHT1
        002306  072132                          EDT1
        002310  072160                          EFT1

    137
    138                             ;ERROR  56      CANNOT WRITE SHIFTING ONES RMOF
    139
        002312  065374                          EMT56
        002314  072032                          EHT1
        002316  072132                          EDT1
        002320  072160                          EFT1

    140
    141                             ;ERROR  57      DEVICE IS NOT AN RM05/3/2
    142
        002322  065410                          EMT57
        002324  072052                          EHT57
        002326  072140                          EDT57
        002330  072166                          EFT57

    143
    144                             ;ERROR  60      DEVICE AVAILABLE IS NOT SET
    145
        002332  065424                          EMT60
        002334  072032                          EHT1
        002336  072132                          EDT1
        002340  072160                          EFT1
```

```
146
147                                     ;ERROR  61      CANNOT WRITE ZEROS RMHR
148
002342  065440                                  EMT61
002344  072032                                  EHT1
002346  072132                                  EDT1
002350  072160                                  EFT1

149
150                                     ;ERROR  62      CANNOT WRITE ONES RMHR
151
002352  065460                                  EMT62
002354  072032                                  EHT1
002356  072132                                  EDT1
002360  072160                                  EFT1

152
153                                     ;ERROR  63      CANNOT WRITE SHIFTING ONES RMHR
154
002362  065500                                  EMT63
002364  072032                                  EHT1
002366  072132                                  EDT1
002370  072160                                  EFT1

155
156                                     ;ERROR  64      CANNOT CLEAR ILR STATUS
157
002372  065514                                  EMT64
002374  072032                                  EHT1
002376  072132                                  EDT1
002400  072160                                  EFT1

158
159                                     ;ERROR  65      ILR ERROR SHOULD NOT BE SET
160
002402  065526                                  EMT65
002404  072056                                  EHT65
002406  072142                                  EDT65
002410  072170                                  EFT65

161
162                                     ;ERROR  66      ILR ERROR SHOULD BE SET
163
002412  065542                                  EMT66
002414  072056                                  EHT65
002416  072142                                  EDT65
002420  072170                                  EFT65

164
165                                     ;ERROR  67      CANNOT CLEAR PAR STATUS-DPE IS RESET
166
002422  065556                                  EMT67
002424  072032                                  EHT1
002426  072132                                  EDT1
002430  072160                                  EFT1
```

ERROR POINTER TABLE

```
                                ;ERROR  70      CANNOT CLEAR PAR AND DPE STATUS

002432  065574                          EMT70
002434  072032                          EHT1
002436  072132                          EDT1
002440  072160                          EFT1

                                ;ERROR  71      'PAR'' ERROR SHOULD NOT BE SET-'PAT'' IS OFF

002442  065614                          EMT71
002444  072062                          EHT71
002446  072144                          EDT71
002450  072172                          EFT71

                                ;ERROR  72      'PAR'' ERROR SHOULD BE SET-'PAT'' IS ON

002452  065640                          EMT72
002454  072062                          EHT71
002456  072144                          EDT71
002460  072172                          EFT71

                                ;ERROR  73      'MCPE'' ERROR SHOULD NOT BE SET

002462  065664                          EMT73
002464  072062                          EHT71
002466  072144                          EDT71
002470  072172                          EFT71

                                ;ERROR  74      UNEXPECTED BUS TIMEOUT

002472  065704                          EMT74
002474  072066                          EHT74
002476  072146                          EDT74
002500  072174                          EFT74

                                ;ERROR  75      CANT CLEAR 'DMD''

002502  065714                          EMT75
002504  072032                          EHT1
002506  072132                          EDT1
002510  072160                          EFT1

                                ;ERROR  76      CANT WRITE ZERO 'DMD''

002512  065726                          EMT76
002514  072032                          EHT1
002516  072132                          EDT1
002520  072160                          EFT1
```

```
189                                     ;ERROR  77      CANT WRITE ONE 'DMD'
190
    002522  065742                              EMT77
    002524  072032                              EHT1
    002526  072132                              EDT1
    002530  072160                              EFT1

191
192                                     ;ERROR  100     DMD SET BY WRONG BIT
193
    002532  065756                              EMT100
    002534  072062                              EHT7;
    002536  072142                              EDT65
    002540  072170                              EFT65

194
195                                     ;ERROR  101     CANT CLEAR 'MOL' IN DIAGNOSTIC MODE
196
    002542  065776                              EMT101
    002544  072032                              EHT1
    002546  072132                              EDT1
    002550  072160                              EFT1

197
198                                     ;ERROR  102     CANT SET 'MOL' IN DIAGNOSTIC MODE
199
    002552  066016                              EMT102
    002554  072032                              EHT1
    002556  072132                              EDT1
    002560  072160                              EFT1

200
201                                     ;ERROR  103     'MUR' SET BY WRONG BIT
202
    002562  066036                              EMT103
    002564  072062                              EHT71
    002566  072142                              EDT65
    002570  072170                              EFT65

203
204                                     ;ERROR  104     CANT RESET 'WRL' IN DIAGNOSTIC MODE
205
    002572  066062                              EMT104
    002574  072032                              EHT1
    002576  072132                              EDT1
    002600  072160                              EFT1

206
207                                     ;ERROR  105     CANT SET 'WRL' IN DIAGNOSTIC MODE
208
    002602  066102                              EMT105
    002604  072032                              EHT1
    002606  072132                              EDT1
    002610  072160                              EFT1

209
210                                     ;ERROR  106     'MWP' SET BY WRONG BIT
```

ERROR POINTER TABLE

```
211
     002612  066122                          EMT106
     002614  072062                          EHT71
     002616  072142                          EDT65
     002620  072170                          EFT65

212
213                                 ;ERROR  107     CANT RESET 'DVC' USING 'MDVC'
214
     002622  066146                          EMT107
     002624  072032                          EHT1
     002626  072132                          EDT1
     002630  072160                          EFT1

215
216                                 ;ERROR  110     'DVC' IS RESET BUT 'UNS' IS SET
217
     002632  066166                          EMT110
     002634  072032                          EHT1
     002636  072132                          EDT1
     002640  072160                          EFT1

218
219                                 ;ERROR  111     'DVC' IS SET BUT 'UNS' IS NOT SET
220
     002642  066210                          EMT111
     002644  072032                          EHT1
     002646  072132                          EDT1
     002650  072160                          EFT1

221
222                                 ;ERROR  112     CANT SET 'DVC' USING MDVC'
223
     002652  066230                          EMT112
     002654  072032                          EHT1
     002656  072132                          EDT1
     002660  072160                          EFT1

224
225                                 ;ERROR  113     'DVC' IS RESET BUT 'UNS' IS SET
226
     002662  066250                          EMT113
     002664  072032                          EHT1
     002666  072132                          EDT1
     002670  072160                          EFT1

227
228                                 ;ERROR  114     'DVC' IS SET BUT 'UNS' IS NOT SET
229
     002672  066274                          EMT114
     002674  072032                          EHT1
     002676  072132                          EDT1
     002700  072160                          EFT1

230
231                                 ;ERROR  115     'MDF' IS SET BY WRONG BIT
232
```

ERROR POINTER TABLE

```
002702  066316                          EMT115
002704  072072                          EHT115
002706  072150                          EDT115
002710  072176                          EFT115

233
234                             ;ERROR  116     CANT RESET ''SKI'' USING 'MSER''
235
002712  066342                          EMT116
002714  072032                          EHT1
002716  072132                          EDT1
002720  072160                          EFT1

236
237                             ;ERROR  117     CANT SET ''SKI'' USING 'MSER''
238
002722  066362                          EMT117
002724  072032                          EHT1
002726  072132                          EDT1
002730  072160                          EFT1

239
240                             ;ERROR  120     ''SKI'' SET BY WRONG BIT
241
002732  066402                          EMT120
002734  072072                          EHT115
002736  072150                          EDT115
002740  072176                          EFT115

242
243                             ;ERROR  121     CANT RESET ''PIP'' USING 'MOC''
244
002742  066426                          EMT121
002744  072032                          EHT1
002746  072132                          EDT1
002750  072160                          EFT1

245
246                             ;ERROR  122     CANT SET ''PIP'' USING 'MOC''
247
002752  066446                          EMT122
002754  072032                          EHT1
002756  072132                          EDT1
002760  072160                          EFT1

248
249                             ;ERROR  123     ''MOC'' SET BY WRONG BIT
250
002762  066466                          EMT123
002764  072072                          EHT115
002766  072150                          EDT115
002770  072176                          EFT115

251
252                             ;ERROR  124     CANT CLEAR ''EBL''
253
002772  066512                          EMT124
```

ERROR POINTER TABLE

```
003000  072160                          EFT1

254
255                             ;ERROR  125     "EBL" NOT ZERO IN DIAGNOSTIC MODE
256
003002  066530                          EMT125
003004  072032                          EHT1
003006  072132                          EDT1
003010  072160                          EFT1

257
258                             ;ERROR  126     CANT SET "EBL" USING "DEBL"
259
003012  066550                          EMT126
003014  072032                          EHT1
003016  072132                          EDT1
003020  072160                          EFT1

260
261                             ;ERROR  127     "DEBL" SET BY WRONG BIT
262
003022  066566                          EMT127
003024  072072                          EHT115
003026  072150                          EDT115
003030  072176                          EFT115

263
264                             ;ERROR  130     "LS" NOT CORRECT ACCORDING TO RMDA
265
003032  066612                          EMT130
003034  072076                          EHT130
003036  072152                          EDT130
003040  072200                          EFT130

266
267                             ;ERROR  131     "LST" NOT CORRECT ACCORDING TO RMDA
268
003042  066630                          EMT131
003044  072076                          EHT130
003046  072152                          EDT130
003050  072200                          EFT130

269
270                             ;ERROR  132     CANNOT INCREMENT SECTOR ADDRESS USING "DEBL"
271
003052  066646                          EMT132
003054  072102                          EHT132
003056  072154                          EDT132
003060  072202                          EFT132

272
273                             ;ERROR  133     CANNOT INCREMENT TRACK ADDRESS USING "DEBL"
274
003062  066666                          EMT133
003064  072102                          EHT132
```

```
003066  072154                          EDT132
003070  072202                          EFT132

275
276                             ;ERROR  134     UNUSED BITS OF RMDC NOT ZERO
277
003072  066706                          EMT134
003074  072032                          EHT1
003076  072132                          EDT1
003100  072160                          EFT1

278
279                             ;ERROR  135     "VV" NOT RESET BY UNIT READY
280
003102  066722                          EMT135
003104  072032                          EHT1
003106  072132                          EDT1
003110  072160                          EFT1

281
282                             ;ERROR  136     SERIAL NUMBER IS INCONSISTENT
283
003112  06674C                          EMT136
003114  072032                          EHT1
003116  072132                          EDT1
003120  072160                          EFT1

284
285                             ;ERROR  137     CANT CLEAR "GO" BIT
286
003122  066752                          EMT137
003124  072032                          EHT1
003126  072132                          EDT1
003130  072160                          EFT1

287
288                             ;ERROR  140     CANT INCREMENT CYLINDER USING "DEBL"
289
003132  066770                          EMT140
003134  072102                          EHT132
003136  072154                          EDT132
003140  072202                          EFT132

290
291                             ;ERROR  141     CANT RESET "LBT" BY WRITING RMDA
292
003142  067010                          EMT141
003144  072106                          EHT142
003146  072154                          EDT132
003150  072202                          EFT132

293
294                             ;ERROR  142     CANT SET "LBT" USING "DEBL"
295
003152  067024                          EMT142
003154  072106                          EHT142
003156  072154                          EDT132
```

```
003160  072202                          EFT132

296
297                             ;ERROR  143     CANT READ ZERO FROM COMP ERROR
298
003162  067042                          EMT143
003164  072032                          EHT1
003166  072132                          EDT1
003170  072160                          EFT1

299
300                             ;ERROR  144     CANT SET COMP ERROR WITH RMER1 OR RMER2
301
003172  067056                          EMT144
003174  072032                          EHT1
003176  072132                          EDT1
003200  072160                          EFT1

302
303                             ;ERROR  145     COMP ERROR DID NOT SET
304
003202  067100                          EMT145
003204  072112                          EHT145
003206  072152                          EDT130
003210  072200                          EFT130

305
306                             ;ERROR  146     CANT SET ''GO'' BIT
307
003212  067122                          EMT146
003214  072032                          EHT1
003216  072132                          EDT1
003220  072160                          EFT1

308
309                             ;ERROR  147     CANT READ A ONE FROM ''TST''
310
003222  067140                          EMT147
003224  072032                          EHT1
003226  072132                          EDT1
003230  072160                          EFT1

311
312                             ;ERROR  150     ''TST'' IS INCORRECT FOR THE FUNCTION CODE
313
003232  067152                          EMT150
003234  072116                          EHT150
003236  072150                          EDT115
003240  072176                          EFT115

314
315                             ;ERROR  151     CANT SET THE ''GO'' BIT
316
003242  067174                          EMT151
003244  072032                          EHT1
003246  072132                          EDT1
003250  072160                          EFT1
```

```
                                ;ERROR  152     'DRY' NOT THE COMPLEMENT OF 'GO'

003252  067206                          EMT152
003254  072032                          EHT1
003256  072132                          EDT1
003260  072160                          EFT1

                                ;ERROR  153     'GO' RESET EARLY

003262  067222                          EMT153
003264  072032                          EHT1
003266  072132                          EDT1
003270  072160                          EFT1

                                ;ERROR  154     'GO' DIDNT RESET ON TIME

003272  067242                          EMT154
003274  072032                          EHT1
003276  072132                          EDT1
003300  072160                          EFT1

                                ;ERROR  155     CANT CLEAR CONTINUE

003302  067262                          EMT155
003304  072032                          EHT1
003306  072132                          EDT1
003310  072160                          EFT1

                                ;ERROR  156     CONTINUE IS INCORRECT FOR THE FUNCTION CODE

003312  067300                          EMT156
003314  072116                          EHT150
003316  072150                          EDT115
003320  072176                          EFT115

                                ;ERROR  157     CANT CLEAR IVC

003322  067322                          EMT157
003324  072032                          EHT1
003326  072132                          EDT1
003330  072160                          EFT1

                                ;ERROR  160     IVC IS INCORRECT FOR THE FUNCTION CODE

003332  067340                          EMT160
003334  072116                          EHT150
003336  072150                          EDT115
003340  072176                          EFT115
```

ERROR POINTER TABLE

```
338
339                                 ;ERROR  161     CANT CLEAR LSC
340
003342  067370                              EMT161
003344  072032                              EHT1
003346  072132                              EDT1
003350  072160                              EFT1

341
342                                 ;ERROR  162     CANT SET LSC
343
003352  067406                              EMT162
003354  072032                              EHT1
003356  072132                              EDT1
003360  072160                              EFT1

344
345                                 ;ERROR  163     COMMAND DECODE WAS ENABLED WITH COMP ERROR SET
346
003362  067424                              EMT163
003364  072032                              EHT1
003366  072132                              EDT1
003370  072160                              EFT1

347
348                                 ;ERROR  164     COMMAND DECODE WAS ENABLED WITH COMP E.ROR SET
349
003372  067446                              EMT164
003374  072032                              EHT1
003376  072132                              EDT1
003400  072160                              EFT1

350
351                                 ;ERROR  165     DECODE DOES NOT SET
352
003402  067470                              EMT165
003404  000000                              0
003406  000000                              0
003410  000000                              0

353
354                                 ;ERROR  166     CANT CLEAR OCCUPIED
355
003412  067514                              EMT166
003414  072032                              EHT1
003416  072132                              EDT1
003420  072160                              EFT1

356
357                                 ;ERROR  167     ILF SET WITHOUT GO BIT
358
003422  067534                              EMT167
003424  072032                              EHT1
003426  072132                              EDT1
003430  072160                              EFT1

359
```

```
360                                     ;ERROR  170     CANT SET VOLUME VALID
361
003432  067556                                  EMT170
003434  072116                                  EHT150
003436  072150                                  EDT115
003440  072176                                  EFT115

362
363                                     ;ERROR  171     ILF IS INCORRECT
364
003442  067570                                  EMT171
003444  072116                                  EHT150
003446  072150                                  EDT115
003450  072176                                  EFT115

365
366                                     ;ERROR  172     CANT SET OFFSET DIRECTION BIT
367
003452  067604                                  EMT172
003454  072032                                  EHT1
003456  072132                                  EDT1
003460  072160                                  EFT1

368
369                                     ;ERROR  173     OCCUPIED IS INCORRECT FOR FUNCTION CODE
370
003462  067622                                  EMT173
003464  072116                                  EHT150
003466  072150                                  EDT115
003470  072176                                  EFT115

371
372                                     ;ERROR  174     READ IN PRESET DIDNT CLEAR RMDA, RMDC OR RMOF
373
003472  067644                                  EMT174
003474  000000                                  0
003476  000000                                  0
003500  000000                                  0

374
375                                     ;ERROR  175     READ IN PRESET DIDNT CLEAR RMOF
376
003502  067672                                  EMT175
003504  072032                                  EHT1
003506  072132                                  EDT1
003510  072160                                  EFT1

377
378                                     ;ERROR  176     READ IN PRESET DIDNT CLEAR RMDA
379
003512  067710                                  EMT176
003514  072032                                  EHT1
003516  072132                                  EDT1
003520  072160                                  EFT1

380
381                                     ;ERROR  177     --RESERVED FOR POWER MONITOR BIT FAILURE--
```

```
382
003522  000000                          0
003524  000000                          0
003526  000000                          0
003530  000000                          0

383
384                             ;ERROR  200     CANT SET OFFSET MODE BY OFFSET COMMAND
385
003532  067726                          EMT200
003534  072032                          EHT1
003536  072132                          EDT1
003540  072160                          EFT1

386
387                             ;ERROR  201     CANT RESET OFFSET MODE BY RTC COMMAND
388
003542  067744                          EMT201
003544  072032                          EHT1
003546  072132                          EDT1
003550  072160                          EFT1

389
390                             ;ERROR  202     CANT RESET OFD BY RTC COMMAND
391
003552  067762                          EMT202
003554  072032                          EHT1
003556  072132                          EDT1
003560  072160                          EFT1

392
393                             ;ERROR  203     CANT RESET OM BY RMDC
394
003562  070000                          EMT203
003564  072032                          EHT1
003566  072132                          EDT1
003570  072160                          EFT1

395
396                             ;ERROR  204     CANT RESET OM BY EBL
397
003572  070022                          EMT204
003574  072032                          EHT1
003576  072132                          EDT1
003600  072160                          EFT1

398
399                             ;ERROR  205     RUN AND GO NOT CORRECT FOR FUNCTION CODE
400
003602  070042                          EMT205
003604  072116                          EHT150
003606  072150                          EDT115
003610  072176                          EFT115

401
402                             ;ERROR  206     CANT SET IAE ERROR
403
```

ERROR POINTER TABLE

```
003612  070062                          EMT206
003614  000000                          0
003616  000000                          0
003620  000000                          0

404
405                             ;ERROR  207     IAE IS INCORRECT FOR FUNCTION CODE
406
003622  070112                          EMT207
003624  072116                          EHT150
003626  072150                          EDT115
003630  072176                          EFT115

407
408                             ;ERROR  210     IAE IS INCORRECT FOR RMDA
409
003632  070126                          EMT210
003634  072072                          EHT115
003636  072150                          EDT115
003640  072176                          EFT115

410
411                             ;ERROR  211     IAE IS INCORRECT FOR RMDC
412
003642  070144                          EMT211
003644  072072                          EHT115
003646  072150                          EDT115
003650  072176                          EFT115

413
414                             ;ERROR  212     CANT SET AOE
415
003652  070162                          EMT212
003654  072106                          EHT142
003656  072154                          EDT132
003660  072202                          EFT132

416
417                             ;ERROR  213     RMR SET WHEN WRITING RMAS OR RMCS
418
003662  070174                          EMT213
003664  072122                          EHT213
003666  072150                          EDT115
003670  072176                          EFT115

419
420                             ;ERROR  214     CANT SET RMR
421
003672  070224                          EMT214
003674  072122                          EHT213
003676  072150                          EDT115
003700  072176                          EFT115

422
423                             ;ERROR  215     DRQ IS 0 AND PGM IS 1
424
003702  070236                          EMT215
```

```
                003704  072032                          EMT1
                003706  072132                          EDT1
                003710  072160                          EFT1

425
426                                     ;ERROR  216     DVA IS NOT SET
427
                003712  070256                          EMT216
                003714  072032                          EMT1
                003716  072132                          EDT1
                003720  072160                          EFT1

428
429                                     ;ERROR  217     DPR IS NOT SET
430
                003722  070272                          EMT217
                003724  072032                          EMT1
                003726  072132                          EDT1
                003730  072160                          EFT1

431
432                                     ;ERROR  220     CANT SET PORT REQUEST BY READING RMCS1
433
                003732  070306                          EMT220
                003734  072126                          EMT220
                003736  072156                          EDT220
                003740  072204                          EFT220

434
435                                     ;ERROR  221     CANT SET PORT REQUEST BY WRITING RMAS
436
                003742  070324                          EMT221
                003744  072126                          EMT220
                003746  072156                          EDT220
                003750  072204                          EFT220

437
438                                     ;ERROR  222     CANT SET PORT REQUEST BY WRITING RMDA
439
                003752  070342                          EMT222
                003754  072126                          EMT220
                003756  072156                          EDT220
                003760  072204                          EFT220

440
441                                     ;ERROR  223     CANT RESET PORT REQUEST BY RELEASE COMMAND
442
                003762  070360                          EMT223
                003764  072126                          EMT220
                003766  072156                          EDT220
                003770  072204                          EFT220

443
444                                     ;ERROR  224     CANT CLEAR ATA BY RMAS
445
                003772  070376                          EMT224
                003774  072032                          EMT1
```

```
003776  072132                          EDT1
004000  072160                          EFT1

446
447                             ;ERROR  225     ATA IS RESET BUT RMAS NOT ZERO
448
004002  070416                          EMT225
004004  072032                          EHT1
004006  072132                          EDT1
004010  072160                          EFT1

449
450                             ;ERROR  226     CANT RESET ATA BY GO
451
004012  070440                          EMT226
004014  072032                          EHT1
004016  072132                          EDT1
004020  072160                          EFT1

452
453                             ;ERROR  227     ATA NOT SET BY UNIT READY
454
004022  070456                          EMT227
004024  072032                          EHT1
004026  072132                          EDT1
004030  072160                          EFT1

455
456                             ;ERROR  230     ATA NOT SET BY UNIT READY
457
004032  070474                          EMT230
004034  072032                          EHT1
004036  072132                          EDT1
004040  072160                          EFT1

458
459                             ;ERROR  231     ATA NOT SET BY COMP ERROR
460
004042  070512                          EMT231
004044  072032                          EHT1
004046  072132                          EDT1
004050  072160                          EFT1

461
462                             .ERROR  232     ATA SET/DID NOT SET WHEN REGISTER WRITTEN
463                             ;               WHILE COMP ERROR WAS SET
464
004052  070526                          EMT232
004054  072122                          EHT213
004056  072150                          EDT115
004060  072176                          EFT115

465
466                             ;ERROR  233     ATA NOT SET BY COMMAND SEQUENCER
467
004062  070550                          EMT233
004064  072116                          EHT150
```

```
004066  072150                          EDT115
004070  072176                          EFT115

468
469                             ;ERROR  234     WLE INCORRECT ACCORDING TO FUNCTION CODE
470
004072  070572                          EMT234
004074  072116                          EHT150
004076  072150                          EDT115
004100  072176                          EFT115

471
472                             ;ERROR  235     CANT CLEAR EXCEPTION
473
004102  070620                          EMT235
004104  072032                          EHT1
004106  072132                          EDT1
004110  072160                          EFT1

474
475                             ;ERROR  236     CANT SET EXCEPTION
476
004112  070640                          EMT236
004114  072072                          EHT115
004116  072150                          EDT115
004120  072176                          EFT115

477
478                             ;ERROR  237     CANT CLEAR IVC
479
004122  070672                          EMT237
004124  072032                          EHT1
004126  072132                          EDT1
004130  072160                          EFT1

480
481                             ;ERROR  240     CANT SET IVC
482
004132  070672                          EMT237
004134  072116                          EHT150
004136  072150                          EDT115
004140  072176                          EFT115

483
484                             ;ERROR  241     OPI NOT SET DURING RECALIBRATE
485
004142  070736                          EMT241
004144  072032                          EHT1
004146  072132                          EDT1
004150  072160                          EFT1

486
487                             ;ERROR  242     RECALIBRATE DID NOT ABORT WHEN DRIVE FAULT SET
488
004152  070762                          EMT242
004154  072032                          EHT1
004156  072132                          EDT1
```

ERROR POINTER TABLE

```
004160  072160                          EFT1

489
490                             ;ERROR  243     OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
491                             ;               DROPPED DURING RECALIBRATE
492
004162  071006                          EMT243
004164  072032                          EHT1
004166  072132                          EDT1
004170  072160                          EFT1

493
494                             ;ERROR  244     ATA NOT SET DURING RECALIBRATE
495
004172  071032                          EMT244
004174  072032                          EHT1
004176  072132                          EDT1
004200  072160                          EFT1

496
497                             ;ERROR  245     GO RESET EARLY DURING RECALIBRATE
498
004202  071050                          EMT245
004204  072032                          EHT1
004206  072132                          EDT1
004210  072160                          EFT1

499
500                             ;ERRCR  246     GO NOT RESET DURING RECALIBRATE
501
004212  071066                          EMT246
004214  072032                          EHT1
004216  072132                          EDT1
004220  072160                          EFT1

502
503                             ;ERROR  247     INCORRECT TAG BUS DURING RECALIBRATE
504
004222  071112                          EMT247
004224  072032                          EHT1
004226  072132                          EDT1
004230  072160                          EFT1

505
506                             ;ERROR  250     OPI SHOULD HAVE SET DURING SEEK BECAUSE UNIT
507                             ;               READY DROPPED
508
004232  071130                          EMT250
004234  072032                          EHT1
004236  072132                          EDT1
004240  072160                          EFT1

509
510                             ;ERROR  251     SEEK DID NOT ABORT WHEN DRIVE FAULT SET
511
004242  071154                          EMT251
004244  072032                          EHT1
```

ERROR POINTER TABLE

```
004246  072132                          EDT1
004250  072160                          EFT1

                                ;ERROR  252     OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
                                ;               DROPPED DURING SEEK

004252  071200                          EMT252
004254  072032                          EHT1
004256  072132                          EDT1
004260  072160                          EFT1

                                ;ERROR  253     ATA NOT SET DURING SEEK

004262  071224                          EMT253
004264  072032                          EHT1
004266  072132                          EDT1
004270  072160                          EFT1

                                ;ERROR  254     GO RESET EARLY DURING SEEK

004272  071242                          EMT254
004274  072032                          EHT1
004276  072132                          EDT1
004300  072160                          EFT1

                                ;ERROR  255     GO DID NOT RESET DURING SEEK

004302  071260                          EMT255
004304  072032                          EHT1
004306  072132                          EDT1
004310  072160                          EFT1

                                ;ERROR  256     INCORRECT TAG BUS DURING SEEK

004312  071304                          EMT256
004314  072032                          EHT1
004316  072132                          EDT1
004320  072160                          EFT1

                                ;ERROR  257     OPI NOT SET DURING SEARCH

004322  071322                          EMT257
004324  072032                          EHT1
004326  072132                          EDT1
004330  072160                          EFT1

                                ;ERROR  260     SEARCH DID NOT ABORT WHEN DRIVE FAULT SET

004332  071346                          EMT260
004334  072032                          EHT1
```

```
004336  072132                          EDT1
004340  072160                          EFT1

534
535                             ;ERROR  261     OPI SHOULD HAVE SET BECAUSE ON CYLINDER NEVER
536                             ;               DROPPED DURING SEARCH
537
004342  071372                          EMT261
004344  072032                          EHT1
004346  072132                          EDT1
004350  072160                          EFT1

538
539                             ;ERROR  262     ATA NOT SET DURING SEARCH
540
004352  071416                          EMT262
004354  072032                          EHT1
004356  072132                          EDT1
004360  072160                          EFT1

541
542                             ;ERROR  263     GO RESET EARLY DURING SEARCH
543
004362  071434                          EMT263
004364  072032                          EHT1
004366  072132                          EDT1
004370  072160                          EFT1

544
545                             ;ERROR  264     GO DID NOT RESET DURING SEARCH
546
004372  071452                          EMT264
004374  072032                          EHT1
004376  072132                          EDT1
004400  072160                          EFT1

547
548                             ;ERROR  265     SEARCH ENABLE DIDNT SET DURING SEARCH
549
004402  071476                          EMT265
004404  072032                          EHT1
004406  072132                          EDT1
004410  072160                          EFT1

550
551                             ;ERROR  266     INCORRECT TAG BUS DURING SEARCH
552
004412  071514                          EMT266
004414  072032                          EHT1
004416  072132                          EDT1
004420  072160                          EFT1

553
554                             ;ERROR  267     OPI NOT SET BY SEARCH TIMEOUT
555
004422  071532                          EMT267
004424  072032                          EHT1
```

```
004426  072132                          EDT1
004430  072160                          EFT1

556
557                             ;ERROR  270     OPI NOT SET DURING DATA COMMAND
558
004432  071546                          EMT270
004434  072032                          EHT1
004436  072132                          EDT1
004440  072160                          EFT1

559
560                             ;ERROR  271     DATA COMMAND DID NOT ABORT WHEN DRIVE FAULT SET
561
004442  071572                          EMT271
004444  072032                          EHT1
004446  072132                          EDT1
004450  072160                          EFT1

562
563                             ;ERROR  272     EBL RESET EARLY DURING DATA COMMAND
564
004452  071616                          EMT272
004454  072032                          EHT1
004456  072132                          EDT1
004460  072160                          EFT1

565
566                             ;ERROR  273     EBL DIDNT RESET ON TIME DURING DATA COMMAND
567
004462  071634                          EMT273
004464  072032                          EHT1
004466  072132                          EDT1
004470  072160                          EFT1

568
569                             ;ERROR  274     GO NOT RESET DURING DATA COMMAND
570
004472  071652                          EMT274
004474  072032                          EHT1
004476  072132                          EDT1
004500  072160                          EFT1

571
572                             ;ERROR  275     RUN AND GO NOT SET DURING DATA COMMAND
573
004502  071670                          EMT275
004504  072032                          EHT1
004506  072132                          EDT1
004510  072160                          EFT1

574
575                             ;ERROR  276     INCORRECT TAG BUS DURING DATA COMMAND
576
004512  071710                          EMT276
004514  072032                          EHT1
004516  072132                          EDT1
```

ERROR POINTER TABLE

```
004520  072160                          EFT1

577
578                             ;ERROR  277     OPI NOT SET DURING DATA COMMAND WHEN ON
579                             ;               CYLINDER DIDNT DROP
580
004522  071726                          EMT277
004524  072032                          EHT1
004526  072132                          EDT1
004530  072160                          EFT1

581
582                             ;ERROR  300     DATA COMMAND DID NOT ABORT WHEN SEEK ERROR SET
583
004532  071752                          EMT300
004534  072032                          EHT1
004536  072132                          EDT1
004540  072160                          EFT1

584
585                             ;ERROR  301     SEARCH NOT ENABLED DURING DATA COMMAND
586
004542  071776                          EMT301
004544  072032                          EHT1
004546  072132                          EDT1
004550  072160                          EFT1

587
588                             ;ERROR  302     READ IN PRESET DIDNT CLEAR RMDC
589
004552  072014                          EMT302
004554  072032                          EHT1
004556  072132                          EDT1
004560  072160                          EFT1

590                             ;PUT ERROR TABLE HERE
```

```
                                        ;THIS ROUTINE HANDLES UNEXPECTED TIMEOUTS

004562  011600                  BADTMO: MOV     (SP),R0         ;SAVE PC WHERE THE TIME OUT OCCURED
004564  005740                          TST     -(R0)           ;ADJUST PC -2
004566  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK POINTER
004570  104401  004576                  TYPE    ,65$            ;;TYPE ASCIZ STRING
004574  000417                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>/UNEXPECTED BUS TIMEOUT, PC=/
004634                          64$:
004634  010046                          MOV     R0,-(SP)        ;SETUP FOR TYPING OUT PC
004636  104402                          TYPOC
004640  000240                          NOP                     ;PUT 'HALT(0)' INSTRUCTION HERE IF YOU WISH
                                                                ;TO STOP ON UNEXPECTED TIMEOUT.


                                .SBTTL  START OF PROGRAM

004642  012737  177777  001330  START1: MOV     #-1,CHGADR      ;CHANGE RH/RM BUS ADDRESS
004650  000402                          BR      START2

C04652  005037  0C1330          START:  CLR     CHGADR          ;NO CHANGE IN ADDRESS
004656  000240                  START2: NOP
004660  005227  000000                  INC     #0              ;TTY LOOP, WAIT FOR INCREMENT
004664  001375                          BNE     .-4             ;OF WORD
004666  000005                          RESET                   ;RESET THE WORLD


                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
004670  012706  001114                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
004674  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
004676  022706  001154                  CMP     #SWR,R6         ;;DONE?
004702  001374                          BNE     .-6             ;;LOOP BACK IF NO
004704  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
004710  012737  056100  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
004716  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
004724  012737  057020  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
004732  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
004740  012737  062054  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
004746  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
004754  012737  062162  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
004762  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
004770  013737  053656  053650          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
004776  005037  001206                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
005002  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
005006  112737  000001  001131          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
005014  012737  005014  001122          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
005022  012737  005022  001124          MOV     #.,$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.
005030  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
005034  012737  005070  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
005042  012737  177570  001154          MOV     #DSWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
005050  012737  177570  001156          MOV     #DDISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
005056  022777  177777  174070          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
005064  001012                          BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                ;;AND  THE HARDWARE SWR IS NOT - -1
005066  000403                          BR      65$             ;;BRANCH IF NO TIMEOUT
```

```
                005070  012716  005076          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
                005074  000002                          RTI
                005076  012737  000176  001154  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
                005104  012737  000174  001156          MOV     #DISPREG,DISPLAY
                005112  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

                005116  005037  001230                  CLR     $PASS           ;;CLEAR PASS COUNT
                005122  132737  000200  001243          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
                005130  001403                          BEQ     67$             ;;YES,USE NON-APT SWITCH
                005132  012737  001244  001154          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
                005140                          67$:
                                ;SETUP ''TIMEOUT'' TRAP VECTOR FOR UNEXPECTED BUS TIMEOUTS
                005140  012737  004562  000004          MOV     #BADTMO,ERRVEC  ;SETUP FOR UNEXPECTED TIMEOUT
                005146  012737  000300  000006          MOV     #PR6,ERRVEC+2   ;LEVEL 6

                                .SBTTL  TYPE PROGRAM NAME
                                ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
                005154  005227  177777                  INC     #-1             ;;FIRST TIME?
                005160  001034                          BNE     68$             ;;BRANCH IF NO
                005162  022737  054020  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
                005170  001430                          BEQ     68$             ;;BRANCH IF YES
                005172  104401  005200                  TYPE    ,69$            ;;TYPE ASCIZ STRING
                005176  000425                          BR      68$             ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  <CRLF>@CZRMPB0 - RM05/3/2 DISKLESS TEST, PT 1@<CRLF>
                005252                          68$:
                                .SBTTL  GET VALUE FOR SOFTWARE SWITCH REGISTER
                005252  005737  000042                  TST     @#42            ;;ARE WE RUNNING UNDER XXDP/ACT?
                005256  001012                          BNE     70$             ;;BRANCH IF YES
                005260  123727  001242  000001          CMPB    $ENV,#1         ;;ARE WE RUNNING UNDER APT?
                005266  001406                          BEQ     70$             ;;BRANCH IF YES
                005270  023727  001154  000176          CMP     SWR,#SWREG      ;;SOFTWARE SWITCH REG SELECTED?
                005276  001005                          BNE     71$             ;;BRANCH IF NO
                005300  104407                          GTSWR                   ;;GET SOFT-SWR SETTINGS
                005302  000403                          BR      71$
                005304  112737  000001  001150  70$:    MOVB    #1,$AUTOB       ;;SET AUTO-MODE INDICATOR
                005312                          71$:

                                ;THE FOLLOWING FINDS OUT THE PROGRAM CONTROL MODE:
                                ;PAPER TAPE (MANUAL), ACT11, XXDP CHAIN OR DUMP

                005312  005037  001332                  CLR     XXDP            ;CLEAR 'XXDP' LOAD DEVICE STORAGE
                005316  122737  000016  000041          CMPB    #16,@#41        ;LOADED FROM AN RM05/3/2 ?
                005324  001160                          BNE     5$              ;BRANCH IF NOT
                005326  013737  000040  001332          MOV     @#40,XXDP       ;GET DEVICE INDICATOR AND NUMBER
                005334  122737  000007  001332          CMPB    #7,XXDP         ;IS IT A VALID NUMBER ?
                005342  103002                          BHIS    1$              ;YES
                005344  105037  001332                  CLRB    XXDP            ;NO, DEFAULT TO DRIVE 0
                005350  005737  000042          1$:     TST     @#42            ;CHAIN MODE OR ACT11 AUTO ACCEPT ?
                005354  001425                          BEQ     3$              ;BR IF NEITHER
                005356  104401  005364                  TYPE    ,73$            ;;TYPE ASCIZ STRING
                005362  000412                          BR      72$             ;;GET OVER THE ASCIZ
                                ;;73$:  .ASCIZ  <CRLF>/NOT TESTING DRIVE /
                005410                          72$:
                005410  005046                          CLR     -(SP)           ;CLEAR WORD ON STACK
                005412  113716  001332                  MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
                005416  104403                          TYPOS                   ;TYPE THE ADDRESS
                005420     001                          .BYTE   1               ;ONLY 1 CHARACTER
```

```
 47 005421     000                          .BYTE   0               ;SUPRESS LEADING ZEROS
 48 005422  104401  001217                  TYPE    ,$CRLF          ;CR-LF
 49 005426  000517                          BR      5$              ;GET NUMBER OF DRIVES
 50
 51 005430  005227  177777          3$:     INC     #-1             ;FIRST TIME THRU HERE ?
 52 005434  001114                          BNE     5$              ;NO
 53 005436  104401  005444                  TYPE    ,75$            ;;TYPE ASCIZ STRING
    005442  000410                          BR      74$             ;;GET OVER THE ASCIZ
                                    ;;75$:  .ASCIZ  <CRLF>/TO TEST DRIVE /
    005464                          74$:
 54 005464  005046                          CLR     -(SP)           ;CLEAR WORD ON STACK
 55 005466  113716  001332                  MOVB    XXDP,(SP)       ;GET DRIVE ADDRESS
 56 005472  104403                          TYPOS                   ;TYPE DRIVE ADDRESS
 57 005474     001                          .BYTE   1               ;ONLY 1 CHARACTER
 58 005475     000                          .BYTE   0               ;SUPRESS LEADING ZEROS
 59 005476  104401  005504                  TYPE    ,77$            ;;TYPE ASCIZ STRING
    005502  000431                          BR      76$             ;;GET OVER THE ASCIZ
                                    ;;77$:  .ASCIZ  /, HALT PROGRAM, REMOVE RRDP PACK AND REPLACE IT/<CRLF>
    005566                          76$:
 60 005566  104401  005574                  TYPE    ,78$            ;;TYPE ASCIZ STRING
    005572  000435                          BR      5$              ;;GET OVER THE ASCIZ
                                    ;;78$:  .ASCIZ  /WITH A WORK PACK, CLEAR LOCATION 40 AND RESTART PROGRAM./<CRLF>
    005666                          5$:
 64                                 ;CHECK FOR AUTO MODE OR STANDALONE
 65 005666  005037  001326                  CLR     AUTSIZ          ;LET AUTO DRIVE SIZING OCCUR
 66 005672  005737  000042                  TST     @#42            ;RUNNING IN AUTO MODE ?
 67 005676  001561                          BEQ     STANDALONE      ;BR IF NO
 68 005700  012737  000377  001300          MOV     #377,$DEVM      ;SET DEVICE MAP FOR ALL DRIVES
 69
 70                                 ;PROGRAM IS RUNNING IN AUTO MODE - SEE IF SIZING IS ALLOWED
 71 005706                          XSIZ:
 72 005706  132737  000200  001243          BITB    #BIT7,$ENVM     ;SIZING ALLOWED ?
 73 005714  001146                          BNE     12$             ;NO
 74
 75 005716  005001                          CLR     R1              ;START FROM DRIVE 0
 76 005720  013700  001276                  MOV     $BASE,R0        ;LOAD THE BASE ADDRESS
 77 005724  104401  063234                  TYPE    ,SYSTAT         ;TYPE 'UNIT STATUS:'
 78
 79 005730  136137  063516  001300  1$:     BITB    ATNTBL(R1),$DEVM        ;IS DEVICE PRESENT IN MAP ?
 80 005736  001531                          BEQ     11$             ;BR IF NO
 81 005740  104401  001217                  TYPE    ,$CRLF          ;CR-LF
 82 005744  010146                          MOV     R1,-(SP)        ;;SAVE R1 FOR TYPEOUT
    005746  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
    005750     002                          .BYTE   2               ;;TYPE 2 DIGIT(S)
    005751     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS
 83 005752  104401  063410                  TYPE    ,BLNKS4         ;TYPE 4 BLANKS
 84
 85 005756  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR MASS BUS
 86 005764  010160  000010                  MOV     R1,RMCS2(R0)    ;LOAD THE DRIVE ADDRESS
 87 005770  005760  000012                  TST     RMDS(R0)        ;ACCESS DRIVE REGISTER
 88 005774  032760  010000  000010          BIT     #NED,RMCS2(R0)  ;IS DRIVE PRESENT ?
 89 006002  001051                          BNE     3$              ;BR IF NO
 90 006004  032760  004000  000000          BIT     #DVA,RMCS1(R0)  ;IS DRIVE AVAILABLE ?
 91 006012  001450                          BEQ     4$              ;BR IF NO
 92 006014  012737  063252  006220          MOV     #$RM02,10$      ;ASSUME RM02 DEVICE
 93 006022  016002  000026                  MOV     RMDT(R0),R2     ;SAVE DRIVE TYPE REGISTER IN R2
 94 006026  022702  020025                  CMP     #20025,R2       ;SINGLE PORT RM02 ?
```

```
 95 006032  001430                        BEQ     2$              ;BR IF YES
 96 006034  022702  024025                CMP     #24025,R2       ;DUAL PORT RM02 ?
 97 006040  001425                        BEQ     2$              ;BR IF YES
 98 006042  012737  063257  006220        MOV     #$RM03,10$      ;ASSUME RM03 DEVICE
 99 006050  022702  020024                CMP     #20024,R2       ;SINGLE PORT RM03 ?
100 006054  001417                        BEQ     2$              ;BR IF YES
101 006056  022702  024024                CMP     #24024,R2       ;DUAL PORT RM03 ?
102 006062  001414                        BEQ     2$              ;BR IF YES
103 006064  012737  063264  006220        MOV     #$RM05,10$      ;ASSUME RM05 DEVICE
104 006072  022702  020027                CMP     #20027,R2       ;SINGLE PORT RM05 ?
105 006076  001406                        BEQ     2$              ;BR IF YES
106 006100  022702  024027                CMP     #24027,R2       ;DUAL PORT RM05 ?
107 006104  001403                        BEQ     2$              ;BR IF YES
108 006106  104401  063271                TYPE    ,NOTRM          ;DRIVE NOT AN RM05/3/2
109 006112  000443                        BR      11$             ;CHECK NEXT DRIVE
110 006114  032760  010000  000012  2$:   BIT     #MOL,RMDS(R0)   ;IS MEDIUM ON LINE ?
111 006122  001415                        BEQ     6$              ;BR IF NO
112 006124  000417                        BR      7$
113
114 006126  104401  063327          3$:   TYPE    ,NOTPRS         ;DRIVE NOT PRESENT
115 006132  000402                        BR      5$              ;CHECK NEXT DRIVE
116
117 006134  104401  063344          4$:   TYPE    ,NOTAVL         ;DRIVE NOT AVAILABLE
118 006140  005737  001326          5$:   TST     AUTSIZ          ;AUTO SIZING ON ?
119 006144  001026                        BNE     11$             ;BR IF NO
120 006146  146137  063516  001300        BICB    ATNTBL(R1),$DEVM        ;CLEAR DEVICE FROM BIT MAP
121 006154  000422                        BR      11$             ;CHECK NEXT DRIVE
122
123 006156  104401  063363          6$:   TYPE    ,UNTOFF         ;DRIVE OFFLINE
124 006162  000413                        BR      9$              ;PRINT DRIVE TYPE
125
126 006164  005737  001332          7$:   TST     XXDP            ;LOADED FROM RM80 ?
127 006170  001406                        BEQ     8$              ;NO
128 006172  123701  001332                CMPB    XXDP,R1         ;IS THIS THE DRIVE ?
129 006176  001360                        BNE     5$              ;BR IF NO
130 006200  104401  063312                TYPE    ,LODEV          ;DRIVE IS LOAD DEVICE
131 006204  000755                        BR      5$
132
133 006206  104401  063374          8$:   TYPE    ,UNTON          ;DRIVE ONLINE
134 006212  104401  063412          9$:   TYPE    ,BLNKS2         ;TYPE 2 BLANKS
135 006216  104401                        TYPE                    ;PRINT DRIVE TYPE
136 006220  000000                10$:   .WORD   0               ;MESSAGE ADDRESS HERE
137
138 006222  005201                11$:   INC     R1              ;INCREMENT THE DRIVE ADDRESS
139 006224  020127  000007                CMP     R1,#7           ;ALL DRIVES ARE CHECKED ?
140 006230  003637                        BLE     1$              ;BRANCH IF NOT
141
142 006232  104401  001217          12$:  TYPE    ,$CRLF          ;CR-LF
143 006236  000137  006724                JMP     CMNSTART        ;JUMP TO COMMON START
```

```
                                        .SBTTL  STANDALONE INPUT ROUTINES
                                
006242                                  STANDALONE:
006242  004737  060470                          JSR     PC,$TKINT       ;INITIALIZE CONSOLE
                                
006246  005227  177777                          INC     #-1             ;FIRST TIME THRU HERE ?
006252  001023                          BNE     2$              ;BR IF NO
                                
                                        ;SEE IF OPERATOR WANTS HELP TEXT
                                
006254  104401  062624                  TYPE    ,MSHELP         ;WANT HELP ?
006260  104411                          RDCHR                   ;GET RESPONSE
006262  012637  001176                  MOV     (SP)+,$TMP1     ;SAVE AND ECHO RESPONSE
006266  123727  001176  000131          CMPB    $TMP1,#'Y       ;WAS IT A YES RESPONSE ?
006274  001005                          BNE     1$              ;NO
006276  104401  001176                  TYPE    ,$TMP1          ;TYPE 'Y'
006302  104401  104312                  TYPE    ,HELP           ;YES - TYPE HELP TEXT
006306  000414                          BR      3$
006310  104401  063404          1$:     TYPE    ,N              ;TYPE 'N'
006314  104401  001217                  TYPE    ,$CRLF          ;CR-LF
006320  000407                          BR      3$
                                
                                        ;SEE IF USER WANTS TO CHANGE UNIBUS ADDRESS
006322                                  2$:
006322  005737  001330                  TST     CHGADR          ;CHANGE RH/RM BUS ADDRESS ?
006326  001457                          BEQ     7$              ;BR IF NO
006330  005037  001330                  CLR     CHGADR          ;NO CHANGE NEXT TIME
006334  104401  001217                  TYPE    ,$CRLF          ;CR-LF
                                
                                        ;DIALOGUE TO CHANGE THE UNIBUS ADDRESS, VECTOR ADDRESS AND INTERRUPT PRIORITY
006340                                  3$:
006340  104401  062655                  TYPE    ,CNSL01         ;TYPE CURRENT BUS ADDRESS
006344  013746  001276                  MOV     $BASE,-(SP)     ;;SAVE $BASE FOR TYPEOUT
006350  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
006352  104401  063412                  TYPE    ,BLNKS2         ;TYPE 2 BLANKS
006356  104413                          RDOCT                   ;GET NEW BUS ADDRESS
006360  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN ?
006364  001412                          BEQ     5$              ;YES-SKIP TO NEXT ENTRY
006366  022737  160000  001176          CMP     #160000,$TMP1   ;BASE ADDRESS IN I/O PAGE ?
006374  101403                          BLOS    4$              ;YES
006376  104401  062665                  TYPE    ,CNSL02         ;TYPE WARNING MESSAGE
006402  000756                          BR      3$              ;TRY AGAIN
006404  013737  001176  001276  4$:     MOV     $TMP1,$BASE     ;STORE NEW BUS ADDRESS
                                
006412  104401  062727          5$:     TYPE    ,CNSL03
006416  005046                          CLR     -(SP)
006420  113716  001272                  MOVB    $VECT1,(SP)     ;GET CURRENT VECTOR ADDRESS
006424  104402                          TYPOC
006426  104401  063412                  TYPE    ,BLNKS2         ;TYPE 2 BLANKS
006432  104413                          RDOCT                   ;GET NEW VECTOR ADDRESS
006434  012637  001176                  MOV     (SP)+,$TMP1     ;CARRIAGE RETURN?
006440  001412                          BEQ     7$              ;YES-SKIP TO NEXT ENTRY
006442  022737  001000  001176          CMP     #1000,$TMP1     ;VECTOR ADDRESS < 1000 ?
006450  101003                          BHI     6$              ;YES!!
006452  104401  062736                  TYPE    ,CNSL04         ;TYPE WARNING MESSAGE
006456  000755                          BR      5$              ;RETRY
006460  113737  001176  001272  6$:     MOVB    $TMP1,$VECT1    ;STORE NEW VECTOR ADDRESS
```

```
                                        ;DIALOGUE TO INPUT DEVICE NUMBERS
006466  005227  177777          7$:     INC     #-1             ;FIRST TIME THRU ?
006472  001002                          BNE     8$              ;BR IF NO
006474  104401  062772                  TYPE    ,CNSL07         ;TYPE INPUT INSTRUCTIONS
006500  104401  001217          8$:     TYPE    ,$CRLF          ;CR-LF
006504  005037  001300          9$:     CLR     $DEVM           ;CLEAR DEVICE MAP
006510  104401  063212                  TYPE    ,MSDRVS         ;TYPE 'DRIVE(S): '
006514  104411                          RDCHR
006516  012637  001176                  MOV     (SP)+,$TMP1     ;GET RESPONSE
006522  023727  001176  000101          CMP     $TMP1,#'A       ;IS INPUT ''A'' ?
006530  001007                          BNE     10$             ;NO
006532  104401  062610                  TYPE    ,ALL            ;YES, TYPE ''ALL'' AND GO
006536  012737  000377  001300          MOV     #377,$DEVM      ;SET DEVICE MAP FOR ALL DRIVES
006544  000137  005706                  JMP     XSIZ            ;AUTO SIZE.

006550  023727  001176  000015  10$:    CMP     $TMP1,#CR       ;CARRIAGE RETURN ?
006556  001436                          BEQ     12$             ;YES
006560  104401  001176                  TYPE    ,$TMP1          ;ECHO RESPONSE
006564  023727  001176  000060          CMP     $TMP1,#'0       ;NUMBER < 0 ?
006572  002430                          BLT     12$             ;YES
006574  023727  001176  000067          CMP     $TMP1,#'7       ;NUMBER > 7 ?
006602  003427                          BLE     13$             ;NO
006604  000423                          BR      12$             ;ILLEGAL INPUT

006606  104411                  11$:    RDCHR
006610  012637  001176                  MOV     (SP)+,$TMP1     ;GET RESPONSE
006614  023727  001176  000015          CMP     $TMP1,#CR       ;CARRIAGE RETURN ?
006622  001432                          BEQ     14$             ;YES
006624  104401  062621                  TYPE    ,COMMA          ;TYPE ', '
006630  104401  001176                  TYPE    ,$TMP1          ;ECHO RESPONSE
006634  023727  001176  000060          CMP     $TMP1,#'0       ;NUMBER < 0 ?
006642  002404                          BLT     12$             ;YES
006644  023727  001176  000067          CMP     $TMP1,#'7       ;NUMBER > 7 ?
006652  003403                          BLE     13$             ;NO
006654  104401  063134          12$:    TYPE    ,CNSL08         ;TYPE '' ?ILLEGAL INPUT''
006660  000711                          BR      9$              ;RETRY

006662  013701  001176          13$:    MOV     $TMP1,R1        ;R1 - DRIVE NUMBER
006666  042701  177770                  BIC     #^C7,R1
006672  156137  063516  001300          BISB    ATNTBL(R1),$DEVM        ;SET DEVICE IN MAP
006700  122737  000377  001300          CMPB    #377,$DEVM      ;DONE ?
006706  101337                          BHI     11$             ;NO
006710  005237  001326          14$:    INC     AUTSIZ          ;DO NOT AUTO SIZE WHEN TYPING DRIVE STATUS
006714  104401  001217                  TYPE    ,$CRLF          ;CR-LF
006720  000137  005706                  JMP     XSIZ            ;GO SIZE DEVICES
```

```
                                        ;ASSEMBLE TEST QUE FROM DEVICE MAP
006724                                  CMNSTART:
006724  104401  063156                        TYPE    ,DRIVES         ;TYPE 'DRIVE(S) TO BE TESTED'
006730  013700  001300                        MOV     $DEVM,R0        ;R0 = DEVICE MAP
006734  001004                              BNE     1$              ;BR IF DRIVES TO TEST
006736  104401  062621                        TYPE    ,COMMA          ;TYPE ', '
006742  104401  063205                        TYPE    ,NONE           ;TYPE 'NONE'
006746  012701  001470                1$:     MOV     #TSTQUE+2,R1    ;R1 = ADDRESS OF FIRST ENTRY IN QUE
006752  010137  001466                        MOV     R1,TSTQUE       ;INITIALIZE ENTRY POINTER
006756  012702  000001                        MOV     #1,R2           ;R2 = DEVICE POINTER
006762  005003                              CLR     R3              ;R3 = DEVICE NUMBER
006764  030200                        2$:     BIT     R2,R0           ;IS THIS DEVICE IN MAP ?
006766  001413                              BEQ     3$              ;NO !!
006770  104401  062621                        TYPE    ,COMMA          ;TYPE ', '
006774  010311                              MOV     R3,(R1)         ;YES - ENTER DEVICE NUMBER IN QUE
006776  010346                              MOV     R3,-(SP)        ;;SAVE R3 FOR TYPEOUT
007000  104403                              TYPOS                   ;;GO TYPE--OCTAL ASCII
007002     001                              .BYTE   1               ;;TYPE 1 DIGIT(S)
007003     000                              .BYTE   0               ;;SUPPRESS LEADING ZEROS
007004  116361  063516  000001                MOVB    ATNTBL(R3),1(R1);ENTER ATTENTION BIT IN QUE
007012  062701  000002                        ADD     #2,R1           ;ADVANCE ENTRY POINTER
007016  006302                        3$:     ASL     R2              ;ADVANCE DEVICE POINTER
007020  105702                              TSTB    R2              ;DONE ALL DEVICES ?
007022  001402                              BEQ     4$              ;YES
007024  005203                              INC     R3              ;ADVANCE DEVICE NUMBER
007026  000756                              BR      2$              ;ENTER NEXT DEVICE
007030  005011                        4$:     CLR     (R1)            ;TERMINATE TEST QUE
007032  104401  001217                        TYPE    ,$CRLF          ;TYPE CR-LF

                                        ;SIZE FOR CLOCK
007036  004737  054040                        JSR     PC,SIZCLK       ;SEE IF CLOCK PRESENT
007042  000425                              BR      6$              ;YES - CLOCK IS PRESENT
007044  104401  007052                        TYPE    ,65$            ;;TYPE ASCIZ STRING
007050  000413                              BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <CRLF>/NO 'L' OR 'P' CLOCK/
007100                                  64$:
007100  005737  000042                        TST     @#42            ;ANY MONITOR PRESENT ?
007104  001002                              BNE     5$              ;BR IF YES
007106  000137  004652                        JMP     START           ;JUMP TO START
007112  000137  054010                5$:     JMP     $GET42          ;RETURN CONTROL TO MONITOR
007116  000413                        6$:     BR      READY1

007120  000240                        READY:  NOP                     ;READY TO START TEST
007122  105737  001300                        TSTB    $DEVM           ;ANY DRIVES IN MAP ?
007126  001007                              BNE     2$              ;BR IF YES
007130  005737  000042                        TST     @#42            ;ANY MONITOR PRESENT ?
007134  001002                              BNE     1$              ;BR IF YES
007136  000137  004652                        JMP     START           ;JUMP TO START
007142  000137  054010                1$:     JMP     $GET42          ;RETURN CONTROL TO MONITOR
007146                                  2$:

007146  105037  001116                READY1: CLRB    $TSTNM          ;RESET TEST NUMBER
007152  005037  001206                        CLR     $TIMES          ;INITIALIZE NUMBER OF ITERATIONS
007156  004737  060470                        JSR     PC,$TKINT       ;INITIALIZE TTY
007162  012746  000000                        MOV     #PR0,-(SP)      ;;PUT NEW PS ON STACK
007166  012746  007174                        MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
007172  000002                              RTI                     ;;POP NEW PC AND PS
```

```
                007174                          64$:
     50 007174  117737  172266  001234                 MOVB    @TSTQUE,$UNIT   ;LOAD DRIVE NUMBER
     51
     52                                         ;CLEAR MASSBUS CONTROLLER, SELECT DRIVE AND DETERMINE THE LAST TRACK
     53                                         ;OF THE DIFFERENT DRIVE TYPES
     54
     55 007202  012737  002000  001334                 MOV     #TA4,LSTRK      ;ASSUME LAST TRACK FOR RM02/3 - 4.
     56 007210  013700  001276                         MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
     57 007214  012760  000040  000010                 MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
     58 007222  117760  172240  000010                 MOVB    @TSTQUE,RMCS2(R0)       ;SELECT DEVICE UNDER TEST
     59 007230  016002  000026                         MOV     RMDT(R0),R2     ;GET RMDT AND
     60 007234  042702  177770                         BIC     #177770,R2      ;SAVE DRIVE TYPE BITS
     61 007240  022702  000007                         CMP     #7,R2           ;IS IT AN RM05 ?
     62 007244  001003                                 BNE     3$              ;NO, MUST BE AN RM02 OR RM03
     63 007246  012737  011000  001334                 MOV     #TA16!TA2,LSTRK ;YES--SET LAST TRACK = 18.
     64
     65                                         ;TYPE DRIVE NUMBER TO BE TESTED($UNIT)
     66
     67 007254  104401  001217                 3$:     TYPE    ,$CRLF          ;CR-LF
     68 007260  104401  063226                         TYPE    ,MSGDRV         ;TYPE 'DRIVE'
     69 007264  013746  001234                         MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                               ;;TYPE DRIVE NUMBER
        007270  104403                              TYPOS                      ;;GO TYPE--OCTAL ASCII
        007272     002                              .BYTE   2                  ;;TYPE 2 DIGIT(S)
        007273     000                              .BYTE   0                  ;;SUPPRESS LEADING ZEROS
     70 007274  005004                              CLR     R4                 ;THESE TWO LOOPS ARE ADDED TO
     71 007276  005304                              DEC     R4                 ;WAIT FOR TTY
     72 007300  001376                              BNE     .-2
     73 007302  005304                              DEC     R4
     74 007304  001376                              BNE     .-2
```

```
                        .SBTTL  REGISTER AND STORAGE USAGE

                        ;REGISTER ASSIGNMENTS
                        ;
                        ;R0     =       UNIBUS ADDRESS OF RH CONTROLLER
                        ;R1     -       ADDRESS OF ENTRY IN TEST QUE CORRESPONDING TO THE
                        ;               UNIT UNDER TEST
                        ;R2,R3  -       WORKING REGISTERS FOR TEST IN PROGRESS, MUST BE
                        ;               SAVED BY SUBROUTINES
                        ;R4,R5          GENERAL WORKING REGISTERS, ARE NOT SAVED BY
                        ;               SUBROUTINES
                        ;R6     -       STACK POINTER
                        ;R7     -       LINKAGE REGISTER TO SUBROUTINES
                        ;
                        ;STORAGE ASSIGNMENTS
                        ;
                        ;$TMP0-$TMP4    TEMPORARY STORAGE, NOT SAVED BY SUBROUTINES
                        ;$GDDAT,$BDDAT  EXPECTED AND RRECEIVED STATUS FOR ERROR TYPEOUT
                        ;$GDADR,$BDADR  ADDRESS OF EXPECTED AND RECEIVED STATUS IF APPLICABLE,
                        ;               ALSO THE ADDRESS OF A REGISTER ERROR
                        ;
                        ;$TSTN  -       TEST NUMBER
                        ;$UNIT  =       NUMBER OF DEVICE BEING TESTED
                        ;RGINBF =       THE REGISTER INPUT BUFFER HAS A STORAGE LOCATION FOR
                        ;               EACH REGISTER, AND IS USED WHEN READING STATUS AND
                        ;               CONTROL DATA
                        ;RGOTBF =       THE REGISTER OUTPUT BUFFER HAS A STORAGE LOCATION FOR
                        ;               EACH REGISTER, AND IS USED FOR ASSEMBLING DATA TO BE
                        ;               WRITTEN IN REGISTERS
```

```
                                ;;*****************************************************************
                                ;*TEST 1         TRANSFER TEST
                                ;;*****************************************************************
007306                          TST1:
007306  000004                          SCOPE                   ;SCOPE CALL
007310  000240                          NOP
007312  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
007316  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
007322  013701  001466                  MOV     TS.QUE,R1       ;R1 = POINTER TO DEVICE
007326  012737  000001  001226          MOV     #1,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
                2
 3 007334  012702  000000              MOV     #0,R2           ;R2 - REGISTER INDEX
                4
                5                       ;CLEAR THE MASSBUS AND VERIFY THAT NONEXISTANT DEVICE ERROR IS RESET
 6 007340                               10$:
 7 007340  004737  054674              JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 8 007344  016037  000010  001142      MOV     RMCS2(R0),$BDDAT        ;STORE RMCS2 AT $BDDAT
 9 007352  032737  010000  001142      BIT     #NED,$BDDAT
10 007360  001417                      BEQ     20$
11 007362  111137  001140              MOVB    (R1),$GDDAT
12 007366  042737  177770  001140      BIC     #^CUNTMSK,$GDDAT
13 007374  052737  000100  001140      BIS     #IR,$GDDAT
14 007402  010037  001136              MOV     R0,$BDADR
15 007406  062737  000010  001136      ADD     #RMCS2,$BDADR
16 007414  104001                      EMT     1
17 007416  000475                      BR      60$
18
19                                      ;READ THE REGISTER WHOSE INDEX IS IN R2 AND EXIT TEST IF THE READ
20                                      ;DOES NOT SET 'NED' ERROR
21 007420                               20$:
22 007420  010003                      MOV     R0,R3           ;R3 - REGISTER ADDRESS
23 007422  060203                      ADD     R2,R3
24 007424  011304                      MOV     (R3),R4         ;READ REGISTER
25 007426  032760  010000  000010      BIT     #NED,RMCS2(R0)  ;IS 'NED' SET??
26 007434  001470                      BEQ     70$             ;NO!!
27
28 007436  004737  054674              JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
29 007442  016037  000010  001142      MOV     RMCS2(R0),$BDDAT        ;STORE RMCS2 AT $BDDAT
30 007450  032737  010000  001142      BIT     #NED,$BDDAT
31 007456  001417                      BEQ     30$
32 007460  111137  001140              MOVB    (R1),$GDDAT
33 007464  042737  177770  001140      BIC     #^CUNTMSK,$GDDAT
34 007472  052737  000100  001140      BIS     #IR,$GDDAT
35 007500  010037  001136              MOV     R0,$BDADR
36 007504  062737  000010  001136      ADD     #RMCS2,$BDADR
37 007512  104001                      EMT     1
38 007514  000436                      BR      60$
39
40                                      ;WRITE THE REGISTER WHOSE INDEX IS IN R2 AND EXIT TEST IF THE WRITE
41                                      ;DOES NOT SET 'NED' ERROR
42 007516                               30$:
43 007516  012713  000000              MOV     #0,(R3)         ;WRITE REGISTER
44 007522  032760  010000  000010      BIT     #NED,RMCS2(R0)  ;IS 'NED' SET??
45 007530  001432                      BEQ     70$             ;NO!!
46
47                                      ;COULD NOT READ OR WRITE THE REGISTER WITHOUT SETING 'NED' ERROR -
```

```
                                        ;ADVANCE THE REGISTER INDEX AND REPEAT THE TEST FOR THE NEXT
                                        ;AVAILABLE DEVICE REGISTER
007532                                  40$:
007532  062702  000002                        ADD     #2,R2           ;ADVANCE TO NEXT REGISTER
007536  022702  000002                        CMP     #RMWC,R2        ;IS THIS RMWC??
007542  001773                              BEQ     40$             ;YES - TRY NEXT REGISTER
007544  022702  000004                        CMP     #RMBA,R2        ;IS THIS RMBA??
007550  001770                              BEQ     40$             ;YES - TRY NEXT REGISTER
007552  022702  000010                        CMP     #RMCS2,R2       ;IS THIS RMCS5??
007556  001765                              BEQ     40$             ;YES - TRY ANOTHER REGISTER
007560  022702  000016                        CMP     #RMAS,R2        ;IS THIS RMAS ??
007564  001762                              BEQ     40$             ;YES - TRY ANOTHER REGISTER
007566  022702  000022                        CMP     #RMDB,R2        ;IS THIS RMDB??
007572  001757                              BEQ     40$             ;YES - TRY ANOTHER REGISTER
007574  022702  000046                        CMP     #RMEC2,R2       ;IS THIS A LEGAL REGISTER
007600  103257                              BHIS    10$             ;YES - TRY THIS REGISTER

                                        ;GOT 'NONEXISTENT DEVICE' ERROR FOR EVERY REMOTE REGISTER ADDRESS
007602                                  50$:
007602  013737  001276  001136              MOV     $BASE,$BDADR    ;STORE BASE ADDRESS
007610  104002                              EMT     2
007612  000137  053564          60$:        JMP     $EOSP           ;GO SELECT NEXT DEVICE

007616                                  70$:

                                        ;;*****************************************************************
                                        ;*TEST 2         CTOD TEST

                                        ;;*****************************************************************
007616                                  TST2:
007616  000004                              SCOPE                   ;SCOPE CALL
007620  000240                              NOP
007622  012706  001100                      MOV     #STACK,SP       ;LOAD THE STACK POINTER
007626  013700  001276                      MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
007632  013701  001466                      MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
007636  012737  000002  001226              MOV     #2,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

007644  004737  054674                      JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER

                                        ;WRITE ONES IN REMOTE REGISTERS
007650  012760  000076  000000              MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
007656  012760  177777  000006              MOV     #-1,RMDA(R0)    ;LOAD RMDA
007664  012760  001777  000034              MOV     #CYLMSK,RMDC(R0)        ;LOAD RMDC
007672  012760  016200  000032              MOV     #^CXNUOF,RMOF(R0)       ;LOAD RMOF

                                        ;READ REMOTE REGISTERS TWICE
007700  012702  000001                      MOV     #1,R2
007704                                  10$:
007704  016037  000000  001336              MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
007712  016037  000006  001344              MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
007720  016037  000034  001372              MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
007726  016037  000032  001370              MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
007734  005302                              DEC     R2
007736  100362                              BPL     10$

                                        ;SEE IF ANY ONE BITS CAME BACK
007740  042737  177701  001336              BIC     #^CILF76,RMCS1I ;IS RMCS1 0??
```

```
 94 007746  001014                          BNE     20$             ;NO!!
 95 007750  005737  C01344                  TST     RMDAI           ;IS RMDA 0??
 96 007754  001011                          BNE     20$             ;NO!!
 97 007756  042737  176000  001372          BIC     #XNUDC,RMDCI    ;IS RMDC 0??
 98 007764  001005                          BNE     20$             ;NO!!
 99 007766  042737  161577  001370          BIC     #XNUOF,RMOFI    ;IS RMOF 0 ??
100 007774  001001                          BNE     20$             ;NO!!
101
102                                 ;CANNOT READ/WRITE ANY ONE FROM REMOTE REGISTER
103 007776  104003                          EMT     3
104 010000                          20$:
105
106                                 ;;*****************************************************************
                                    ;*TEST 3         MASSBUS INITIALIZE TEST

                                    ;;*****************************************************************
    010000                          TST3:
    010000  000004                          SCOPE                   ;SCOPE CALL
    010002  000240                          NOP
    010004  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
    010010  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
    010014  013701  001466                  MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
    010020  012737  000003  001226          MOV     #3,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX
107
108 010026  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
109
110                                 ;WRITE ONES IN SELECTED REGISTERS
111 010032  012760  000076  000000          MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
112 010040  012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
113 010046  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
114
115                                 ;INITIALIZE MASSBUS WITH A CLEAR
116 010054  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
117
118                                 ;READ THE REGISTERS THAT WERE WRITTEN
119 010060  016037  000000  001336          MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
120 010066  016037  000014  001352          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
121 010074  016037  000042  001400          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
122
123                                 ;SEE IF ANY REGISTER BITS WERE CLEARED
124 010102  052737  177701  001336          BIS     #^CILF76,RMCS1I ;SET ANY BIT NOT WRITTEN
125 010110  052737  001567  001400          BIS     #XNUER2,RMER2I
126 010116  022737  177777  001336          CMP     #-1,RMCS1I      ;ANY ZEROS IN RMCS1??
127 010124  001011                          BNE     10$             ;YES!!
128 010126  022737  177777  001352          CMP     #-1,RMER1I      ;ANY ZEROS IN RMER1??
129 010134  001005                          BNE     10$             ;YES!!
130 010136  022737  177777  001400          CMP     #-1,RMER2I      ;ANY ZEROS IN RMER2??
131 010144  001001                          BNE     10$
132
133                                 ;NONE OF THE BITS WERE CLEARED
134 010146  104004                          EMT     4
135 010150                          10$:
136
137                                 ;;*****************************************************************
                                    ;*TEST 4         CLEAR STUCK ACTIVE TEST

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

```
010150                                  TST4:
010150  000004                                  SCOPE                           ;SCOPE CALL
010152  000240                                  NOP
010154  012706  001100                          MOV     #STACK,SP               ;LOAD THE STACK POINTER
010160  013700  001276                          MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
010164  013701  001466                          MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
010170  012737  000004  001226                  MOV     #4,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

010176  004737  054674                          JSR     PC,CNTCLR               ;GO CLEAR CONTROLLER

                                        ;WRITE ONES IN TEST REGISTERS
010202  012760  177777  000014                  MOV     #-1,RMER1(R0)           ;LOAD RMER1
010210  012760  177777  000042                  MOV     #-1,RMER2(R0)           ;LOAD RMER2
010216  012760  000001  000024                  MOV     #DMD,RMMR1(R0)          ;LOAD RMMR1

                                        ;READ TEST REGISTERS AND SEE IF ANY BITS ARE ON
010224  016037  000014  001352                  MOV     RMER1(R0),RMER1I                ;STORE RMER1 IN INPUT BUFFER
010232  016037  000042  001400                  MOV     RMER2(R0),RMER2I                ;STORE RMER2 IN INPUT BUFFER
010240  016037  000024  001362                  MOV     RMMR1(R0),RMMR1I                ;STORE RMMR1 IN INPUT BUFFER
010246  042737  040000  001352                  BIC     #UNS,RMER1I             ;DONT ACCEPT UNSAFE
010254  001011                                  BNE     10$                     ;BRANCH IF ANY OTHER BITS ON
010256  042737  040200  001400                  BIC     #SKI!DVC,RMER2I         ;DONT ACCEPT SKI OR DVC
010264  001005                                  BNE     10$                     ;BRANCH IF ANY OTHER BITS ON
010266  032737  000001  001362                  BIT     #DMD,RMMR1I             ;BRANCH IF DMD IS ON
010274  001001                                  BNE     10$
010276  104026                                  EMT     26
010300                                  10$:


                                        ;;*******************************************************************
                                        ;;*TEST 5         TRISTATE TRANSFER TEST

                                        ;;*******************************************************************
010300                                  TST5:
010300  000004                                  SCOPE                           ;SCOPE CALL
010302  000240                                  NOP
010304  012706  001100                          MOV     #STACK,SP               ;LOAD THE STACK POINTER
010310  013700  001276                          MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
010314  013701  001466                          MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
010320  012737  000005  001226                  MOV     #5,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

010326  005002                                  CLR     R2                      ;CLEAR ERROR FLAGS
010330  004737  054674                          JSR     PC,CNTCLR               ;GO CLEAR CONTROLLER

                                        ;WRITE ONES IN SELECTED REGISTERS
010334  012760  000076  000000                  MOV     #ILF76,RMCS1(R0)                ;LOAD RMCS1
010342  012760  177777  000006                  MOV     #-1,RMDA(R0)            ;LOAD RMDA
010350  012760  177777  000014                  MOV     #-1,RMER1(R0)           ;LOAD RMER1
010356  012760  177777  000032                  MOV     #-1,RMOF(R0)            ;LOAD RMOF
010364  012760  177777  000042                  MOV     #-1,RMER2(R0)           ;LOAD RMER2

                                        ;WRITE ZEROS IN SELECTED REGISTERS
010372  012760  000000  000000                  MOV     #0,RMCS1(R0)            ;LOAD RMCS1
010400  012760  000000  000006                  MOV     #0,RMDA(R0)             ;LOAD RMDA
010406  012760  000000  000014                  MOV     #0,RMER1(R0)            ;LOAD RMER1
010414  012760  000000  000032                  MOV     #0,RMOF(R0)             ;LOAD RMOF
010422  012760  000000  000034                  MOV     #0,RMDC(R0)             ;LOAD RMDC
010430  012760  000000  000042                  MOV     #0,RMER2(R0)            ;LOAD RMER2
```

```
178
179                                     ;READ BACK ALL REGISTERS
180 010436  016037  000000  001336            MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
181 010444  016037  000006  001344            MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
182 010452  016037  000014  001352            MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
183 010460  016037  000032  001370            MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
184 010466  016037  000034  001372            MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
185 010474  016037  000042  001400            MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
186
187                                     ;CHECK EACH REGISTER CONTENT FOR ZERO BITS WRITTEN & READ
188 010502  012702  177777                    MOV     #-1,R2          ;ACCUMULATE ZEROS IN R2
189 010506  052737  177701  001336            BIS     #^CILF76,RMCS1I ;SET ALL BITS NOT WRITTEN
190 010514  052737  161577  001370            BIS     #XNUOF,RMOFI
191 010522  052737  176000  001372            BIS     #XNUDC,RMDCI
192 010530  052737  001567  001400            BIS     #XNUER2,RMER2I
193 010536  005137  001336                    COM     RMCS1I          ;COMPLEMENT REGISTER CONTENTS
194 010542  005137  001344                    COM     RMDAI
195 010546  005137  001352                    COM     RMER1I
196 010552  005137  001370                    COM     RMOFI
197 010556  005137  001372                    COM     RMDCI
198 010562  005137  001400                    COM     RMER2I
199 010566  043702  001336                    BIC     RMCS1I,R2               ;ACCUMULATE ALL ZERO BITS
200 010572  043702  001344                    BIC     RMDAI,R2
201 010576  043702  001352                    BIC     RMER1I,R2
202 010606  043702  001370                    BIC     RMOFI,R2
203 010606  043702  001372                    BIC     RMDCI,R2
204 010612  043702  001400                    BIC     RMER2I,R2
205 010616  001407                            BEQ     10$             ;BRANCH IF EACH BIT IS ZERO
206
207                                     ;ONE OR MORE BIT POSITIONS ARE NOT ZERO
208 010620  010237  001142                    MOV     R2,$BDDAT               ;SAVE RESULT FOR TYPE
209 010624  005037  001140                    CLR     $GDDAT          ;LOAD EXPECTED RESULT
210 010630  104005                            EMT     5
211 010632  052702  000001                    BIS     #BIT0,R2        ;SET ERROR FLAG
212 010636                              10$:
    010636  004737  054674                    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
213
214                                     ;PRESET SELECTED REGISTERS TO ZEROS
215                                     ;(ASSUME RMCS1, RMER1, RMER2 WERE CLEARED BY INIT)
216 010642  012760  000000  000006            MOV     #0,RMDA(R0)     ;LOAD RMDA
217 010650  012760  000000  000032            MOV     #0,RMOF(R0)     ;LOAD RMOF
218 010656  012760  000000  000034            MOV     #0,RMDC(R0)     ;LOAD RMDC
219
220                                     ;WRITE ONES IN SELECTED REGISTERS
221 010664  012760  000076  000000            MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
222 010672  012760  177777  000006            MOV     #-1,RMDA(R0)    ;LOAD RMDA
223 010700  012760  016200  000032            MOV     #^CXNUOF,RMOF(R0)       ;LOAD RMOF
224 010706  012760  001777  000034            MOV     #^CXNUDC,RMDC(R0)       ;LOAD RMDC
225 010714  012760  177777  000014            MOV     #-1,RMER1(R0)   ;LOAD RMER1
226 010722  012760  176210  000042            MOV     #^CXNUER2,RMER2(R0)     ;LOAD RMER2
227
228                                     ;READ ALL REGISTERS
229 010730  016037  000000  001336            MOV     RMCS1(R0),RMCS1I        ;STORE RMCS1 IN INPUT BUFFER
230 010736  016037  000006  001344            MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
231 010744  016037  000032  001370            MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
232 010752  016037  000034  001372            MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
233 010760  016037  000014  001352            MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
```

```
010766  016037  000042  001400          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER

                                ;CHECK EACH REGISTER CONTENT FOR ONE BITS WRITTEN & READ
010774  042737  177701  001336          BIC     #^CILF76,RMCS1I         ;CLEAR ALL BITS NOT WRITTEN
011002  042737  161577  001370          BIC     #XNUOF,RMOFI
011010  042737  176000  001372          BIC     #XNUDC,RMDCI
011016  042737  001567  001400          BIC     #XNUER2,RMER2I
011024  005002                          CLR     R2              ;ACCUMULATE ONES IN R2
011026  053702  001336                  BIS     RMCS1I,R2               ;ACCUMULATE ALL ONE BITS
011032  053702  001344                  BIS     RMDAI,R2
011036  053702  001370                  BIS     RMOFI,R2
011042  053702  001372                  BIS     RMDCI,R2
011046  053702  001352                  BIS     RMER1I,R2
011052  053702  001400                  BIS     RMER2I,R2
011056  022702  177777                  CMP     #-1,R2          ;SEE IF EACH BIT POSITION WAS ONE
011062  001410                          BEQ     20$             ;BRANCH IF NONE STUCK

                                ;ONE OR MORE BIT POSITIONS ARE NOT ONE
011064  010237  001142                  MOV     R2,$BDDAT               ;SAVE RESULT FOR TYPE
011070  012737  177777  001140          MOV     #-1,$GDDAT              ;EXPECTED RESULT
011076  104006                          EMT     6
011100  052702  000002                  BIS     #BIT1,R2        ;SET ERROR FLAG
011104                          20$:
011104  005702                          TST     R2              ;ANY ERRORS DETECTED ??
011106  001126                          BNE     30$             ;YES - DONT DO BIT TEST
011110  012702  000001                  MOV     #1,R2           ;R2=BIT POSITION
011114                          25$:
011114  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER

                                ;WRITE THE BIT PATTERN IN SELECTED DEVICE REGISTERS
011120  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
011124  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
011130  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
011134  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
011140  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2

                                ;READ BACK THE REGISTERS
011144  016037  000006  001344          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
011152  016037  000032  001370          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
011160  016037  000034  001372          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
011166  016037  000014  001352          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
011174  016037  000042  001400          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER

                                ;CHECK REGISTER CONTENTS FOR CORRECT PATTERN
011202  005003                          CLR     R3              ;R3-ACCUMULATED ONE BIT
011204  012704  177777                  MOV     #-1,R4          ;R4-ACCUMULATED ZERO BITS
011210  013705  001344                  MOV     RMDAI,R5                ;GET ANY GOOD BITS FROM RMDA
011214  050503                          BIS     R5,R3
011216  005105                          COM     R5
011220  040504                          BIC     R5,R4
011222  013705  001370                  MOV     RMOFI,R5                ;GET GOOD BITS FROM RMOF
011226  042705  161577                  BIC     #XNUOF,R5
011232  050503                          BIS     R5,R3
011234  005105                          COM     R5
011236  042705  161577                  BIC     #XNUOF,R5
011242  040504                          BIC     R5,R4
011244  013705  001372                  MOV     RMDCI,R5                ;GET GOOD BITS FROM RMDC
```

```
011250  042705  176000          BIC     #XNUDC,R5
011254  050503                  BIS     R5,R3
011256  005105                  COM     R5
011260  042705  176000          BIC     #XNUDC,R5
011264  040504                  BIC     R5,R4
011266  013705  001352          MOV     RMER1I,R5               ;GET GOOD BITS FROM RMER1
011272  050503                  BIS     R5,R3
011274  005105                  COM     R5
011276  040504                  BIC     R5,R4
011300  013705  001400          MOV     RMER2I,R5               ;GET GOOD BITS FROM RMER2
011304  042705  001567          BIC     #XNUER2,R5
011310  050503                  BIS     R5,R3
011312  005105                  COM     R5
011314  042705  001567          BIC     #XNUER2,R5
011320  040504                  BIC     R5,R4
011322  010205                  MOV     R2,R5           ;RESET ALL ONES IN R3 EXCEPT
011324  005105                  COM     R5              ;FOR THE TEST BIT
011326  040503                  BIC     R5,R3
011330  040204                  BIC     R2,R4           ;RESET TEST BIT IN R4
011332  050403                  BIS     R4,R3           ;COMBINE ACCUMULATED 1'S + 0'S
011334  020302                  CMP     R3,R2           ;IS PATTERN OK??
011336  001406                  BEQ     26$             ;YES!!
011340  010237  001140          MOV     R2,$GDDAT               ;SAVE TEST PATTERN
011344  010337  001142          MOV     R3,$BDDAT               ;SAVE RESULT
011350  104007                  EMT     7
011352  000404                  BR      30$             ;SKIP TO NEXT

                        ;ADVANCE R2 TO THE NEXT PATTERN AND REPEAT TEST
011354                  26$:
011354  006302                  ASL     R2              ;SHIFT THE BIT
011356  001402                  BEQ     30$             ;EXIT IF DONE
011360  000137  011114          JMP     25$
011364                  30$:

                        ;;*****************************************************************
                        ;*TEST 6        REGISTER SELECT TEST
                        ;*
                        ;*NOTE: REGISTER SELECT 16 IS TESTED BY THE ''ILR'' TEST
                        ;*
                        ;;*****************************  *****************************
011364                  TST6:
011364  000004                  SCOPE                   ;SCOPE CALL
011366  000240                  NOP
011370  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
011374  013700  001276          MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
011400  013701  001466          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
011404  012737  000006  001226  MOV     #6,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

                        ;THE FOLLOWING TABLE GIVES MASSBUS REGISTER SELECT VALUES FOR
                        ;EACH DEVICE REGISTER

                        ;       REGISTER        REG SEL
                        ;       NAME            (16,8,4,2,1)
                        ;
                        ;       RMCS1           00000
                        ;       RMDS            00001
                        ;       RMER1           00010
```

```
                                        ;       RMMR1           00011
                                        ;       RMAS            00100
                                        ;       RMDA            00101
                                        ;       RMDT            00110
                                        ;       RMLA            00111
                                        ;       RMSN            01000
                                        ;       RMOF            01001
                                        ;       RMDC            01010
                                        ;       RMHR            01011
                                        ;       RMMR2           01100
                                        ;       RMER2           01101
                                        ;       RMEC1           01110
                                        ;       RMEC2           01111

                                        ;EACH REGISTER SELECT LINE IS TESTED FOR A STUCK AT ONE,
                                        ;STUCK AT ZERO FAULT. AS AN EXAMPLE, TO TEST REG SEL 1,
                                        ;FOR S-A-0, RMER1 IS WRITTEN WITH ZEROS. THEN THE REGISTER
                                        ;THAT HAS THE SAME SELECT VALUE, EXCEPT FOR THE SELECT LINE
                                        ;BEING TESTED, IS WRITTEN WITH ONES. IN THIS EXAMPLE,
                                        ;RMMR1 IS WRITTEN WITH ONES. IF SELECT LINE 1 IS S-A-0,
                                        ;THE ALL ONES WORD WILL BE WRITTEN IN RMER1, AND RMER1
                                        ;WILL NOT BE 0 WHEN READ BACK.

011412  005002                          CLR     R2              ;R2= ZEROS SOURCE
011414  012703  177777                  MOV     #-1,R3          ;R3= ONES  SOURCE

                                        ;TEST REG SEL 1 FOR S-A-0
011420  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
011424  010260  000014                  MOV     R2,RMER1(R0)    ;LOAD RMER1
011430  010260  000034                  MOV     R2,RMDC(R0)     ;LOAD RMDC
011434  010360  000024                  MOV     R3,RMMR1(R0)    ;LOAD RMMR1
011440  010360  000036                  MOV     R3,RMHR(R0)     ;LOAD RMHR
011444  016037  000014  001352          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
011452  016037  000034  001372          MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
011460  020337  001352                  CMP     R3,RMER1I
011464  001007                          BNE     10$
011466  052737  176000  001372          BIS     #XNUDC,RMDCI
011474  020337  001372                  CMP     R3,RMDCI
011500  001001                          BNE     10$
011502  104010                          EMT     10

                                        ;TEST REG SEL 1 FOR S-A-1
011504                                  10$:
011504  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
011510  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
011514  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
011520  010260  000042                  MOV     R2,RMER2(R0)    ;LOAD RMER2
011524  010360  000016                  MOV     R3,RMAS(R0)     ;LOAD RMAS
011530  010360  000030                  MOV     R3,RMSN(R0)     ;LOAD RMSN
011534  010360  000040                  MOV     R3,RMMR2(R0)    ;LOAD RMMR2
011540  016037  000006  001344          MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
011546  016037  000032  001370          MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
011554  016037  000042  001400          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
011562  020337  001344                  CMP     R3,RMDAI
011566  001015                          BNE     20$
011570  052737  161577  001370          BIS     #XNUOF,RMOFI
011576  020337  001370                  CMP     R3,RMOFI
```

```
397 011602 001007                          BNE     20$
398 011604 052737 001567 001400            BIS     #XNUER2,RMER2I
399 011612 020337 001400                   CMP     R3,RMER2I
400 011616 001001                          BNE     20$
401 011620 104011                          EMT     11
402
403                                ;TEST REG SEL 2 FOR S-A-0
404 011622                         20$:
405 011622 004737 054674                   JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
406 011626 010260 000006                   MOV     R2,RMDA(R0)     ;LOAD RMDA
407 011632 010260 000032                   MOV     R2,RMOF(R0)     ;LOAD RMOF
408 011636 010260 000042                   MOV     R2,RMER2(R0)    ;LOAD RMER2
409 011642 010360 000020                   MOV     R3,RMLA(R0)     ;LOAD RMLA
410 011646 010360 000036                   MOV     R3,RMHR(R0)     ;LOAD RMHR
411 011652 010360 000046                   MOV     R3,RMEC2(R0)    ;LOAD RMEC2
412 011656 016037 000006 001344            MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
413 011664 016037 000032 001370            MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
414 011672 016037 000042 001400            MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
415 011700 020337 001344                   CMP     R3,RMDAI
416 011704 001015                          BNE     30$
417 011706 052737 161577 001370            BIS     #XNUOF,RMOFI
418 011714 020337 001370                   CMP     R3,RMOFI
419 011720 001007                          BNE     30$
420 011722 052737 001567 001400            BIS     #XNUER2,RMER2I
421 011730 020337 001400                   CMP     R3,RMER2I
422 011734 001001                          BNE     30$
423 011736 104012                          EMT     12
424
425                                ;TEST REG SEL 2 FOR S-A-1
426 011740                         30$:
427 011740 004737 054674                   JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
428 011744 010260 000014                   MOV     R2,RMER1(R0)    ;LOAD RMER1
429 011750 010260 000034                   MOV     R2,RMDC(R0)     ;LOAD RMDC
430 011754 012760 000076 000000            MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
431 011762 010360 000030                   MOV     R3,RMSN(R0)     ;LOAD RMSN
432 011766 016037 000014 001352            MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
433 011774 016037 000034 001372            MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
434 012002 052737 177701 001352            BIS     #^CILF76,RMER1I
435 012010 020337 001352                   CMP     R3,RMER1I
436 012014 001007                          BNE     40$
437 012016 052737 176000 001372            BIS     #XNUDC,RMDCI
438 012024 020337 001372                   CMP     R3,RMDCI
439 012030 001001                          BNE     40$
440 012032 104013                          EMT     13
441
442                                ;TEST REG SEL 4 FOR S-A-0
443 012034                         40$:
444 012034 004737 054674                   JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
445 012040 010260 000014                   MOV     R2,RMER1(R0)    ;LOAD RMER1
446 012044 010260 000032                   MOV     R2,RMOF(R0)     ;LOAD RMOF
447 012050 010260 000034                   MOV     R2,RMDC(R0)     ;LOAD RMDC
448 012054 010360 000026                   MOV     R3,RMDT(R0)     ;LOAD RMDT
449 012060 010360 000042                   MOV     R3,RMER2(R0)    ;LOAD RMER2
450 012064 010360 000044                   MOV     R3,RMEC1(R0)    ;LOAD RMEC1
451 012070 016037 000014 001352            MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
452 012076 016037 000032 001370            MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
453 012104 016037 000034 001372            MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFFR
```

```
 454 012112  020337  001352                    CMP     R3,RMER1I
 455 012116  001015                            BNE     50$
 456 012120  052737  161577  001370            BIS     #XNUOF,RMOFI
 457 012126  020337  001370                    CMP     R3,RMOFI
 458 012132  001007                            BNE     50$
 459 012134  052737  176000  001372            BIS     #XNUDC,RMDCI
 460 012142  020337  001372                    CMP     R3,RMDCI
 461 012146  001001                            BNE     50$
 462 012150  104014                            EMT     14
 463
 464                                    ;TEST REG SEL 4 FOR S-A-1
 465 012152                             50$:
 466 012152  004737  054674                    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 467 012156  010260  000006                    MOV     R2,RMDA(R0)     ;LOAD RMDA
 468 012162  010260  000042                    MOV     R2,RMER2(R0)    ;LOAD RMER2
 469 012166  010360  000012                    MOV     R3,RMDS(R0)     ;LOAD RMDS
 470 012172  010360  000032                    MOV     R3,RMOF(R0)     ;LOAD RMOF
 471 012176  016037  000006  001344            MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
 472 012204  016037  000042  001400            MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
 473 012212  020337  001344                    CMP     R3,RMDAI
 474 012216  001007                            BNE     60$
 475 012220  052737  001567  001400            BIS     #XNUER2,RMER2I
 476 012226  020337  001400                    CMP     R3,RMER2I
 477 012232  001001                            BNE     60$
 478 012234  104015                            EMT     15
 479
 480                                    ;TEST REG SEL 8 FOR S-A-0
 481 012236                             60$:
 482 012236  004737  054674                    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 483 012242  010260  000014                    MOV     R2,RMER1(R0)    ;LOAD RMER1
 484 012246  010260  000006                    MOV     R2,RMDA(R0)     ;LOAD RMDA
 485 012252  010360  000034                    MOV     R3,RMDC(R0)     ;LOAD RMDC
 486 012256  010360  000042                    MOV     R3,RMER2(R0)    ;LOAD RMER2
 487 012262  016037  000014  001352            MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
 488 012270  016037  000006  001344            MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
 489 012276  020337  001352                    CMP     R3,RMER1I
 490 012302  001004                            BNE     70$
 491 012304  020337  001344                    CMP     R3,RMDAI
 492 012310  001001                            BNE     70$
 493 012312  104016                            EMT     16
 494
 495                                    ;TEST REG SEL 8 FOR S-A-1
 496 012314                             70$:
 497 012314  004737  054674                    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 498 012320  010260  000032                    MOV     R2,RMOF(R0)     ;LOAD RMOF
 499 012324  010260  000034                    MOV     R2,RMDC(R0)     ;LOAD RMDC
 500 012330  010260  000042                    MOV     R2,RMER2(R0)    ;LOAD RMER2
 501 012334  010360  000012                    MOV     R3,RMDS(R0)     ;LOAD RMDS
 502 012340  010360  000014                    MOV     R3,RMER1(R0)    ;LOAD RMER1
 503 012344  010360  000006                    MOV     R3,RMDA(R0)     ;LOAD RMDA
 504 012350  016037  000032  001370            MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
 505 012356  016037  000034  001372            MOV     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
 506 012364  016037  000042  001400            MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
 507 012372  052737  161577  001370            BIS     #XNUOF,RMOFI
 508 012400  001015                            BNE     80$
 509 012402  022737  176000  001372            CMP     #XNUDC,RMDCI
 510 012410  020337  001372                    CMP     R3,RMDCI
```

```
 012414  001007                           BNE     80$
 012416  052737  001567  001400           BIS     #XNUER2,RMER2I
 012424  020337  001400                   CMP     R3,RMER2I
 012430  001001                           BNE     80$
 012432  104017                           EMT     17
 012434                          80$:

                                 ;;*****************************************************************
                                 ;*TEST 7         DRIVE TYPE TEST

                                 ;;*****************************************************************
 012434                          TST7:
 012434  000004                           SCOPE                           ;SCOPE CALL
 012436  000240                           NOP
 012440  012706  001100                   MOV     #STACK,SP               ;LOAD THE STACK POINTER
 012444  013700  001276                   MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
 012450  013701  001466                   MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
 012454  012737  000007  001226           MOV     #7,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

 012462  016002  000026                   MOV     RMDT(R0),R2             ;STORE RMDT AT R2
 012466  022702  020024                   CMP     #SNGPRT,R2              ;SINGLE PORT RM03 ?
 012472  001431                           BEQ     10$                     ;YES !!
 012474  022702  024024                   CMP     #DULPRT,R2              ;DUAL PORT RM03 ?
 012500  001426                           BEQ     10$                     ;YES !!

 012502  022702  020025                   CMP     #SNGPRT!BIT0,R2 ;SINGLE PORT RM02 ?
 012506  001423                           BEQ     10$                     ;YES !!
 012510  022702  024025                   CMP     #DULPRT!BIT0,R2 ;DUAL PORT RM02 ?
 012514  001420                           BEQ     10$                     ;YES !!

 012516  022702  020027                   CMP     #SNGPRT!BIT1!BIT0,R2    ;SINGLE PORT RM05 ?
 012522  001415                           BEQ     10$                             ;YES !!
 012524  022702  024027                   CMP     #DULPRT!BIT1!BIT0,R2    ;DUAL PORT RM05 ?
 012530  001412                           BEQ     10$

 012532  010237  001142                   MOV     R2,$BDDAT               ;GET RECIEVED DRIVE TYPE
 012536  010037  001136                   MOV     R0,$BDADR               ;LOAD BAD ADDRESS
 012542  062737  000026  001136           ADD     #RMDT,$BDADR
 012550  104057                           EMT     57
 012552  000137  053564                   JMP     $EOSP                   ;GO TO NEXT DEVICE
 012556                          10$:

                                 ;;*****************************************************************
                                 ;*TEST 10        DEVICE AVAILABLE TEST

                                 ;;*****************************************************************
 012556                          TST10:
 012556  000004                           SCOPE                           ;SCOPE CALL
 012560  000240                           NOP
 012562  012706  001100                   MOV     #STACK,SP               ;LOAD THE STACK POINTER
 012566  013700  001276                   MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
 012572  013701  001466                   MOV     TSTQUE,R1               ;R1   POINTER TO DEVICE
 012576  012737  000010  001226           MOV     #10,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

 012604  004737  054674                   JSR     PC,CNTCLR               ;GO CLEAR CONTROLLER
 012610  016037  000000  001142           MOV     RMCS1(R0),$BDDAT                ;STORE RMCS1 AT $BDDAT
 012616  042737  173777  001142           BIC     #^CDVA,$BDDAT   ;CLEAR ALL BUT DVA
```

```
012624  001006                          BNE     10$             ;BRANCH IF DVA SET
012626  012737  004000  001140          MOV     #DVA,$GDDAT     ;SETUP EXPECTED
012634  010037  001136                  MOV     R0,$BDADR       ;SETUP REG ADDRESS
012640  104060                          EMT     60
012642                          10$:

                                ;;*************************************************************
                                ;*TEST 11       HOLDING REGISTER TRANSFER TEST

                                ;;*************************************************************
012642                          TST11:
012642  000004                          SCOPE                   ;SCOPE CALL
012644  000240                          NOP
012646  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
012652  013700  001276                  MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
012656  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
012662  012737  000011  001226          MOV     #11,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

012670  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
012674  005003                          CLR     R3              ;CLEAR ERROR FLAGS
012676  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
012702  062737  000036  001136          ADD     #RMHR,$BDADR

                                ;WRITE ONES THEN ZEROS IN RMHR AND CHECK FOR S-A-1 BITS.
                                ;NOTE THAT IT IS NECESSARY TO WRITE SOME OTHER REGISTER IN
                                ;ORDER TO WRITE THE HOLDING REGISTER, AND RMDA IS USED FOR THIS
                                ;PURPOSE.
012710  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
012716  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
012724  016037  000036  001142          MOV     RMHR(R0),$BDDAT ;STORE RMHR AT $BDDAT
012732  005137  001142                  COM     $BDDAT          ;ANY ERROR??
012736  001405                          BEQ     10$             ;NO!!
012740  005037  001140                  CLR     $GDDAT          ;LOAD EXPECTED
012744  104061                          EMT     61
012746  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAGS

                                ;WRITE ZEROS THEN ONES IN RMHR AND CHECK FOR S-A-0 BITS.
012752                          10$:
012752  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
012760  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
012766  016037  000036  001142          MOV     RMHR(R0),$BDDAT ;STORE RMHR AT $BDDAT
012774  005137  001142                  COM     $BDDAT          ;RMHR IS COMPLEMENTED WHEN READ
013000  012737  177777  001140          MOV     #-1,$GDDAT      ;SETUP EXPECTED
013006  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;ANY ERROR??
013014  001403                          BEQ     20$             ;NO!!
013016  104062                          EMT     62
013020  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG

                                ;IF NO PREVIOUS ERRORS, WRITE AND READ SHIFTING ONE BIT PATTERN.
013024                          20$:
013024  005703                          TST     R3              ;ANY FLAGS SET??
013026  001025                          BNE     50$             ;YES!!
013030  012702  000001                  MOV     #1,R2           ;R2=DATA PATTERN
013034                          30$:
013034  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
013042  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
013046  016037  000036  001142          MOV     RMHR(R0),$BDDAT ;STORE RMHR AT $BDDAT
```

```
 598 013054  005137  001142                COM     $BDDAT          ;RMHR IS COMPLEMENTED
 599 013060  023702  001142                CMP     $BDDAT,R2       ;ANY ERROR??
 600 013064  001404                        BEQ     40$             ;NO!.
 601 013066  010237  001140                MOV     R2,$GDDAT       ;SETUP EXPECTED
 602 013072  104063                        EMT     63
 603 013074  000402                        BR      50$             ;DO NOT COLLECT ALL ERRORS
 604
 605 013076  006302                40$:    ASL     R2              ;SHIFT TO NEXT PATTERN
 606 013100  001355                        BNE     30$             ;CONTINUE IF NOT DONE
 607
 608 013102                        50$:
 609
 610                               ;;*****************************************************************
                                   ;*TEST 12        CONTROL STATUS #1 TRANSFER TEST

                                   ;;*****************************************************************
     013102                        TST12:
     013102  000004                        SCOPE                   ;SCOPE CALL
     013104  000240                        NOP
     013106  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     013112  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     013116  013701  001466                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     013122  012737  000012  001226        MOV     #12,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 611
 612 013130  005003                        CLR     R3              ;R3 = ERROR INDICATOR
 613 013132  004737  054674                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 614
 615                               ;WRITE ONES IN RMCS1, BITS 01-05, THEN CLEAR. READ AND
 616                               ;CHECK FOR S-A-1 BITS.
 617 013136  012760  000076  000000        MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
 618 013144  004737  054674                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 619 013150  016037  000000  001142        MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 620 013156  042737  177701  001142        BIC     #^CILF76,$BDDAT
 621 013164  001410                        BEQ     5$
 622 013166  005037  001140                CLR     $GDDAT
 623 013172  010037  001136                MOV     R0,$BDADR
 624 013176  062737  000000  001136        ADD     #RMCS1,$BDADR
 625 013204  104043                        EMT     43
 626
 627                               ;WRITE ONES IN RMCS1, BITS 01-05, THEN WRITE ZEROS. READ AND CHECK FOR
 628                               ;S-A-1 BITS.
 629 013206                        5$:
 630 013206  012760  000076  000000        MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
 631 013214  012760  000000  000000        MOV     #0,RMCS1(R0)    ;LOAD RMCS1
 632 013222  016037  000000  001142        MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 633 013230  042737  177701  001142        BIC     #^CILF76,$BDDAT
 634 013236  001412                        BEQ     10$
 635 013240  005037  001140                CLR     $GDDAT
 636 013244  010037  001136                MOV     R0,$BDADR
 637 013250  062737  000000  001136        ADD     #RMCS1,$BDADR
 638 013256  104023                        EMT     23
 639 013260  052703  000001                BIS     #BIT0,R3        ;SET ERROR FLAG
 640
 641                               ;WRITE ZEROS IN RMCS1, THEN ONES, READ AND CHECK S-A-0 BITS.
 642 013264                        10$:
 643 013264  012760  000000  000000        MOV     #0,RMCS1(R0)    ;LOAD RMCS1
 644 013272  012760  000076  000000        MOV     #ILF76,RMCS1(R0)        ;LOAD RMCS1
```

```
 645 013300  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 646 013306  042737  177701  001142          BIC     #^CILF76,$BDDAT
 647 013314  012737  000076  001140          MOV     #ILF76,$GDDAT
 648 013322  023737  001140  001142          CMP     $GDDAT,$BDDAT
 649 013330  001410                          BEQ     20$
 650 013332  010037  001136                  MOV     R0,$BDADR
 651 013336  062737  000000  001136          ADD     #RMCS1,$BDADR
 652 013344  104024                          EMT     24
 653 013346  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
 654
 655                                  ;WRITE A SHIFTING ONE BIT PATTERN IN RMCS1, READ AND CHECK FOR STUCK BITS.
 656 013352                           20$:
 657 013352  005703                          TST     R3              ;OMIT IF ANY ERRORS
 658 013354  001035                          BNE     50$
 659 013356  012702  000002                  MOV     #2,R2           ;R2 - TEST PATTERN
 660 013362                           30$:
 661 013362  010203                          MOV     R2,R3           ;R3 = EXPECTED RESULT, BITS 1-5
 662 013364  042703  177701                  BIC     #^CILF76,R3
 663 013370  012760  000000  000000          MOV     #0,RMCS1(R0)    ;LOAD RMCS1
 664 013376  010260  000000                  MOV     R2,RMCS1(R0)    ;LOAD RMCS1
 665 013402  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 666 013410  042737  177701  001142          BIC     #^CILF76,$BDDAT
 667 013416  020337  001142                  CMP     R3,$BDDAT
 668 013422  001410                          BEQ     40$
 669 013424  010337  001140                  MOV     R3,$GDDAT
 670 013430  010037  001136                  MOV     R0,$BDADR
 671 013434  062737  000000  001136          ADD     #RMCS1,$BDADR
 672 013442  104025                          EMT     25
 673 013444  006302                   40$:   ASL     R2              ;SHIFT TO NEXT BIT
 674 013446  001345                          BNE     30$             ;CONTINUE IF R2 NOT ZERO
 675 013450                           50$:
 676
 677                                  ;;*************************************  *************************
                                      ;*TEST 13        ERROR REGISTER #1 TRANSFER TEST

                                      ;;*************************************************************
     013450                           TST13:
     013450  000004                          SCOPE                   ;SCOPE CALL
     013452  000240                          NOP
     013454  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     013460  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     013464  013701  001466                  MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
     013470  012737  000013  001226          MOV     #13,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 678
 679 013476  005003                          CLR     R3              ;CLEAR ERROR FLAG
 680
 681                                  ;WRITE ONES IN RMER1, CLEAR AND CHECK FOR S-A-1 BITS
 682 013500  012760  177777  000014          MOV     #-1,RMER1(R0)   ;LOAD RMER1
 683
 684 013506  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 685 013512  016037  000014  001352          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
 686 013520  013737  001352  001142          MOV     RMER1I,$BDDAT
 687 013526  042737  177760  001142          BIC     #^C<PAR.RMR!ILF.ILR>,$BDDAT
 688 013534  001410                          BEQ     10$
 689 013536  005037  001140                  CLR     $GDDAT
 690 013542  010037  001136                  MOV     R0,$BDADR
 691 013546  062737  000014  001136          ADD     #RMER1,$BDADR
```

```
692 013554  104027                          EMT     27
693 013556                          10$:
694 013556  013737  001352  001142          MOV     RMER1I,$BDDAT
695 013564  042737  074017  001142          BIC     #^C<DCK!IAE.AOE!HCRC.HCE!ECH!WCF!FER>,$BDDAT
696 013572  001410                          BEQ     20$
697 013574  005037  001140                  CLR     $GDDAT
698 013600  010037  001136                  MOV     R0,$BDADR
699 013604  062737  000014  001136          ADD     #RMER1,$BDADR
700 013612  104030                          EMT     30
701 013614                          20$:
702 013614  013737  001352  001142          MOV     RMER1I,$BDDAT
703 013622  042737  147777  001142          BIC     #^C<OPI!DTE>,$BDDAT
704 013630  001410                          BEQ     30$
705 013632  005037  001140                  CLR     $GDDAT
706 013636  010037  001136                  MOV     R0,$BDADR
707 013642  062737  000014  001136          ADD     #RMER1,$BDADR
708 013650  104031                          EMT     31
709
710                                 ;WRITE ONES THEN ZEROS IN RMER1, READ AND CHECK FOR S-A-1 BITS
711 013652                          30$:
712 013652  012760  177777  000014          MOV     #-1,RMER1(R0)    ;LOAD RMER1
713 013660  012760  000000  000014          MOV     #0,RMER1(R0)     ;LOAD RMER1
714 013666  016037  000014  001352          MOV     RMER1(R0),RMER1I        ;STORE RMER1 IN INPUT BUFFER
715 013674  013737  001352  001142          MOV     RMER1I,$BDDAT
716 013702  042737  177770  001142          BIC     #^C<RMR!ILF!ILR>,$BDDAT
717 013710  001412                          BEQ     40$
718 013712  005037  001140                  CLR     $GDDAT
719 013716  010037  001136                  MOV     R0,$BDADR
720 013722  062737  000014  001136          ADD     #RMER1,$BDADR
721 013730  104032                          EMT     32
722 013732  052703  000001                  BIS     #BIT0,R3         ;SET ERROR FLAG
723 013736  013737  001352  001142  40$:    MOV     RMER1I,$BDDAT
724 013744  042737  074017  001142          BIC     #^C<DCK!IAE!AOE.HCRC!HCE!ECH!WCF!FER>,$BDDAT
725 013752  001412                          BEQ     50$
726 013754  005037  001140                  CLR     $GDDAT
727 013760  010037  001136                  MOV     R0,$BDADR
728 013764  062737  000014  001136          ADD     #RMER1,$BDADR
729 013772  104033                          EMT     33
730 013774  052703  000001                  BIS     #BIT0,R3         ;SET ERROR FLAG
731 014000                          50$:
732 014000  013737  001352  001142          MOV     RMER1I,$BDDAT
733 014006  042737  147777  001142          BIC     #^C<OPI!DTE>,$BDDAT
734 014014  001412                          BEQ     60$
735 014016  005037  001140                  CLR     $GDDAT
736 014022  010037  001136                  MOV     R0,$BDADR
737 014026  062737  000014  001136          ADD     #RMER1,$BDADR
738 014034  104034                          EMT     34
739 014036  052703  030000                  BIS     #BIT,R3
740
741                                 ;WRITE ZEROS THEN ONES IN RMER1,READ AND CHECK FOR S-A-0 BITS
742 014042                          60$:
743 014042  012760  000000  000014          MOV     #0,RMER1(R0)     ;LOAD RMER1
744 014050  012760  177777  000014          MOV     #-1,RMER1(R0)    ;LOAD RMER1
745 014056  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
746 014064  012737  177777  001140          MOV     #-1,$GDDAT
747 014072  023737  001140  001142          CMP     $GDDAT,$BDDAT
748 014100  001410                          BEQ     70$
```

```
 749 014102  010037  001136                MOV     R0,$BDADR
 750 014106  062737  000014  001136        ADD     #RMER1,$BDADR
 751 014114  104035                        EMT     35
 752 014116  052703  000002                BIS     #BIT1,R3
 753
 754                                ;WRITE A SHIFTING 1 BIT IN RMER1 AND CHECK FOR STUCK BITS
 755                                ;NOTE:  DONT TEST UNSAFE OR PARITY
 756 014122                         70$:
 757 014122  005703                        TST     R3              ;SKIP THIS PART IF ANY ERRORS
 758 014124  001042                        BNE     120$
 759 014126  012702  000001                MOV     #1,R2           ;R2 = TEST PATTERN
 760 014132                         80$:
 761 014132  004737  054674                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 762 014136  010260  000014                MOV     R2,RMER1(R0)    ;LOAD RMER1
 763 014142  016037  000014  001142        MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
 764 014150  032702  000010                BIT     #PAR,R2         ;DONT TEST PAR - 0
 765 014154  001003                        BNE     90$
 766 014156  042737  000010  001142        BIC     #PAR,$BDDAT
 767 014164  032702  040000         90$:   BIT     #UNS,R2         ;DONT TEST UNS   0
 768 014170  001003                        BNE     100$
 769 014172  042737  040000  001142        BIC     #UNS,$BDDAT
 770 014200  020237  001142         100$:  CMP     R2,$BDDAT
 771 014204  001410                        BEQ     110$
 772 014206  010237  001140                MOV     R2,$GDDAT
 773 014212  010037  001136                MOV     R0,$BDADR
 774 014216  062737  000014  001136        ADD     #RMER1,$BDADR
 775 014224  104036                        EMT     36
 776
 777 014226  006302                 110$:  ASL     R2              ;SHIFT TO NEXT BIT
 778 014230  001340                        BNE     80$             ;CONTINUE IF R2 NOT ZERO
 779 014232                         120$:
 780
 781                                ;;*****************************************************************
                                    ;*TEST 14        CLEAR OFFSET STUCK ACTIVE TEST

                                    ;;*****************************************************************
     014232                         TST14:
     014232  000004                        SCOPE                   ;SCOPE CALL
     014234  000240                        NOP
     014236  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     014242  013700  001276                MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
     014246  013701  001466                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     014252  012737  000014  001226        MOV     #14,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 782
 783 014260  004737  054674                JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 784 014264  012760  177777  000032        MOV     #-1,RMOF(R0)    ;LOAD RMOF
 785 014272  016037  000032  001142        MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
 786 014300  042737  177577  001142        BIC     #^COFD,$BDDAT
 787 014306  001011                        BNE     10$             ;BRANCH IF OFD IS A ONE
 788 014310  012737  000200  001140        MOV     #OFD,$GDDAT     ;SETUP ERROR MESSAGE
 789 014316  010037  001136                MOV     R0,$BDADR
 790 014322  062737  000032  001136        ADD     #RMOF,$BDADR
 791 014330  104172                        EMT     172
 792 014332                         10$:                           ;END OF TEST
 793
 794                                ;;*****************************************************************
                                    ;*TEST 15        OFFSET REGISTER TRANSFER TEST
```

```
                                        ;;*****************************************************************
014332                                  TST15:                                                  ;I
014332  000004                                  SCOPE                   ;SCOPE CALL
014334  000240                                  NOP
014336  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
014342  013700  001276                          MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
014346  013701  001466                          MOV     TSTQUE,R1       ;R1   POINTER TO DEVICE
014352  012737  000015  001226                  MOV     #15,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
795
796 014360  005003                              CLR     R3              ;RESET ERROR FLAGS
797 014362  010037  001136                      MOV     R0,$BDADR       ;SETUP BAD ADDRESS
798 014366  062737  000032  001136              ADD     #RMOF,$BDADR
799 014374  005037  001140                      CLR     $GDDAT          ;SETUP EXPECTED DATA
800
801                                     ;WRITE ONES THEN ZEROS IN RMOF AND CHECK FOR S-A-1 BITS.
802 014400  012760  177777  000032              MOV     #-1,RMOF(R0)    ;LOAD RMOF
803 014406  012760  000000  000032              MOV     #0,RMOF(R0)     ;LOAD RMOF
804 014414  016037  000032  001370              MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
805
806                                     ;CHECK UNUSED BITS OF RMOF FOR ZERO
807 014422  013737  001370  001142              MOV     RMOFI,$BDDAT    ;GET UNUSED BITS
808 014430  042737  016200  001142              BIC     #^CXNUOF,$BDDAT
809 014436  001403                              BEQ     10$             ;BRANCH IF NO ERROR
810 014440  104053                              EMT     53
811 014442  052703  000001                      BIS     #BIT0,R3        ;SET ERROR FLAG
812
813                                     ;CHECK USED BITS OF RMOF FOR ZERO
814 014446                              10$:
815 014446  013737  001370  001142              MOV     RMOFI,$BDDAT    ;GET USED BITS
816 014454  042737  161577  001142              BIC     #XNUOF,$BDDAT
817 014462  001403                              BEQ     20$             ;BRANCH IF NO ERROR
818 014464  104054                              EMT     54
819 014466  052703  000001                      BIS     #BIT0,R3        ;SET ERROR FLAG
820
821                                     ;WRITE ZEROS THEN ONES ON RMOF AND CHECK FOR S-A-0 BITS.
822 014472                              20$:
823 014472  012760  000000  000032              MOV     #0,RMOF(R0)     ;LOAD RMOF
824 014500  012760  177777  000032              MOV     #-1,RMOF(R0)    ;LOAD RMOF
825 014506  016037  000032  001370              MOV     RMOF(R0),RMOFI  ;STORE RMOF IN INPUT BUFFER
826
827                                     ;CHECK UNUSED BITS OF RMOF FOR ZERO.
828 014514  013737  001370  001142              MOV     RMOFI,$BDDAT    ;GET UNUSED BITS
829 014522  042737  016200  001142              BIC     #^CXNUOF,$BDDAT
830 014530  001403                              BEQ     30$             ;BRANCH IF NO ERROR
831 014532  104053                              EMT     53
832 014534  052703  000001                      BIS     #BIT0,R3
833
834                                     ;CHECK USED BITS OF RMOF FOR ONE.
835 014540                              30$:
836 014540  013737  001370  001142              MOV     RMOFI,$BDDAT    ;GET USED BITS
837 014546  042737  161577  001142              BIC     #XNUOF,$BDDAT
838 014554  012737  016200  001140              MOV     #^CXNUOF,$GDDAT ;SETUP EXPECTED STATUS
839 014562  023737  001140  001142              CMP     $GDDAT,$BDDAT
840 014570  001403                              BEQ     40$             ;BRANCH IF NO ERROR
841 014572  104055                              EMT     55
842 014574  052703  000002                      BIS     #BIT1,R3
```

```
                                        ;IF NO PREVIOUS ERRORS, TEST RMOF WITH SHIFTING ONE BIT PATTERN.
014600                                  40$:
014600  005703                          TST     R3              ;ANY ERROR??
014602  001025                          BNE     70$             ;YES!!
014604  012702  000001                  MOV     #1,R2           ;STARTING DATA PATTERN
014610                                  50$:
014610  012760  000000  000032          MOV     #0,RMOF(R0)     ;LOAD RMOF
014616  010260  000032                  MOV     R2,RMOF(R0)     ;LOAD RMOF
014622  016037  000032  001142          MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
014630  010203                          MOV     R2,R3           ;SETUP EXPECTED RESULT
014632  042703  161577                  BIC     #XNUOF,R3       ;CLEAR UNUSED BITS
014636  020337  001142                  CMP     R3,$BDDAT       ;COMPARE EXPECTED & RECEIVED
014642  001403                          BEQ     60$             ;BRANCH IF NO ERROR
014644  010337  001140                  MOV     R3,$GDDAT       ;LOAD EXPECTED
014650  104056                          EMT     56
014652  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
014654  001355                          BNE     50$             ;CONTINUE IF NOT DONE

014656                                  70$:

                                        ;;*****************************************************************
                                        ;*TEST 16       ERROR REGISTER #2 TRANSFER TEST

                                        ;;*****************************************************************
014656                                  TST16:
014656  000004                          SCOPE                   ;SCOPE CALL
014660  000240                          NOP
014662  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
014666  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
014672  013701  001466                  MOV     TSTQUE,R1       ;R1   POINTER TO DEVICE
014676  012737  000016  001226          MOV     #16,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

014704  005003                          CLR     R3              ;RESET ERROR FLAGS
014706  010037  001136                  MOV     R0,$BDADR       ;SETUP BAD ADDRESS
014712  062737  000042  001136          ADD     #RMER2,$BDADR
014720  005037  001140                  CLR     $GDDAT          ;SETUP EXPECTED DATA
014724  012737  010000  001444          MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT MODE TO ALLOW
                                                                ;''SSE'' TO BE SET IN RMER2

                                        ;WRITE ONES IN RMER2, CLEAR AND CHECK FOR S-A-1 BITS
014732  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2

014740  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
014744  013760  001444  000032          MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
014752  016037  000042  001400          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER

                                        ;TEST UNUSED BITS FOR ZERO-FAILURE ON IF
014760  013737  001400  001142          MOV     RMER2I,$BDDAT
014766  042737  176210  001142          BIC     #^CXNUER2,$BDDAT
014774  001403                          BEQ     10$             ;BRANCH IF NO ERROR
014776  104044                          EMT     44
015000  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG

                                        ;TEST ''DPE'', ''IVC'', ''LSC'' FOR ZERO-FAILURE ON CS, IF
015004                                  10$:
015004  013737  001400  001142          MOV     RMER2I,$BDDAT
```

```
890 015012  042737  143777  001142          BIC     #^C<OPE!IVC!LSC>,$BDDAT
891 015020  001403                          BEQ     20$             ;BRANCH IF NO ERROR
892 015022  104045                          EMT     45
893 015024  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
894
895                                 ;TEST 'LBC', 'DPE' FOR ZERO-FAILURE ON DS, IF
896 015030                          20$:
897 015030  013737  001400  001142          MOV     RMER2I,$BDDAT
898 015036  042737  175767  001142          BIC     #^C<LBC!DPE>,$BDDAT
899 015044  001403                          BEQ     30$             ;BRANCH IF NO ERROR
900 015046  104046                          EMT     46
901 015050  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
902
903                                 ;WRITE ONES IN RMER2 THEN WRITE ZEROS AND CHECK FOR S-A-1 BITS.
904 015054                          30$:
905 015054  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
906 015062  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
907 015070  016037  000042  001400          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
908
909                                 ;TEST 'DPE', 'IVC', 'LSC' FOR ZERO-FAILURE ON CS, IF
910 015076  013737  001400  001142          MOV     RMER2I,$BDDAT
911 015104  042737  143777  001142          BIC     #^C<OPE!IVC!LSC>,$BDDAT
912 015112  001403                          BEQ     40$             ;BRANCH IF NO ERROR
913 015114  104047                          EMT     47
914 015116  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
915
916                                 ;TEST 'LBC', 'DPE' FOR ZERO-FAILURE ON DS, IF
917 015122                          40$:
918 015122  013737  001400  001142          MOV     RMER2I,$BDDAT
919 015130  042737  175767  001142          BIC     #^C<LBC!DPE>,$BDDAT
920 015136  001403                          BEQ     50$             ;BRANCH IF NO ERROR
92  015140  104050                          EMT     50
922 015142  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
923
924                                 ;WRITE ZEROS IN RMER2 THEN WRITE ONES AND CHECK FOR S-A-0 BITS.
925 015146                          50$:
926 015146  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
927 015154  012760  177777  000042          MOV     #-1,RMER2(R0)   ;LOAD RMER2
928 015162  016037  000042  001400          MOV     RMER2(R0),RMER2I        ;STORE RMER2 IN INPUT BUFFER
929                                 ;TEST UNUSED BITS FOR ZERO-FAILURE ON IF
930 015170  013737  001400  001142          MOV     RMER2I,$BDDAT
931 015176  042737  176210  001142          BIC     #^CXNUER2,$BDDAT
932 015204  001403                          BEQ     60$             ;BRANCH IF NO ERROR
933 015206  104044                          EMT     44
934 015210  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
935
936                                 ;TEST USED BITS FOR ONE-FAILURE ON IF
937 015214                          60$:
938 015214  013737  001400  001142          MOV     RMER2I,$BDDAT
939 015222  042737  001567  001142          BIC     #XNUER2,$BDDAT
940 015230  012737  176210  001140          MOV     #^CXNUER2,$GDDAT
941 015236  023737  001140  001142          CMP     $GDDAT,$BDDAT
942 015244  001403                          BEQ     70$             ;BRANCH IF NO ERROR
943 015246  104051                          EMT     51
944 015250  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
945
946                                 ;IF NO PREVIOUS ERROR, TEST BIT INTERFERENCE WITH SHIFTING ONE BIT.
```

```
947 015254                                  70$:
948 015254  005703                             TST     R3              ;ANY ERRORS?
949 015256  001044                             BNE     120$            ;YES.!
950 015260  012702  000001                     MOV     #1,R2           ;R2=DATA PATTERN
951 015264                                  80$:
952 015264  004737  054674                     JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
953 015270  013760  001444  000032             MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
954 015276  010260  000042                     MOV     R2,RMER2(R0)    ;LOAD RMER2
955 015302  016037  000042  001142             MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
956 015310  032702  040000                     BIT     #SKI,R2         ;IS SKI BEING SET?
957 015314  001003                             BNE     90$             ;YES!!
958 015316  042737  040000  001142             BIC     #SKI,$BDDAT     ;DONT TEST SKI FOR ZERO
959 015324  032702  000200                 90$:BIT     #DVC,R2         ;IS DVC BEING SET??
960 015330  001003                             BNE      00$            ;YES!!
961 015332  042737  000200  001142             BIC     #DVC,$BDDAT     ;DONT TEST DVC FOR ZERO
962 015340  010237  001140                 100$:MOV    R2,*GDDAT
963 015344  042737  001567  001140             BIC     #XNUER2,$GDDAT  ;UNUSED BITS SHOULD BE ZERO
964 015352  023737  001140  001142             CMP     $GDDAT,$BDDAT   ;ANY ERRORS??
965 015360  001401                             BEQ     110$            ;NO!!
966 015362  104052                             EMT     52
967
968 015364  006302                 110$:       ASL     R2              ;SHIFT TO NEXT DATA BIT
969 015366  001336                             BNE     80$             ;CONTINUE IF NOT DONE
970
971 015370                                  120$:
972
973                                         ;;*****************************************************************
                                            ;*TEST 17        SERIAL NUMBER TEST
                                            ;;*****************************************************************
    015370                                  TST17:
    015370  000004                             SCOPE                   ;SCOPE CALL
    015372  000240                             NOP
    015374  012706  001100                     MOV     #STACK,SP       ;LOAD THE STACK POINTER
    015400  013700  001276                     MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
    015404  013701  001466                     MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
    015410  012737  000017  001226             MOV     #17,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
974
975 015416  004737  054674                     JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
976 015422  010037  001136                     MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS FOR TYPEOUT
977 015426  062737  000030  001136             ADD     #RMSN,$BDADR
978 015434  012702  000031                     MOV     #25.,R2         ;READ RMSN 25 TIMES
979
980                                         ;READ RMSN AND USE THE RESULT AS EXPECTED VALUE
981 015440  016037  000030  001140             MOV     RMSN(R0),$GDDAT ;STORE RMSN AT $GDDAT
982
983                                         ;READ RMSN AND COMPARE WITH INITIAL VALUE
984 015446                                  10$:
985 015446  016037  000030  001142             MOV     RMSN(R0),$BDDAT ;STORE RMSN AT $BDDAT
986 015454  023737  001140  001142             CMP     $GDDAT,$BDDAT
987 015462  001401                             BEQ     20$             ;BRANCH IF SERIAL NUMBER CONSISTENT
988 015464  104136                             EMT     136
989
990                                         ;DECREMENT COUNT AND CONTINUE IF NOT DONE
991 015466                                  20$:
992 015466  005302                             DEC     R2
993 015470  100366                             BPL     10$
```

```
 015472                          30$:                                        ;END OF TEST

                                 ;;*************************************************************
                                 ;*TEST 20          CONTROL BUS PARITY DETECTION TEST

                                 ;;*************************************************************
 015472                          TST20:
 015472  000004                          SCOPE                               ;SCOPE CALL
 015474  000240                          NOP
 015476  012706  001100                  MOV     #STACK,SP                   ;LOAD THE STACK POINTER
 015502  013700  001276                  MOV     $BASE,R0                    ;R0 = UNIBUS ADDRESS
 015506  013701  001466                  MOV     TSTQUE,R1                   ;R1 = POINTER TO DEVICE
 015512  012737  000020  001226          MOV     #20,$TESTN                  ;;SET TEST NUMBER IN APT MAIL BOX

                                 ;SETUP FOR FIRST TEST LOOP (NO ERROR)
 015520  005037  001140                  CLR     $GDDAT                      ;'PAR'' SHOULD BE ZERO
 015524  111137  001422                  MOVB    (R1),RMCS20                 ;SETUP RMCS2 VALUE
 015530  042737  177770  001422          BIC     #^CUNTMSK,RMCS20
 015536  012737  000001  001450          MOV     #1,RMHR0                    ;INITIALIZE DATA PATTERN
 015544  000402                          BR      6$                          ;SKIP INCREMENT FIRST TIME
 015546  006337  001450          5$:     ASL     RMHR0                       ;SHIFT TO NEXT PATTERN

                                 ;CLEAR AND VERIFY THAT 'PAR'' IS RESET
 015552                          6$:
 015552  004737  054674                  JSR     PC,CNTCLR                   ;GO CLEAR CONTROLLER
 015556  016037  000014  001142          MOV     RMER1(R0),$BDDAT                    ;STORE RMER1 AT $BDDAT
 015564  016037  000042  001400          MOV     RMER2(R0),RMER2I                    ;STORE RMER2 IN INPUT BUFFER
 015572  042737  177767  001142          BIC     #^CPAR,$BDDAT               ;DID 'PAR'' RESET?
 015600  001415                          BEQ     20$                         ;YES!.
 015602  010037  001136                  MOV     R0,$BDADR                   ;SETUP REGISTER ADDRESS
 015606  062737  000014  001136          ADD     #RMER1,$BDADR
 015614  032737  000010  001400          BIT     #DPE,RMER2I                 ;IS 'DPE'' SET??
 015622  001002                          BNE     10$                         ;YES!.
 015624  104067                          EMT     67
 015626  000453                          BR      50$
 015630                          10$:
 015630  104070                          EMT     70
 015632  000451                          BR      50$

                                 ;WRITE TEST PATTERN AND VERIFY 'PAR'' STATUS
 015634                          20$:
 015634  013760  001422  000010          MOV     RMCS20,RMCS2(R0)                    ;LOAD RMCS2
 015642  013760  001450  000036          MOV     RMHR0,RMHR(R0)      ;LOAD RMHR
 015650  016037  000014  001142          MOV     RMER1(R0),$BDDAT                    ;STORE RMER1 AT $BDDAT
 015656  042737  177767  001142          BIC     #^CPAR,$BDDAT
 015664  023737  001140  001142          CMP     $GDDAT,$BDDAT               ;IS 'PAR'' CORRECT??
 015672  001410                          BEQ     40$                         ;YES!!
 015674  032737  000020  001422          BIT     #PAT,RMCS20                 ;SHOULD 'PAR'' BE SET?
 015702  001002                          BNE     30$                         ;YES!!
 015704  104071                          EMT     71
 015706  000423                          BR      50$
 015710                          30$:
 015710  104072                          EMT     72
 015712  000421                          BR      50$                         ;SKIP TO NEXT
                                 ;GO TO NEXT PATTERN
 015714  005737  001450          40$:    TST     RMHR0                       ;IS DATA PATTERN COMPLETE??
 015720  001312                          BNE     5$                          ;NO!!
```

```
1039 015722  032737  000020  001422          BIT     #PAT,RMCS20     ;IS TEST COMPLETE??
1040 015730  001012                          BNE     50$             ;YES!!
1041                                ;SETUP FOR SECOND TEST LOOP (ERROR)
1042 015732  052737  000020  001422          BIS     #PAT,RMCS20     ;TURN ON BAD PARITY
1043 015740  012737  000010  001140          MOV     #PAR,$GDDAT     ;EXPECT ERROR
1044 015746  012737  000001  001450          MOV     #1,RMHRO        ;INITIALIZE DATA PATTERN
1045 015754  000676                          BR      6$              ;START LOOP
1046
1047 015756                                 50$:                    ;END OF TEST
1048
1049                                ;;*****************************************************************
                                    ;*TEST 21        CONTROL BUS PARITY GENERATION TEST

                                    ;;*****************************************************************
     015756                         TST21:
     015756  000004                         SCOPE                   ;SCOPE CALL
     015760  000240                         NOP
     015762  012706  001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
     015766  013700  001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     015772  013701  001466                 MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
     015776  012737  000021  001226         MOV     #21,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1050
1051                                ;SETUP FOR TEST (NO ERROR)
1052 016004  012737  000001  001450          MOV     #1,RMHRO        ;INITIALIZE DATA PATTERN
1053 016012  005037  001140                  CLR     $GDDAT          ;MCPE SHOULD BE ZERO
1054 016016  000402                          BR      20$             ;DONT SHIFT FIRST TIME
1055
1056                                ;SHIFT DATA PATTERN
1057 016020                                 10$:
1058 016020  006337  001450                  ASL     RMHRO
1059 016024                                 20$:
1060 016024  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1061 016030  013760  001450  000036          MOV     RMHRO,RMHR(R0)  ;LOAD RMHR
1062
1063                                ;TRANSFER DATA TO RH, VERIFY NO 'MCPE'' ERROR
1064 016036  016037  000036  001374          MOV     RMHR(R0),RMHRI  ;STORE RMHR IN INPUT BUFFER
1065 016044  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
1066 016052  042737  157777  001142          BIC     #^CMCPE,$BDDAT  ;WAS BAD PARITY DETECTED??
1067 016060  001402                          BEQ     30$             ;NO!.
1068 016062  104073                          EMT     73
1069 016064  000403                          BR      40$
1070
1071                                ;GO TO NEXT PATTERN
1072 016066                                 30$:
1073 016066  005737  001450                  TST     RMHRO           ;DONE ALL PATTERNS??
1074 016072  001352                          BNE     10$             ;NO!.
1075 016074                                 40$:                    ;END OF TEST
1076
1077                                ;;*****************************************************************
                                    ;*TEST 22        RMDA,RMDC FAULT TEST

                                    ;;*****************************************************************
     016074                         TST22:
     016074  000004                         SCOPE                   ;SCOPE CALL
     016076  000240                         NOP
     016100  012706  001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
     016104  013700  001276                 MOV     $BASE,R0        ;R0   UNIBUS ADDRESS
```

```
ZRMPB0 RMOS/3/2 DSKLS TST *    MACRO ..... --APR-8* *:*4:25 PAGE *3-22
T22    RMDA,RMDC FAUL* TEST
```

```
016110  013701  00*466          MO.     *S*QUE,R1       ;R1 = POINTER TO DEVICE
016114  012737  C00C22  00*22e   MO.     #22,$*EST*      ;;SET TEST NUMBER IN APT MAIL BOX
1078
1079 016122  004737  0546*4      JSR     PC,CN*CLR       ;GO CLEAR CONTROLLER
1080
1081                             ;WR*TE ZEROS, *HEN ONES IN RMDA,RMDC-READ AND *EST FOR S-A-0
1082 016126  012760  000000  000006   MO.     #0,RMDA(R0)     ;LOAD RMDA
1083 016134  012760  000000  000034   MO.     #0,RMDC(R0)     ;LOAD RMDC
1084 016142  012760  177777  000006   MO.     #-1,RMDA(R0)    ;LOAD RMDA
1085 016150  012760  177777  000034   MO.     #-1,RMDC(R0)    ;LOAD RMDC
1086 016156  016037  000006  001344   MOV     RMDA(R0),RMDAI  ;STORE RMDA IN INPUT BUFFER
1087 016164  016037  000034  001372   MO.     RMDC(R0),RMDCI  ;STORE RMDC IN INPUT BUFFER
1088 016172  022737  177777  001344   CMP     #-1,RMDAI       ;IS ANY REGISTER ALL ONES??
1089 016200  001*0*                   BEQ     *0$             ;YES.!
1090 016202  052737  176000  00*372   BIS     #RMLDC,RMDCI    ;SET UNUSED BITS
1091 016210  022737  177777  00*372   CMP     #-1,RMDCI
1092 016216  001401                   BEQ     *0$             ;YES.
1093 016220  104042                   EM*     42
1094
1095 016222                      *0$:
1096
1097                             ;*****************************************************************
                                 ;*TES* 23        DISK ADDRESS *RANSFER TEST
                                 ;*****************************************************************
016222                           *0*23:
016222  000004                   SCOPE                   ;SCOPE CALL
016224  000240                   NOP
016226  012706  001*0C          MO.     #S*ACK,SP       ;LOAD *HE STACK POINTER
016232  013700  00*276          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
016236  013701  001466          MO.     *S*QUE,R1       ;R1 = POINTER TO DEVICE
016242  012737  000023  00*22e   MO.     #23,$*EST*      ;;SET TEST NUMBER IN APT MAIL BOX
1098
1099 016250  005003              CLR     R3              ;R3 = ERROR INDICATOR
1100 016252  004737  054674      JSR     PC,CN*CLR       ;GO CLEAR CONTROLLER
1101
1102                             ;WR*TE ONES IN RMDA, *HEN WRITE ZEROS, READ BACK AND CHECK FOR S-A-1 BITS
1103 016256  012760  177777  000006   MOV     #-1,RMDA(R0)    ;LOAD RMDA
1104 016264  012760  000000  000006   MO.     #0,RMDA(R0)     ;LOAD RMDA
1105 016272  016037  000006  00*142   MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
1106 016300  005737  001142          *ST     $BDDAT
1107 016304  001412                  BEQ     10$
1108 016306  005037  001140          CLR     $GDDAT
1109 016312  010037  001*36          MOV     R0,$BDADR
1110 016316  062737  000006  001*36   ADD     #RMDA,$BDADR
1111 016324  104020                  EMT     20
1112 016326  052703  000001          BIS     #BIT0,R3        ;SET ERROR FLAG
1113                             ;WR*TE ZEROS IN RMDA, *HEN WRITE ONES, READ BACK AND CHECK FOR S-A-0 BITS
1114 016332                      10$:
016332  012760  000000  000006       MOV     #0,RMDA(R0)     ;LOAD RMDA
1115 016340  012760  177777  000006   MOV     #-1,RMDA(R0)    ;LOAD RMDA
1116 016346  016037  000006  001142   MO.     RMDA(R0),$BDDA* ;STORE RMDA AT $BDDAT
1117 016354  023727  001142  177777   CMP     $BDDAT,#-1
1118 016362  001413                  BEQ     20$
1119 016364  012737  177777  001*40   MOV     #-1,$GDDAT
1120 016372  010037  001*36          MOV     R0,$BDADR
1121 016376  062737  000006  001*36   ADD     #RMDA,$BDADR
```

```
     016404  104021                           EMT     21
     016406  052703  000002                   BIS     #BIT1,R3          ;SET ERROR FLAG
                                       ;WRITE A SHIFTING 1 BIT PATTERN IN RMDA, READ, AND CHECK FOR STUCK BITS
     016412  005703                   20$:    TST     R3                ;OMIT BIT TEST IF ANY ERROR
     016414  001027                           BNE     50$
     016416  012702  000001                   MOV     #1,R2             ;R2 = TEST PATTERN
     016422                           30$:
     016422  012760  000000  000006           MOV     #0,RMDA(R0)       ;LOAD RMDA
     016430  010260  000006                   MOV     R2,RMDA(R0)       ;LOAD RMDA
     016434  016037  000006  001142           MOV     RMDA(R0),$BDDAT   ;STORE RMDA AT $BDDAT
     016442  023702  001142                   CMP     $BDDAT,R2
     016446  001410                           BEQ     40$
     016450  010237  001140                   MOV     R2,$GDDAT
     016454  010037  001136                   MOV     R0,$BDADR
     016460  062737  000006  001136           ADD     #RMDA,$BDADR
     016466  104022                           EMT     22
     016470  006302                   40$:    ASL     R2                ;SHIFT TO NEXT BIT
     016472  001353                           BNE     30$               ;CONTINUE IF R2 NOT ZERO
     016474                           50$:

                                       ;*************************************************************
                                       ;*TEST 24        DESIRED CYLINDER TRANSFER TEST

                                       ;*************************************************************
     016474                           T24:
     016474  000004                           SCOPE*                    ;SCOPE CALL
     016476  000240                           NOP
     016500  012706  001100                   MOV     #STACK,SP         ;LOAD THE STACK POINTER
     016504  013700  001276                   MOV     $BASE,R0          ;R0 = UNIBUS ADDRESS
     016510  013701  001466                   MOV     TSTQUE,R1         ;R1 = POINTER TO DEVICE
     016514  012737  000024  001226           MOV     #24,$TESTN        ;;SET TEST NUMBER IN APT MAIL BOX

     016522  005003                           CLR     R3                ;RESET ERROR FLAGS
     016524  004737  054674                   JSR     PC,CNTCLR         ;GO CLEAR CONTROLLER
     016530  005037  001140                   CLR     $GDDAT            ;LOAD EXPECTED
     016534  010037  001136                   MOV     R0,$BDADR         ;LOAD REG ADDRESS
     016540  062737  000034  001136           ADD     #RMDC,$BDADR

                                       ;WRITE ONES IN RMDC AND VERIFY THAT UNUSED BITS ARE ZERO
     016546  012760  177777  000034           MOV     #-1,RMDC(R0)      ;LOAD RMDC
     016554  016037  000034  001142           MOV     RMDC(R0),$BDDAT   ;STORE RMDC AT $BDDAT
     016562  042737  001777  001142           BIC     #^CXNUDC,$BDDAT            ;CLEAR ALL USED BITS
     016570  001403                           BEQ     5$                         ;BRANCH IF NO ERROR
     016572  104134                           EMT     134
     016574  052703  000001                   BIS     #BIT0,R3          ;SET ERROR FLAG

                                       ;WRITE ONES IN RMDC, THEN WRITE ZEROS, READ AND CHECK FOR S-A-1 BITS
     016600                           5$:
     016600  012760  177777  000034           MOV     #-1,RMDC(R0)      ;LOAD RMDC
     016606  012760  000000  000034           MOV     #0,RMDC(R0)       ;LOAD RMDC
     016614  016037  000034  001142           MOV     RMDC(R0),$BDDAT   ;STORE RMDC AT $BDDAT
     016622  042737  176000* 001142           BIC     #XNUDC,$BDDAT     ;CLEAR UNUSED BITS
     016630  001403                           BEQ     10$               ;BRANCH IF NO ERROR
     016632  104037                           EMT     37
     016634  052703  000001                   BIS     #BIT0,R3          ;SET ERROR FLAG

                                       ;WRITE ZEROS, THEN ONES IN RMDC, READ AND CHECK FOR S-A-0 BITS
```

```
016640                              10$:
016640  012760  000000  000034              MOV     #0,RMDC(R0)         ;LOAD RMDC
016646  012760  177777  000034              MOV     #-1,RMDC(R0)        ;LOAD RMDC
016654  016037  000034  001142              MOV     RMDC(R0),$BDDAT     ;STORE RMDC AT $BDDAT
016662  052737  176000  001142              BIS     #XMUDC,$BDDAT       ;SET UNUSED BITS
016670  012737  177777  001140              MOV     #-1,$GDDAT          ;LOAD EXPECTED RESULT
016676  023737  001140  001142              CMP     $GDDAT,$BDDAT       ;IS RMDC ALL ONES ??
016704  001403                              BEQ     20$                 ;YES !!
016706  104040                              EMT     40
016710  052703  000002                      BIS     #BIT1,R3            ;SET ERROR FLAG

                                    ;OMIT BIT TEST IF ANY ERRORS
016714                              20$:
016714  005703                              TST     R3
016716  001026                              BNE     60$
016720  012702  000001                      MOV     #1,R2               ;R2 = TEST PATTERN

                                    ;TEST RMDC WITH SHIFTING ONE BIT
016724                              30$:
016724  012760  000000  000034              MOV     #0,RMDC(R0)         ;LOAD RMDC
016732  010260  000034                      MOV     R2,RMDC(R0)         ;LOAD RMDC
016736  010203                              MOV     R2,R3               ;R3 = EXPECTED RESULT
016740  042703  176000                      BIC     #XMUDC,R3           ;CLEAR ANY UNUSED BITS
016744  016037  000034  001142              MOV     RMDC(R0),$BDDAT     ;STORE RMDC AT $BDDAT
016752  023703  001142                      CMP     $BDDAT,R3
016756  001404                              BEQ     50$
016760  010337  001140                      MOV     R3,$GDDAT
016764  104041                              EMT     41
016766  000402                              BR      60$                 ;SKIP TO NEXT

016770  006302                      50$:    ASL     R2                  ;SHIFT TO NEXT BIT
016772  001354                              BNE     30$                 ;CONTINUE IF R2 NOT ZERO

016774                              60$:

                                    ;;*********************************************************
                                    ;;*TEST 25          ILLEGAL REGISTER TEST
                                    ;;*********************************************************
016774                              TST25:
016774  000004                              SCOPE                       ;SCOPE CALL
016776  000240                              NOP
017000  012706  001100                      MOV     #STACK,SP           ;LOAD THE STACK POINTER
017004  013700  001276                      MOV     $BASE,R0            ;R0 - UNIBUS ADDRESS
017010  013701  001466                      MOV     TSTQUE,R1           ;R1 - POINTER TO DEVICE
017014  012737  000025  001226              MOV     #25,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX

017022  005037  001140                      CLR     $GDDAT              ;"ILR" SHOULD BE ZERO
017026  012702  000000                      MOV     #0,R2               ;R2=REGISTER INDEX

                                    ;CLEAR AND VERIFY THAT "ILR" STATUS IS ZERO
017032                              10$:
017032  004737  054674                      JSR     PC,CNTCLR           ;GO CLEAR CONTROLLER
017036  016037  000014  001142              MOV     RMER1(R0),$BDDAT            ;STORE RMER1 AT $BDDAT
017044  042737  177775  001142              BIC     #^CILR,$BDDAT
017052  001411                              BEQ     20$                 ;BRANCH IF ILR IS RESET
017054  005037  001140                      CLR     $GDDAT              ;SETUP GOOD DATA, REG ADR
```

```
1215 017060  010037  001134                    MOV     R0,$BDADR
1216 017064  062737  000004  001134            ADD     #RMER1,$BDADR
1217 017072  104064                            EMT     64
1218 017074  000550                            BR      140$
1219
1220                                   ;WRITE THE REGISTER (INDEX=R2) AND TEST ILR STATUS
1221 017076                            20$:
1222 017076  010003                            MOV     R0,R3           ;R3=REG ADDRESS
1223 017100  060203                            ADD     R2,R3
1224 017102  013746  000004                    MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
1225 017106  013746  000006                    MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
1226 017112  012737  017154  000004            MOV     #40$,ERRVEC     ;SETUP FOR BUS TIMEOUT
1227 017120  012737  000300  000006            MOV     #PR6,ERRVEC+2
1228 017126  005004                            CLR     R4              ;R4=REGISTER VALUE
1229 017130  022702  000010                    CMP     #RMCS2,R2
1230 017134  001001                            BNE     30$
1231 017136  111104                            MOVB    (R1),R4         ;SELECT DRIVE IF RMCS2
1232 017140  010413                   30$:     MOV     R4,(R3)         ;WRITE TEST REGISTER
1233 017142  012637  000006                    MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
1234 017146  012637  000004                    MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
1235 017152  000416                            BR      60$
1236 017154  012716  017162           40$:     MOV     #45$,(SP)       ;DUMMY RTI ADDRESS
1237 017160  000002                            RTI                     ;RESTORE PRIORITY
1238 017162                            45$:
     017162  012637  000006                    MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
1239 017166  012637  000004                    MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
1240 017172  020227  000046                    CMP     R2,#RMEC2       ;WERE ALL REGISTERS READ??
1241 017176  101003                            BHI     50$             ;YES !
1242 017200  010337  001136                    MOV     R3,$BDADR
1243 017204  104074                            EMT     74
1244 017206  000503                   50$:     BR      140$
1245
1246 017210                            60$:
     017210  016037  000014  001142            MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
1247 017216  042737  177775  001142            BIC     #^C<ILR>,$BDDAT
1248 017224  023737  001140  001142            CMP     $GDDAT,$BDDAT   ;IS "ILR" OK??
1249 017232  001411                            BEQ     80$             ;YES !
1250 017234  010337  001174                    MOV     R3,$TMP0        ;SAVE ADDRESS
1251 017240  032737  000002  001140            BIT     #ILR,$GDDAT     ;SHOULD "ILR" BE SET??
1252 017246  001002                            BNE     70$             ;YES!!
1253 017250  104065                            EMT     65
1254 017252  000401                            BR      80$
1255 017254                            70$:
     017254  104066                            EMT     66
1256
1257                                   ;ADVANCE TO THE NEXT REGISTER ADDRESS
1258 017256                            80$:
1259 017256  062702  000002                    ADD     #2,R2           ;INCREMENT INDEX
1260 017262  022702  000050                    CMP     #50,R2          ;TIME TO TRY RH70 ?
1261 017266  101261                            BHI     10$             ;BRANCH IF NOT
1262 017270  103437                            BLO     110$            ;BRANCH IF ALREADY CHECKED
1267 017272  013746  000004                    MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
1268 017276  013746  000006                    MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
1269 017302  012737  017400  000004            MOV     #130$,ERRVEC    ;SETUP FOR TIMEOUT
1270 017310  012737  000300  000006            MOV     #PR6,ERRVEC+2
1283 017316  052737  000002  001140            BIS     #ILR,$GDDAT     ;SET ILR
1284 017324  012702  000054                    MOV     #54,R2          ;START AT INDEX 54 IF RH70 WITH 22 REG
```

```
017330  012760  001400  [illegible]         MOV     #A17!A16,RMCS1(R0)      ;SET EXTEND BITS
017336  016037  000050  [illegible]         MOV     50(R0),RMBAEI           ;READ THE EXTENDED BITS
017344  042737  177774  [illegible]         BIC     #177774,RMBAEI          ;CHOP OFF
017352  001002                              BNE     90$                     ;BRANCH IF RH70 WITH 22 REG
017354  012702  000050                      MOV     #50,R2                  ;OTHERWISE NOT A RH70 OR RH70 WITH 32 REG
017360  012637  000006          [illegible]$:  MOV  (SP)+,ERRVEC+2          ;
017364  012637  000004                      MOV     (SP)+,ERRVEC            ;
017370  022702  000074          [illegible]$:  CMP  #74,R2                  ;DONE ALL TESTS
017374  101410                              BLOS    140$                    ;YES..
017376  000615                  [illegible]$:  BR   [illegible]$
017400  012716  017406          130$:       MOV     #[illegible]$,(SP)      ;DUMMY RTI ADDRESS
017404  000002                              RTI                             ;RESTORE PRIORITY
017406                          [illegible]$:
017406  012637  000004                      MOV     (SP)+,ERRVEC            ;;POP STACK INTO ERRVEC
017412  012637  000006                      MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
017416                          140$:                                       ;END OF TEST

                                ;;*************************************************************
                                ;*TEST 26        RESET GO BY INIT TEST
                                ;;*************************************************************
017416                          TST26:
017416  000004                              SCOPE                           ;SCOPE CALL
017420  000240                              NOP
017422  012706  001100                      MOV     #STACK,SP               ;LOAD THE STACK POINTER
017426  013700  001276                      MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
017432  013701  001466                      MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
017436  012737  000026  001226              MOV     #26,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

017444  004737  054674                      JSR     PC,CNTCLR               ;GO CLEAR CONTROLLER
017450  010037  001136                      MOV     R0,$BDADR               ;SETUP REGISTER ADDRESS FOR MSG

                                ;SET GO, INITIALIZE AND VERIFY THAT GO IS RESET
017454  012760  000001  000000              MOV     #GO,RMCS1(R0)           ;LOAD RMCS1

017462  004737  054674                      JSR     PC,CNTCLR               ;GO CLEAR CONTROLLER
017466  016037  000000  001142              MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
017474  042737  177776  001142              BIC     #^CGO,$BDDAT
017502  001403                              BEQ     10$                     ;BRANCH IF GO IS RESET
017504  005037  001140                      CLR     $GDDAT
017510  104137                              EMT     137

017512                          10$:                                        ;END OF TEST

                                ;;*************************************************************
                                ;*TEST 27         DIAGNOSTIC MODE TEST
                                ;;*************************************************************
017512                          TST27:
017512  000004                              SCOPE                           ;SCOPE CALL
017514  000240                              NOP
017516  012706  001100                      MOV     #STACK,SP               ;LOAD THE STACK POINTER
017522  013700  001276                      MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
017526  013701  001466                      MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
017532  012737  000027  001226              MOV     #27,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
```

```
017540  010037  001134            MOV     R0,$SBDADR        ;SETUP REGISTER ADDRESS
017544  062737  000024  001134    ADD     #RMMR1,$SBDADR
017552  005003                    CLR     R3                ;INITIALIZE ERROR FLAGS

                                  ;INITIALIZE AND VERIFY THAT 'DMD'' IS RESET
017554  004737  054674            JSR     PC,INITCLR        ;GO CLEAR CONTROLLER
017560  016037  000024  001142    MOV     RMMR1(R0),$BDDAT          ;STORE RMMR1 AT $BDDAT
017566  042737  177776  001142    BIC     #^CDMD,$BDDAT
017574  001403                    BEQ     1$                ;BRANCH IF 'DMD'' IS ZERO
017576  005037  001140            CLR     $GDDAT
017602  104075                    EMT     75

                                  ;SET AND RESET 'DMD'' USING REGISTER TRANSFER-VERIFY 'DMD'' NOT S-A-1
017604                            1$:
017604  012760  000001  000024    MOV     #DMD,RMMR1(R0)    ;LOAD RMMR1
017612  012760  000000  000024    MOV     #0,RMMR1(R0)      ;LOAD RMMR1
017620  016037  000024  001142    MOV     RMMR1(R0),$BDDAT          ;STORE RMMR1 AT $BDDAT
017626  042737  177776  001142    BIC     #^CDMD,$BDDAT
017634  001405                    BEQ     20$               ;BRANCH IF DMD NOT S-A-1
017636  005037  001140            CLR     $GDDAT
017642  104076                    EMT     76
017644  052703  000001            BIS     #BIT0,R3          ;SET ERROR FLAG

                                  ;RESET AND SET 'DMD'' USING REGISTER TRANSFER-VERIFY 'DMD'' NOT S-A-0
017650                            20$:
017650  012760  000000  000024    MOV     #0,RMMR1(R0)      ;LOAD RMMR1
017656  012760  000001  000024    MOV     #DMD,RMMR1(R0)    ;LOAD RMMR1
017664  016037  000024  001142    MOV     RMMR1(R0),$BDDAT          ;STORE RMMR1 AT $BDDAT
017672  042737  177776  001142    BIC     #^CDMD,$BDDAT
017700  001006                    BNE     30$               ;BRANCH IF DMD NOT S-A-0
017702  012737  000001  001140    MOV     #DMD,$GDDAT
017710  104077                    EMT     77
017712  052703  000002            BIS     #BIT1,R3          ;SET ERROR FLAG

                                  ;IF NO PREVIOUS ERROR, TEST FOR BIT INTERFERENCE WITH SHIFTING
                                  ;ONE BIT PATTERN
017716                            30$:
017716  005703                    TST     R3                ;ANY ERRORS DETECTED??
017720  001027                    BNE     60$               ;YES!!
017722  012702  000002            MOV     #BIT1,R2          ;INITAILIZE DATA PATTERN
017726                            40$:
017726  012760  000000  000024    MOV     #0,RMMR1(R0)      ;LOAD RMMR1
017734  010260  000024            MOV     R2,RMMR1(R0)      ;LOAD RMMR1
017740  016037  000024  001142    MOV     RMMR1(R0),$BDDAT          ;STORE RMMR1 AT $BDDAT
017746  042737  177776  001142    BIC     #^CDMD,$BDDAT
017754  001407                    BEQ     50$               ;BRANCH IF DMD NOT SET
017756  010237  001174            MOV     R2,$TMP0          ;SAVE DATA
017762  010237  001174            MOV     R2,$TMP0          ;SAVE DATA
017766  005037  001140            CLR     $GDDAT            ;DMD SHOULD BE ZERO
017772  104100                    EMT     100

                                  ;SHIFT TO NEXT DATA BIT AND CONTINUE TEST IF NOT DONE
017774                            50$:
017774  006302                    ASL     R2
017776  001353                    BNE     40$
020000                            60$:                      ;END OF TEST
```

```
                                        ;;*****************************************************************
                                        ;;TEST 30          MOL TEST
                                        ;;
                                        ;;*****************************************************************
      020000                            TST30:
      020000  000004                           SCOPE                            ;SCOPE CALL
      020002  000240                           NOP
      020004  012706  001100                   MOV     #STACK,SP                ;LOAD THE STACK POINTER
      020010  013700  001276                   MOV     $BASE,R0                 ;R0 = UNIBUS ADDRESS
      020014  013701  001466                   MOV     TSTQUE,R1                ;R1 - POINTER TO DEVICE
      020020  012737  000030  001226           MOV     #30,$TESTN               ;;SET TEST NUMBER IN APT MAIL BOX

      020026  004737  054674                   JSR     PC,CNTCLR                ;GO CLEAR CONTROLLER
      020032  010037  001136                   MOV     R0,$BDADR                ;R0=REGISTER ADDRESS
      020036  062737  000012  001136           ADD     #RMDS,$BDADR
      020044  005003                           CLR     R3                       ;R3=ERROR FLAG

                                        ;SET DIAGNOSTIC MODE AND VERIFY THAT 'MOL'' IS ZERO
      020046  012760  000001  000024           MOV     #DMD,RMMR1(R0)           ;LOAD RMMR1
      020054  016037  000012  001142           MOV     RMDS(R0),$BDDAT          ;STORE RMDS AT $BDDAT
      020062  042737  167777  001142           BIC     #^CMOL,$BDDAT
      020070  001405                           BEQ     10$
      020072  005037  001140                   CLR     $GDDAT
      020076  104101                           EMT     101
      020100  052703  000001                   BIS     #BIT0,R3                 ;SET ERROR FLAG

                                        ;SET MAINTENANCE UNIT READY AND VERIFY THAT 'MOL'' IS ONE
      020104                            10$:
      020104  012760  000001  000024           MOV     #DMD,RMMR1(R0)           ;LOAD RMMR1
      020112  012760  001001  000024           MOV     #DMD!MUR,RMMR1(R0)               ;LOAD RMMR1
      020120  016037  000012  001142           MOV     RMDS(R0),$BDDAT          ;STORE RMDS AT $BDDAT
      020126  042737  167777  001142           BIC     #^CMOL,$BDDAT
      020134  001006                           BNE     20$
      020136  012737  010000  001140           MOV     #MOL,$GDDAT
      020144  104102                           EMT     102
      020146  052703  000002                   BIS     #BIT1,R3

                                        ;IF NO PREVIOUS ERROR, VERIFY VOLUME VALID IS RESET AND
                                        ;TEST FOR BIT INTERFERENCE
      020152                            20$:
      020152  005703                           TST     R3                       ;ANY ERROR DETECTED??
      020154  001057                           BNE     70$                      ;YES !
      020156  016037  000012  001142           MOV     RMDS(R0),$BDDAT          ;STORE RMDS AT $BDDAT
      020164  042737  177677  001142           BIC     #^CVV,$BDDAT
      020172  001403                           BEQ     25$                      ;BRANCH IF VV RESET
      020174  005037  001140                   CLR     $GDDAT
      020200  104135                           EMT     135
      020202                            25$:
      020202  012702  000001                   MOV     #1,R2                    ;INITIALIZE DATA PATTERN
      020206                            30$:
      020206  012760  000001  000024           MOV     #DMD,RMMR1(R0)           ;LOAD RMMR1
      020214  010260  000024                   MOV     R2,RMMR1(R0)             ;LOAD RMMR1
      020220  016037  000012  001142           MOV     RMDS(R0),$BDDAT          ;STORE RMDS AT $BDDAT
      020226  042737  167777  001142           BIC     #^CMOL,$BDDAT
      020234  005003                           CLR     R3                       ;SETUP EXPECTED 'MOL''
      020236  032702  001000                   BIT     #MUR,R2
      020242  001402                           BEQ     40$
```

```
020244 052703 000000          BIS     #MOL,R3           ;"MOL" SHOULD BE ONE
020250 020337 001142    4$:   CMP     R3,$BDDAT         ;IS MOL OK??
020254 001405                 BEQ     10$               ;YES..
020256 010237 001174          MOV     R2,$TMP0          ;SAVE TEST PATTERN
020262 010337 001140          MOV     R3,$GDDAT
020266 104103                 EMT     103

                        ;SHIFT TO NEXT PATTERN
020270                  10$:
020270 042702 000001          BIC     #DMD,R2           ;DONT SHIFT DMD
020274 001002                 BNE     20$
020276 012702 000001          MOV     #DMD,R2           ;DONT TRUNCATE TEST
020302 006302           20$:  ASL     R2                ;SHIFT TO NEXT BIT
020304 001403                 BEQ     30$               ;BRANCH IF DONE
020306 052702 000001          BIS     #DMD,R2           ;KEEP DMD ON
020312 000735                 BR      3$                ;CONTINUE

020314                  30$:                            ;END OF TEST

                        ;*********************************************************
                        ;*TEST 31         WRITE LOCK TEST
                        ;*********************************************************
       020314           T31::
       020314 000004          SCOPE                     ;SCOPE CALL
       020316 000240          NOP
       020320 012706 001100   MOV     #STACK,SP         ;LOAD THE STACK POINTER
       020324 013700 001276   MOV     $BASE,R0          ;R0 = UNIBUS ADDRESS
       020330 013701 001466   MOV     @STOUE,R1         ;R1 = POINTER TO DEVICE
       020334 012737 000031 001226   MOV #31,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX

020342 005003                 CLR     R3                ;R3=ERROR FLAG
020344 010037 001136          MOV     R0,$BDADR         ;SETUP REGISTER ADDRESS
020350 062737 000012 001136   ADD     #RMDS,$BDADR
020356 005037 001140          CLR     $GDDAT            ;WRL SHOULD BE ZERO
020362 004737 054674          JSR     PC,CNTCLR         ;GO CLEAR CONTROLLER

                        ;SET DIAGNOSTIC MODE AND VERIFY "WRL" IS ZERO
020366 012760 000001 000024   MOV     #DMD,RMMR1(R0)    ;LOAD RMMR1
020374 016037 000012 001142   MOV     RMDS(R0),$BDDAT   ;STORE RMDS AT $BDDAT
020402 042737 173777 001142   BIC     #^C WRL,$BDDAT
020410 001403                 BEQ     10$               ;BRANCH IF WRL IS ZERO
020412 104104                 EMT     104
020414 052703 000001          BIS     #BIT0,R3          ;SET ERROR FLAG

                        ;SET MAINTENANCE WRITE PROTECT AND VERIFY "WRL" IS ONE
020420                  10$:
020420 012760 000001 000024   MOV     #DMD,RMMR1(R0)    ;LOAD RMMR1
020426 012760 000011 000024   MOV     #DMD!MWP,RMMR1(R0)       ;LOAD RMMR1
020434 016037 000012 001142   MOV     RMDS(R0),$BDDAT   ;STORE RMDS AT $BDDAT
020442 042737 173777 001142   BIC     #^C WRL,$BDDAT
020450 001006                 BNE     20$               ;BRANCH IF WRL IS NOE
020452 012737 004000 001140   MOV     #WRL,$GDDAT       ;WRL SHOULD BE SET
020460 104105                 EMT     105
020462 052703 000002          BIS     #BIT1,R3          ;SET ERROR FLAG

                        ;IF NO PREVIOUS ERROR, TEST FOR BIT INTERFERENCE ON "MWP"
```

```
020466                                   10$:
020466  005703                                   TST     R3                      ;ANY OTHER ERROR??
020470  001045                                   BNE     70$                     ;YES.!
020472  012702  000001                           MOV     #1,R2                   ;INITIALIZE DATA PATTERN

                                         ;TRANSFER DATA TO RMMR1, READ RMDS AND VERIFY WRL
020476                                   30$:
020476  012760  000001  000024                   MOV     #DMD,RMMR1(R0)          ;LOAD RMMR1
020504  010260  000024                           MOV     R2,RMMR1(R0)            ;LOAD RMMR1
020510  016037  000012  001142                   MOV     RMDS(R0),$BDDAT         ;STORE RMDS AT $BDDAT
020516  042737  173777  001142                   BIC     #^C<WRL,$BDDAT          ;CLEARUP RECEIVED "WRL"
020524  005003                                   CLR     R3                      ;GENERATE EXPECTED "WRL"
020526  032702  000010                           BIT     #WP,R2
020532  001402                                   BEQ     40$
020534  052703  004000                           BIS     #WRL,R3                 ;WRL SHOULD BE SET
020540  020337  001142                   40$:    CMP     R3,$BDDAT               ;IS WRL OK??
020544  001405                                   BEQ     50$                     ;YES!!
020546  010337  001140                           MOV     R3,$GDDAT               ;SAVE EXPECTED
020552  010237  001174                           MOV     R2,$TMPC                ;SAVE DATA PATTERN
020556  104106                                   ERROR   106

                                         ;ADVANCE TO NEXT DATA BIT
020560                                   50$:
020560  042702  000001                           BIT     #DMD,R2                 ;DONT SHIFT DMD
020564  001002                                   BNE     60$
020566  012702  000001                           MOV     #DMD,R2                 ;DONT TRUNCATE TEST
020572  006302                           60$:    ASL     R2                      ;SHIFT DATA BIT
020574  001403                                   BEQ     70$                     ;EXIT IF DONE
020576  052702  000001                           BIS     #DMD,R2                 ;KEEP DIAGNOSTIC MODE ON
020602  000735                                   BR      30$                     ;CONTINUE TEST

020604                                   70$:                                    ;END OF TEST

                                         ;;*****************************************************************
                                         ;*TEST 32        DRIVE FAULT TEST

                                         ;;*****************************************************************
020604                                   TST32:
020604  000004                                   SCOPE                           ;SCOPE CALL
020606  000240                                   NOP
020610  012706  001100                           MOV     #STACK,SP               ;LOAD THE STACK POINTER
020614  013700  001276                           MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
020620  013701  001466                           MOV     STODE,R1                ;R1 = POINTER TO DEVICE
020624  012737  000032  001226                   MOV     #32,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX

020632  004737  054674                           JSR     PC,CNTCLR               ;GO CLEAR CONTROLLER
020636  005003                                   CLR     R3                      ;INITIALIZE ERROR FLAGS
020640  010037  001136                           MOV     R0,$BDADR               ;SETUP REGISTER ADDRESS
020644  062737  000042  001136                   ADD     #RMER2,$BDADR
020652  005037  001140                           CLR     $GDDAT                  ;"DVC" AND "UNS" SHOULD BE ZERO

                                         ;SET AND RESET MAINTENANCE DRIVE FAULT, VERIFY THAT "DVC" IS NOT
                                         ;STUCK-AT-ONE.
020656  012760  000001  000024                   MOV     #DMD,RMMR1(R0)          ;LOAD RMMR1
020664  012760  000000  000042                   MOV     #0,RMER2(R0)            ;LOAD RMER2
020672  012760  000000  000014                   MOV     #0,RMER1(R0)            ;LOAD RMER1
020700  016037  000042  001142                   MOV     RMER2(R0),$BDDAT                ;STORE RMER2 AT $BDDAT
```

```
1517 020706  042737  177577  00****2          BIC     #*(DVC,$BDDAT   ;IS "DVC" RESET??
1518 020714  001406                            BEQ     10$             ;YES.!
1519 020716  104107                            EMT     107
1520 020720  052703  000001                    BIS     #B:T0,R3        ;SET ERROR FLAG
1521 020724  012737  040000  00****0          MOV     #UNS,$GDDAT
1522
1523                                    ;VERIFY THAT "UNS" IS SAME AS "DVC"
1524 020732                             10$:
1525 020732  016037  000014  00****2          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
1526 020740  042737  137777  00****2          BIC     #*(UNS,$BDDAT
1527 020746  023737  001140  00****2          CMP     $GDDAT,$BDDAT   ;IS "UNS" OK??
1528 020754  001414                            BEQ     30$             ;YES!.
1529 020756  010037  00****6                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1530 020762  062737  000014  00****6          ADD     #RMER1,$BDADR
1531 020770  032737  040000  00****0          BIT     #UNS,$GDDAT     ;SHOULD "UNS" BE ON??
1532 020776  001002                            BNE     20$             ;YES!!
1533 021000  104110                            EMT     **0
1534 021002  000401                            BR      30$
1535 021004                             20$:
1536 021004  104***                            EMT     ***
1537
1538                                    ;RESET AND SET "MDF", VERIFY THAT "DVC" IS NOT S-A-0.
1539 021006                             30$:
1540 021006  012737  000200  00****0          MOV     #DVC,$GDDAT     ;DVC SHOULD BE ON
1541 021014  012760  000001  000024           MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1542 021022  012760  000101  000024           MOV     #DMD!MDF,RMMR1(R0)      ;LOAD RMMR1
1543 021030  016037  000042  00****2          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
1544 021036  042737  177577  00****2          BIC     #*(DVC,$BDDAT
1545 021044  001012                            BNE     40$             ;BRANCH IF DVC IS SET
1546 021046  010037  001136                    MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1547 021052  062737  000042  00****6          ADD     #RMER2,$BDADR
1548 021060  104112                            EMT     112
1549 021062  052703  000002                    BIS     #BIT1,R3        ;SET ERROR FLAG
1550 021066  005037  001140                    CLR     $GDDAT          ;UNS SHOULD BE OFF
1551
1552                                    ;VERIFY THAT "UNS" IS SAME AS "DVC"
1553 021072                             40$:
1554 021072  005737  001140                    TST     $GDDAT          ;CHANGE DVC TO UNS
1555 021076  001403                            BEQ     50$
1556 021100  012737  040000  00****0          MOV     #UNS,$GDDAT
1557 021106                             50$:
1558 021106  016037  000014  001142           MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
1559 021114  042737  137777  001142           BIC     #*(UNS,$BDDAT
1560 021122  023737  001140  001142           CMP     $GDDAT,$BDDAT
1561 021130  001414                            BEQ     70$             ;BRANCH IF UNS IS OK
1562 021132  010037  001136                    MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1563 021136  062737  000014  001136           ADD     #RMER1,$BDADR
1564 021144  032737  040000  001140           BIT     #UNS,$GDDAT     ;SHOULD UNS BE ON??
1565 021152  001002                            BNE     60$             ;YES!!
1566 021154  104113                            EMT     113
1567 021156  000401                            BR      70$
1568 021160                             60$:
1569 021160  104114                            EMT     114
1570
1571                                    ;IF THERE WERE NO PREVIOUS ERRORS, TEST FOR BIT INTERFERENCE ON
1572                                    ;"MDF"
1573 021162                             70$:
```

```
1574 021162  005703                          TST     R3
1575 021164  001056                          BNE     120$            ;BRANCH IF ANY OTHER ERRORS
1576 021166  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
1577 021172                          8$:
     021172  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1578 021200  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
1579 021206  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
1580 021212  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
1581 021220  042737  177577  001142          BIC     #^CDVC,$BDDAT   ;GET RESULTS
1582 021226  005037  001140                  CLR     $GDDAT          ;SETUP EXPECTED
1583 021232  032702  000100                  BIT     #MDF,R2         ;WAS MDF SET??
1584 021236  001403                          BEQ     90$             ;NO!!
1585 021240  052737  000200  001140          BIS     #DVC,$GDDAT     ;YES-DVC SHOULD BE ON
1586 021246  023737  001140  001142  90$:    CMP     $GDDAT,$BDDAT
1587 021254  001410                          BEQ     100$            ;BRANCH IF DVC IS OK
1588 021256  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1589 021262  062737  000042  001136          ADD     #RMER2,$BDADR
1590 021270  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
1591 021274  104115                          EMT     115
1592
1593                                 ;SHIFT TO NEXT BIT POSITION
1594 021276                          100$:
1595 021276  042702  000001                  BIC     #DMD,R2         ;DONT SHIFT DMD
1596 021302  001002                          BNE     110$
1597 021304  012702  000001                  MOV     #DMD,R2         ;DONT TRUNCATE TEST
1598 021310  006302                  110$:   ASL     R2              ;SHIFT
1599 021312  001403                          BEQ     120$            ;EXIT IF DONE
1600 021314  052702  000001                  BIS     #DMD,R2         ;KEEP DMD ON
1601 021320  000724                          BR      80$             ;CONTINUE
1602
1603 021322                          120$:                           ;END OF TEST
1604
1605                                 ;*****************************************************
                                     ;*TEST 33        SEEK ERROR TEST
                                     ;*****************************************************
     021322                          ;*T33:
     021322  000004                          SCOPE                   ;SCOPE CALL
     021324  000240                          NOP
     021326  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     021332  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     021336  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     021342  012737  000033  001226          MOV     #33,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1606
1607 021350  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1608 021354  005003                          CLR     R3              ;CLEAR ERROR FLAGS
1609 021356  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1610 021362  062737  000042  001136          ADD     #RMER2,$BDADR
1611
1612                                 ;SET DIAGNOSTIC MODE AND VERIFY THAT "SKI" CAN BE RESET
1613 021370  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1614 021376  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
1615 021404  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
1616 021412  042737  137777  001142          BIC     #^CSKI,$BDDAT
1617 021420  001405                          BEQ     10$             ;BRANCH IF SKI IS RESET
1618 021422  005037  001140                  CLR     $GDDAT          ;SKI SHOULD BE ZERO
1619 021426  104116                          EMT     116
```

```
                                        BIS     #BIT0,R3        ;SET ERROR FLAG

                                ;SET MAINTENANCE SEEK ERROR AND VERIFY THAT "SKI" CAN BE SET
                                10$:
021434  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
021442  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
021450  012760  000201  000024          MOV     #DMD!MSER,RMMR1(R0)     ;LOAD RMMR1
021456  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
021464  042737  137777  001142          BIC     #^C<SKI>,$BDDAT
021472  001005                          BNE     20$             ;BRANCH IF SKI IS SET
021474  012737  040000  001140          MOV     #SKI,$GDDAT     ;CANT SET SKI
021502  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG

                                ;IF NO PREVIOUS ERROR, CHECK FOR BIT INTERFERENCE SETTING MAINTENANCE
                                ;SEEK ERROR.
021506                          20$:
021506  005703                          TST     R3
021510  001051                          BNE     70$             ;BRANCH IF ANY OTHER ERRORS
021512  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
021516                          30$:
021516  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
021524  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
021532  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
021536  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
021544  042737  137777  001142          BIC     #^C<SKI>,$BDDAT ;GET SKI STATUS
021552  005037  001140                  CLR     $GDDAT          ;SETUP EXPECTED RESULT
021556  032702  000200                  BIT     #MSER,R2
021562  001403                          BEQ     40$
021564  052737  040000  001140          BIS     #SKI,$GDDAT     ;SKI SHOULD BE ON
021572  023737  001140  001142  40$:    CMP     $GDDAT,$BDDAT
021600  001403                          BEQ     50$             ;BRANCH IF SKI IS OK
021602  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
021606  104120                          EMT     120

                                ;ADVANCE TEST PATTERN IN R2
021610                          50$:
021610  042702  000001                  BIC     #DMD,R2         ;DONT SHIFT DMD BIT
021614  001002                          BNE     60$
021616  012702  000001                  MOV     #DMD,R2         ;DONT TRUNCATE TEST
021622  006302                  60$:    ASL     R2              ;SHIFT TO NEXT BIT
021624  001403                          BEQ     70$             ;EXIT IF DONE
021626  052702  000001                  BIS     #DMD,R2         ;KEEP DMD ON
021632  000731                          BR      30$             ;CONTINUE TEST

021634                          70$:                            ;END OF TEST

                                ;;*****************************************************************
                                ;*TEST 34       PIP TEST
                                ;;*****************************************************************
021634                          T34:
021634  000004                          SCOPE                   ;SCOPE CALL
021636  000240                          NOP
021640  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
021644  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
021650  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
021654  012737  000034  001226          MOV     #34,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

```
1667
1668 021662  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1669 021666  005003                          CLR     R3              ;RESET ERROR FLAGS
1670 021670  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1671 021674  062737  000012  001136          ADD     #RMDS,$BDADR
1672
1673                                 ;SET MAINTENANCE ON CYLINDER 'MOC'' AND VERIFY THAT 'PIP'' CAN BE RESET.
1674 021702  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1675 021710  012760  000401  000024          MOV     #DMD!MOC,RMMR1(R0)      ;LOAD RMMR1
1676 021716  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
1677 021724  042737  157777  001142          BIC     #^CPIP,$BDDAT
1678 021732  001405                          BEQ     10$             ;BRANCH IF PIP IS RESET
1679 021734  005037  001140                  CLR     $GDDAT
1680 021740  104121                          EMT     121
1681 021742  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
1682
1683                                 ;RESET MAINTENANCE ON CYLINDER AND VERIFY THAT 'PIP'' CAN BE SET
1684 021746                          10$:
1685 021746  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1686 021754  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
1687 021762  042737  157777  001142          BIC     #^CPIP,$BDDAT
1688 021770  001006                          BNE     20$             ;BRANCH IF PIP IS SET
1689 021772  012737  020000  001140          MOV     #PIP,$GDDAT
1690 022000  104122                          EMT     122
1691 022002  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
1692
1693                                 ;IF NO PREVIOUS ERROR, TEST FOR ADJACENT BIT SETTING 'MOC''
1694 022006                          20$:
1695 022006  005703                          TST     R3
1696 022010  001046                          BNE     70$             ;BRANCH IF ANY PREVIOUS ERROR
1697 022012  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
1698
1699                                 ;WRITE THE TEST PATTERN, CHECK MOC USING PIP
1700 022016                          30$:
1701 022016  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1702 022024  010260  000024                  MOV     R2,RMMR1(R0)    ;LOAD RMMR1
1703 022030  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
1704 022036  042737  157777  001142          BIC     #^CPIP,$BDDAT   ;GET PIP STATUS
1705 022044  005037  001140                  CLR     $GDDAT          ;SETUP EXPECTED RESULT
1706 022050  032702  000400                  BIT     #MOC,R2
1707 022054  001003                          BNE     40$
1708 022056  052737  020000  001140          BIS     #PIP,$GDDAT     ;PIP SHOULD BE SET
1709 022064  023737  001140  001142  40$.    CMP     $GDDAT,$BDDAT
1710 022072  001403                          BEQ     50$             ;BRANCH IF PIP OK
1711 022074  010237  001174                  MOV     R2,$TMP0        ;SAVE TEST PATTERN
1712 022100  104123                          EMT     123
1713
1714                                 ;ADVANCE THE TEST PATTERN
1715 022102                          50$:
1716 022102  042702  000001                  BIC     #DMD,R2         ;DONT SHIFT DMD
1717 022106  001002                          BNE     60$
1718 022110  012702  000001                  MOV     #DMD,R2         ;DONT TRUNCATE TEST
1719 022114  006302                  60$:    ASL     R2              ;SHIFT BIT
1720 022116  001403                          BEQ     70$             ;EXIT IF DONE
1721 022120  052702  000001                  BIS     #DMD,R2         ;KEEP DMD ON
1722 022124  000734                          BR      30$             ;CONTINUE TEST
1723
```

```
1724 022126                          70$:                            ;END OF TEST
1725
1726                                 ;;*****************************************************************
                                     ;*TEST 35       EBL TEST
                                     ;;*****************************************************************
     022126                          TST35:
     022126  000004                          SCOPE                   ;SCOPE CALL
     022130  000240                          NOP
     022132  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     022136  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     022142  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     022146  012737  000035  001226          MOV     #35,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1727
1728 022154  005003                          CLR     R3              ;RESET ERROR FLAGS
1729 022156  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1730 022162  062737  000024  001136          ADD     #RMMR1,$BDADR
1731 022170  005037  001140                  CLR     $GDDAT          ;SETUP EXPECTED RESULT
1732
1733                                 ;CLEAR AND VERIFY THAT END OF BLOCK IS RESET
1734 022174  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1735 022200  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1736 022206  042737  157777  001142          BIC     #^CEBL,$BDDAT
1737 022214  001403                          BEQ     10$             ;BRANCH IF EBL IS RESET
1738 022216  104124                          EMT     124
1739 022220  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
1740
1741                                 ;SET AND RESET DIAGNOSTIC END OF BLOCK, CHECK FOR EBL S-A-1.
1742 022224                          10$:
1743 022224  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1744 022232  012760  020001  000024          MOV     #DMD!DEBL,RMMR1(R0)     ;LOAD RMMR1
1745 022240  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1746 022246  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1747 022254  042737  157777  001142          BIC     #^CEBL,$BDDAT
1748 022262  001403                          BEQ     20$             ;BRANCH IF EBL IS RESET
1749 022264  104125                          EMT     125
1750 022266  052703  000001                  BIS     #BIT0,R3        ;SET ERROR FLAG
1751
1752                                 ;RESET AND SET DIAGNOSTIC END OF BLOCK, CHECK FOR EBL S-A-0
1753 022272                          20$:
1754 022272  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1755 022300  012760  020001  000024          MOV     #DMD!DEBL,RMMR1(R0)     ;LOAD RMMR1
1756 022306  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1757 022314  042737  157777  001142          BIC     #^CEBL,$BDDAT
1758 022322  001006                          BNE     30$             ;BRANCH IF EBL IS SET
1759 022324  012737  020000  001140          MOV     #EBL,$GDDAT
1760 022332  104126                          EMT     126
1761 022334  052703  000002                  BIS     #BIT1,R3        ;SET ERROR FLAG
1762
1763                                 ;IF NO PREVIOUS ERRORS, TEST FOR ADJACENT BIT INTERFERENCE ON 'DEBL'.
1764 022340                          30$:
1765 022340  005703                          TST     R3
1766 022342  001042                          BNE     70$             ;BRANCH IF ANY ERROR
1767 022344  012702  000001                  MOV     #1,R2           ;INITIALIZE TEST PATTERN
1768
1769                                 ;WRITE, READ AND VERIFY THE TEST PATTERN IN R2
1770 022350                          40$:
```

```
1771 022350 012760 000001 000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1772 022356 010260 000024                 MOV     R2,RMMR1(R0)    ;LOAD RMMR1
1773 022362 016037 000024 001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
1774 022370 042737 157777 001142          BIC     #^CEBL,$BDDAT
1775 022376 010203                        MOV     R2,R3           ;GENERATE EXPECTED RESULT
1776 022400 042703 157777                 BIC     #^CEBL,R3
1777 022404 020337 001142                 CMP     R3,$BDDAT
1778 022410 001405                        BEQ     50$             ;BRANCH IF EBL IS OK
1779 022412 010337 001140                 MOV     R3,$GDDAT       ;SAVE EXPECTED RESULT
1780 022416 010237 001174                 MOV     R2,$TMP0        ;SAVE TEST PATTERN
1781 022422 104127                        EMT     127
1782
1783                                ;SHIFT TO NEXT BIT POSITION
1784 022424                         50$:
1785 022424 042702 000001                 BIC     #DMD,R2         ;DONT SHIFT DMD
1786 022430 001002                        BNE     60$
1787 022432 012702 000001                 MOV     #DMD,R2         ;DONT TRUNCATE DMD
1788 022436 006302                  60$:  ASL     R2              ;SHIFT TO NEXT BIT
1789 022440 001403                        BEQ     70$             ;EXIT IF DONE
1790 022442 052702 000001                 BIS     #DMD,R2         ;KEEP DMD ON
1791 022446 000740                        BR      40$             ;CONTINUE TEST
1792
1793 022450                         70$:                          ;END OF TEST
1794
1807
1808                                ;;*****************************************************************
                                    ;*TEST 36       LAST SECTOR, LAST TRACK TEST
                                    ;*
                                    ;*TRANSFER TEST PATTERN TO RMDA THEN VERIFY LAST SECTOR 'LS' AND LAST
                                    ;*SECTOR/TRACK 'LST' FOR EACH TRANSFER. THE TABLE BELOW LISTS THE VALUE
                                    ;*OF RMDA FOR WHICH LS AND LST ARE SET.
                                    ;*
                                    ;*                      18 BIT MODE     16 BIT MODE
                                    ;*              LS =    XXX035          XXX037
                                    ;*
                                    ;*RM02/03 -     LST=    002035          002037
                                    ;*RM05    -     LST=    011035          011037
                                    ;;*****************************************************************
     022450                         TST36:
     022450 000004                        SCOPE                   ;SCOPE CALL
     022452 000240                        NOP
     022454 012706 001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
     022460 013700 001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     022464 013701 001466                 MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     022470 012737 000036 001226          MOV     #36,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1809
1810 022476 005002                        CLR     R2              ;INITIALIZE TEST PATTERN
1811 022500 010037 001136                 MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1812 022504 062737 000024 001136          ADD     #RMMR1,$BDADR
1813 022512 005037 001444                 CLR     RMOFO           ;START IN 18 BIT MODE
1814 022516 013737 001334 023006          MOV     LSTRK,80$       ;SETUP LAST TRACK AND
1815 022524 112737 000035 023006          MOVB    #035,80$        ;LAST SECTOR (29.)
1816 022532                         10$:
1817 022532 004737 054674                 JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1818 022536 013760 001444 000032          MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
1819 022544 016037 000032 001176          MOV     RMOF(R0),$TMP1  ;STORE RMOF AT $TMP1
1820 022552 010260 000006                 MOV     R2,RMDA(R0)     ;LOAD RMDA
```

```
1821 022556 016037 000024 001362          MOV   RMMR1(R0),RMMR1I      ;STORE RMMR1 IN INPUT BUFFER
1822 022564 013737 001362 001142          MOV   RMMR1I,$BDDAT   ;VERIFY 'LS''
1823 022572 042737 177773 001142          BIC   #^CLS,$BDDAT
1824 022600 005037 001140                 CLR   $GDDAT          ;GENERATE EXPECTED 'LS''
1825 022604 032737 010000 001444          BIT   #FMT16,RMOFO    ;16 BIT MODE ?
1826 022612 001004                        BNE   20$             ;YES!!
1827 022614 123702 023006                 CMPB  80$,R2          ;18 BIT MODE LAST SECTOR ?
1828 022620 001007                        BNE   35$             ;NO !!
1829 022622 000403                        BR    30$
1830 022624 123702 023006          20$:   CMPB  80$,R2          ;16 BIT MODE LAST SECTOR ?
1831 022630 001003                        BNE   35$             ;NO !!
1832
1833 022632 052737 000004 001140   30$:   BIS   #LS,$GDDAT      ;LS SHOULD BE ON-16 BIT MODE
1834 022640 023737 001140 001142   35$:   CMP   $GDDAT,$BDDAT
1835 022646 001404                        BEQ   40$             ;BRANCH IF LS IS CORRECT
1836 022650 010237 001174                 MOV   R2,$TMP0        ;SAVE TEST PATTERN
1837 022654 104130                        EMT   130
1838 022656 000454                        BR    90$             ;SKIP TO NEXT
1839
1840 022660 013737 001362 001142   40$:   MOV   RMMR1I,$BDDAT   ;VERIFY 'LST''
1841 022666 042737 177775 001142          BIC   #^CLST,$BDDAT
1842 022674 005037 001140                 CLR   $GDDAT          ;GENERATE EXPECTED 'LST''
1843 022700 032737 010000 001444          BIT   #FMT16,RMOFO    ;16 BIT MODE??
1844 022706 001004                        BNE   50$             ;YES!!
1845 022710 023702 023006                 CMP   80$,R2          ;18 BIT MODE LAST TRACK/SECTOR ?
1846 022714 001007                        BNE   65$             ;NO !.
1847 022716 000403                        BR    60$
1848 022720 023702 023006          50$:   CMP   80$,R2          ;16 BIT MODE LAST TRACK/SECTOR ?
1849 022724 001003                        BNE   65$             ;NO !.
1850
1851 022726 052737 000002 001140   60$:   BIS   #LST,$GDDAT      ;LST SHOULD BE SET
1852 022734 023737 001140 001142   65$:   CMP   $GDDAT,$BDDAT
1853 022742 001404                        BEQ   70$
1854 022744 010237 001174                 MOV   R2,$TMP0        ;SAVE TEST PATTERN
1855 022750 104131                        EMT   131
1856 022752 000416                        BR    90$             ;SKIP TO NEXT
1857
1858                               ;ADVANCE TO NEXT TEST PATTERN, CHANGE TO 16 BIT MODE IF ALL
1859                               ;18 BIT TESTS DONE.
1860 022754                        70$:
1861 022754 005202                        INC   R2              ;INCREMENT PATTERN
1862 022756 001265                        BNE   10$             ;CONTINUE IF NOT DONE
1863 022760 032737 010000 001444          BIT   #FMT16,RMOFO    ;DONE 16 BIT TEST ?
1864 022766 001010                        BNE   90$             ;YES!!
1865 022770 012737 010000 001444          MOV   #FMT16,RMOFO    ;DO 16 BIT FORMAT TEST
1866 022776 112737 000037 023006          MOVB  #037,80$        ;SET LAST SECTOR FOR 16 BIT MODE (31.)
1867 023004 000652                        BR    10$
1868
1869 023006 000000                 80$:   .WORD 0               ;HOLDS LAST TRACK/SECTOR ADDRESS
1870
1871 023010                        90$:
1872
1876                               ;;*****************************************************************
                                   ;*TEST 37       RMDA COUNT TEST

                                   ;;*****************************************************************
     023010                        TST37:
```

```
;37      RMDA COUNT TEST

        023010  000004                          SCOPE                   ;SCOPE CALL
        023012  000240                          NOP
        023014  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        023020  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        023024  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        023030  012737  000037  001226          MOV     #37,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1877
1878    023036  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1879    023042  062737  000006  001136          ADD     #RMDA,$BDADR
1880    023050  005037  001444                  CLR     RMOFO           ;START WITH 18 BIT FORMAT
1881    023054  012737  000001  001140          MOV     #1,$GDDA1       ;SETUP FIRST COUNT
1882    023062  012737  000035  023462          MOV     #29.,110$       ;LAST SECTOR
1883
1884                                    ;INCREMENT SECTOR COUNT USING DIAGNOSTIC END OF BLOCK STARTING AT
1885                                    ;SECTOR 0 AND CONTINUING UNTIL TRACK ADDRESS INCREMENTS
1886                                    ;       .CLEAR THE MASSBUS
1887                                    ;       .SET FORMAT
1888                                    ;       .LOAD SECTOR AND TRACK ADDRESS
1889                                    ;       .ENABLE DEBUG CLOCK
1890                                    ;       .SET GO BIT
1891    023070                          10$:
1892    023070  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1893    023074  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1894    023102  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
1895    023110  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
1896    023116  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
1897    023124  013760  001444  000032          MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
1898    023132  016037  000032  001174          MOV     RMOF(R0),$TMP0  ;STORE RMOF AT $TMP0
1899    023140  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
1900    023146  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
1901
1902                                    ;SET AND RESET EBL TO INCREMENT RMDA THEN VERIFY RMDA.
1903    023154                          25$:
1904    023154  012760  060001  000024          MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR
1905    023162  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
1906    023170  016037  000006  001142          MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
1907    023176  023737  001142  001140          CMP     $BDDAT,$GDDAT
1908    023204  001402                          BEQ     30$             ;BRANCH IF RMDA OK
1909    023206  104132                          EMT     132
1910    023210  000416                          BR      50$             ;OUT OF SYNC-SKIP TO NEXT
1911
1912                                    ;ADVANCE EXPECTED SECTOR COUNT AND CONTINUE IF ONE CYCLE NOT
1913                                    ;COMPLETE
1914    023212                          30$:
1915    023212  005237  001140                  INC     $GDDAT          ;INCREMENT EXPECTED SECTOR
1916    023216  123737  001142  023462          CMPB    $BDDAT,110$     ;WAS THE LAST SECTOR JUST COUNTED??
1917    023224  001004                          BNE     40$             ;NO.!
1918    023226  105037  001140                  CLRB    $GDDAT          ;YES-NEXT SECTOR SHOULD BE ZERO
1919    023232  105237  001141                  INCB    $GDDAT+1        ;INCREMENT TRACK ADDRESS
1920    023236  105737  001142          40$:    TSTB    $BDDAT          ;HAS A FULL CYCLE BEEN COUNTED??
1921    023242  001401                          BEQ     50$             ;YES-DO NEXT
1922    023244  000743                          BR      25$             ;CONTINUE SECTOR TEST
1923
1924                                    ;INCREMENT TRACK COUNT USING DIAGNOSTIC END OF BLOCK.  START AT TRACK 0,
1925                                    ;LAST SECTOR AND COUNT ONE COMPLETE TRACK CYCLE.
1926    023246                          50$:
1927    023246  013737  023462  001420          MOV     110$,RMDAO      ;START SECTOR ADDRESS = 0
```

```
1928 023254  012737  000400  001140          MOV     #TA1,$GDDAT     ;FIRST VALUE AFTER INCREMENT
1929
1930                                 ;       .CLEAR THE MASSBUS
1931                                 ;       .SET FORMAT
1932                                 ;       .LOAD LAST SECTOR ADDRESS AND TEST TRACK ADDRESS
1933                                 ;       .ENABLE DEBUG CLOCK
1934                                 ;       .SET GO BIT
1935 023262                         60$:
1936 023262  004737  054674          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1937 023266  013760  001444  000032  MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
1938 023274  013760  001420  000006  MOV     RMDAO,RMDA(R0)  ;LOAD RMDA
1939 023302  012760  000001  000024  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
1940 023310  012760  040001  000024  MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
1941 023316  012760  000001  000000  MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
1942
1943                                 ;CLOCK RMDA USING DIAGNOSTIC END OF BLOCK
1944 023324  012760  060001  000024  MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
1945 023332  012760  040001  000024  MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
1946
1947                                 ;VERIFY RMDA ACCORDING TO $GDDAT
1948 023340  016037  000006  001142  MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
1949 023346  023737  001140  001142  CMP     $GDDAT,$BDDAT
1950 023354  001402                 BEQ     70$
1951 023356  104133                 EMT     133
1952 023360  000441                 BR      120$            ;OUT OF SYNC-SKIP TO NEXT
1953
1954                                 ;SETUP FOR NEXT INCREMENT OF RMDA TRACK ADDRESS
1955 023362                         70$:
1956 023362  105237  001141          INCB    $GDDAT+1        ;ADVANCE EXPECTED TRACK
1957 023366  123737  001143  001335  CMPB    $BDDAT+1,LSTRK+1        ;WAS THE LAST TRACK JUST COUNTED??
1958 023374  001002                 BNE     80$             ;NO!!
1959 023376  005037  001140          CLR     $GDDAT          ;YES-NEXT TRACK, SECTOR SHOULD BE ZERO
1960 023402  013737  001142  001420  80$:    MOV     $BDDAT,RMDAO    ;HAS A FULL CYCLE BEEN COUNTED??
1961 023410  001404                 BEQ     90$             ;YES!!
1962 023412  113737  023462  001420  MOVB    110$,RMDAO      ;INCREMENT FROM LAST SECTOR
1963 023420  000720                 BR      60$
1964 023422                         90$:
1965 023422  032737  010000  001444  BIT     #FMT16,RMOFO    ;DONE BOTH FORMATS??
1966 023430  001015                 BNE     120$            ;YES!.
1967 023432  012737  010000  001444  MOV     #FMT16,RMOFO    ;SET FORMAT BIT FOR 16
1968 023440  012737  000037  023462  MOV     #31.,110$       ;SET LAST SECTOR FOR 16 BIT MODE
1969 023446  000400                 BR      100$
1970
1971 023450  012737  000001  001140  100$:   MOV     #1,$GDDAT       ;SET FIRST COUNT VALUE
1972 023456  000137  023070          JMP     10$             ;REPEAT TEST
1973
1974 023462  000000                 110$:   .WORD   0               ;STORAGE FOR LAST SECTOR VALUE
1975
1976 023464                         120$:
1977
1978                                 ;;*****************************************************************
                                     ;*TEST 40       RMDC COUNT TEST
                                     ;;*****************************************************************
     023464                         TST40:
     023464  000004                 SCOPE                   ;SCOPE CALL
     023466  000240                 NOP
```

```
023470  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
023474  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
023500  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
023504  012737  000040  001226          MOV     #40,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
1979
1980                                ;CLEAR RMDC, SET FORMAT AND SETUP PROGRAM PARAMETERS
1981 023512  010037  001136          MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
1982 023516  062737  000034  001136  ADD     #RMDC,$BDADR
1983 023524  005037  001444          CLR     RMOFO           ;START WITH 18 BIT FORMAT
1984 023530  013737  001334  001420  MOV     LSTRK,RMDAO     ;SETUP LAST TRACK AND
1985 023536  112737  000035  001420  MOVB    #29.,RMDAO      ;LAST SECTOR
1986 023544                          10$:
1987 023544  004737  054674          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1988 023550  012737  000001  001140  MOV     #1,$GDDAT       ;LOAD FIRST INCREMENTAL VALUE
1989 023556  012760  000000  000034  MOV     #0,RMDC(R0)     ;LOAD RMDC
1990
1991                                ;       .CLEAR THE MASSBUS
1992                                ;       .SET OFFSET
1993                                ;       .LOAD LAST SECTOR AND TRACK ADDRESS
1994                                ;       .ENABLE DEBUG CLOCK
1995                                ;       .SET GO BIT
1996 023564                          20$:
1997 023564  004737  054674          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
1998 023570  013760  001444  000032  MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
1999 023576  016037  000032  001174  MOV     RMOF(R0),$TMP0  ;STORE RMOF AT $TMP0
2000 023604  013760  001420  000006  MOV     RMDAO,RMDA(R0)  ;LOAD RMDA
2001 023612  012760  000001  000024  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2002 023620  012760  040001  000024  MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
2003 023626  012760  000001  000000  MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
2004
2005                                ;CLOCK THE CYLINDER ADDRESS USING DEBL
2006 023634  012760  060001  000024  MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
2007 023642  012760  040001  000024  MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
2008 023650  016037  000034  001142  MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
2009 023656  023737  001140  001142  CMP     $GDDAT,$BDDAT
2010 023664  001402                  BEQ     30$             ;BRANCH IF RMDC=RMDC+1
2011 023666  104140                  EMT     140
2012 023670  000427                  BR      60$             ;OUT OF SYNC-SKIP TO END
2013
2014                                ;ADVANCE EXPECTED RESULT FOR NEXT INCREMENT
2015 023672                          30$:
2016 023672  005237  001140          INC     $GDDAT          ;ADVANCE NEXT RESULT
2017 023676  022737  002000  001140  CMP     #1024.,$GDDAT   ;SHOULD NEXT VALUE BE ZERO??
2018 023704  001002                  BNE     40$             ;NO!!
2019 023706  005037  001140          CLR     $GDDAT          ;YES-RMDC SHOULD OVERFLOW
2020 023712  005737  001142  40$:    TST     $BDDAT          ;IS ONE CYCLE COMPLETE??
2021 023716  001401                  BEQ     50$             ;YES!!
2022 023720  000721                  BR      20$             ;CONTINUE
2023 023722                          50$:
2024 023722  032737  010000  001444  BIT     #FMT16,RMOFO    ;DONE 16 BIT FORMAT MODE ?
2025 023730  001007                  BNE     60$             ;YES !.
2026 023732  012737  010000  001444  MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT AND
2027 023740  112737  000037  001420  MOVB    #31.,RMDAO      ;LOAD LAST SECTOR FOR 16 BIT MODE
2028 023746  000676                  BR      10$             ;REPEAT TEST
2029 023750                          60$:
2030
2031                                ;;*****************************************************************
                                                                                                    ;
```

```
                                ;*TEST 41       LBT TEST
                                ;;*****************************************************************
        023750                  TST41:
        023750  000004                  SCOPE                   ;SCOPE CALL
        023752  000240                  NOP
        023754  012706  001100          MOV     #STACK,SP       ;LOAD THE STACK POINTER
        023760  013700  001276          MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
        023764  013701  001466          MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        023770  012737  000041  001226  MOV     #41,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
   2032
   2033 023776  010037  001136          MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
   2034 024002  062737  000012  001136  ADD     #RMDS,$BDADR
   2035
   2036 024010  005037  001444          CLR     RMOFO           ;START WITH 18 BIT MODE
   2037 024014  013737  001334  001420  MOV     LSTRK,RMDAO     ;SETUP LAST TRACK AND
   2038 024022  112737  000035  001420  MOVB    #29.,RMDAO      ;LAST SECTOR
   2039
   2040                                ;       CLEAR THE MASSBUS
   2041                                ;       SET FORMAT
   2042                                ;       LOAD LAST TRACK AND SECTOR
   2043                                ;       LOAD LAST CYLINDER
   2044                                ;
   2045                                ;       VERIFY THAT 'LBT' IS RESET
   2046 024030                      10$:
   2047 024030  004737  054674          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
   2048 024034  012760  001466  000034  MOV     #822.,RMDC(R0)  ;LOAD RMDC
   2049 024042  013760  001420  000006  MOV     RMDAO,RMDA(R0)  ;LOAD RMDA
   2050 024050  013760  001444  000032  MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
   2051 024056  016037  000032  001174  MOV     RMOF(R0),$TMPO  ;STORE RMOF AT $TMPO
   2052 024064  016037  000012  001142  MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
   2053 024072  042737  175777  001142  BIC     #^CLBT,$BDDAT
   2054 024100  001403                  BEQ     20$             ;BRANCH IF LBT IS RESET
   2055 024102  005037  001140          CLR     $GDDAT          ;LBT SHOULD BE ZERO
   2056 024106  104141                  EMT     141
   2057
   2058                                ;       ENABLE DEBUG CLOCK
   2059                                ;       SET GO
   2060                                ;       FORCE EBL
   2061                                ;
   2062                                ;       VERIFY THAT LBT IS SET
   2063 024110                      20$:
   2064 024110  012760  000001  000024  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
   2065 024116  012760  000000  000014  MOV     #0,RMER1(R0)    ;LOAD RMER1
   2066 024124  012760  000000  000042  MOV     #0,RMER2(R0)    ;LOAD RMER2
   2067 024132  012760  000001  000000  MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
   2068 024140  012760  060001  000024  MOV     #DMD!DBEN!DEBL,RMMR1(R0)        ;LOAD RMMR1
   2069 024146  012760  040001  000024  MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
   2070 024154  016037  000012  001142  MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
   2071 024162  042737  175777  001142  BIC     #^CLBT,$BDDAT
   2072 024170  001005                  BNE     30$             ;BRANCH IF LBT IS SET
   2073 024172  012737  002000  001140  MOV     #LBT,$GDDAT
   2074 024200  104142                  EMT     142
   2075 024202  000413                  BR      40$
   2076 024204                      30$:
   2077 024204  032737  010000  001444  BIT     #FMT16,RMOFO    ;DONE 16 BIT FORMAT ?
   2078 024212  001007                  BNE     40$             ;YES !.
```

```
2079 024214  012737  010000  001444          MOV     #FMT16,RMOFO    ;SET 16 BIT MODE AND
2080 024222  112737  000037  001420          MOVB    #31.,RMDAO      ;LAST SECTOR
2081 024230  000677                          BR      10$             ;TEST AGAIN
2082 024232                              40$:
2083
2084                                     ;;*****************************************************************
                                         ;*TEST 42       COMPOSITE ERROR TEST
                                         ;;*****************************************************************
     024232                              TST42:
     024232  000004                              SCOPE                   ;SCOPE CALL
     024234  000240                              NOP
     024236  012706  001100                      MOV     #STACK,SP       ;LOAD THE STACK POINTER
     024242  013700  001276                      MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     024246  013701  001466                      MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     024252  012737  000042  001226              MOV     #42,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2085
2086 024260  004737  054674                      JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2087 024264  010037  001136                      MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
2088 024270  062737  000012  001136              ADD     #RMDS,$BDADR
2089
2090                                     ;USING DIAGNOSTIC MODE, CLEAR ALL ERRORS AND VERIFY THAT COMPOSITE
2091                                     ;ERROR IS RESET.
2092 024276  012760  000001  000024              MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2093 024304  012760  000000  000014              MOV     #0,RMER1(R0)    ;LOAD RMER1
2094 024312  012760  000000  000042              MOV     #0,RMER2(R0)    ;LOAD RMER2
2095 024320  016037  000012  001142              MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2096 024326  042737  137777  001142              BIC     #^CERR,$BDDAT
2097 024334  001403                              BEQ     10$             ;BRANCH IF ERR IS RESET
2098 024336  005037  001140                      CLR     $GDDAT
2099 024342  104143                              EMT     143
2100 024344  012737  040000  001140  10$:       MOV     #ERR,$GDDAT
2101
2102                                     ;SET BOTH ERROR REGISTERS AND VERIFY THAT COMPOSITE ERROR IS SET
2103 024352  012760  177777  000014              MOV     #-1,RMER1(R0)   ;LOAD RMER1
2104 024360  012760  177777  000042              MOV     #-1,RMER2(R0)   ;LOAD RMER2
2105 024366  016037  000012  001142              MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2106 024374  042737  137777  001142              BIC     #^CERR,$BDDAT
2107 024402  001001                              BNE     20$             ;BRANCH IF ERR IS SET
2108 024404  104144                              EMT     144
2109
2110                                     ;VERIFY THAT COMPOSITE ERROR SETS FOR EACH BIT OF RMER1
2111 024406                              20$:
2112 024406  012702  000001                      MOV     #1,R2           ;INITIALIZE TEST PATTERN
2113
2114                                     ;WRITE THE TEST PATTERN AND VERIFY THAT ERR IS SET
2115 024412                              30$:
2116 024412  004737  054674                      JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2117 024416  012760  000001  000024              MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2118 024424  012760  000000  000014              MOV     #0,RMER1(R0)    ;LOAD RMER1
2119 024432  012760  000000  000042              MOV     #0,RMER2(R0)    ;LOAD RMER2
2120 024440  010260  000014                      MOV     R2,RMER1(R0)    ;LOAD RMER1
2121 024444  016037  000012  001142              MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2122 024452  042737  137777  001142              BIC     #^CERR,$BDDAT
2123 024460  001005                              BNE     40$             ;BRANCH IF COMPOSITE ERROR SET
2124 024462  010237  001174                      MOV     R2,$TMP0        ;SAVE RMER1 TEST PATTERN
2125 024466  005037  001176                      CLR     $TMP1           ;SAVE RMER2 TEST PATTERN
```

```
2126 024472  104145                          EMT     145
2127
2128                                 ;ADVANCE THE TEST PATTERN FOR RMER1
2129 024474                          40$:
2130 024474  006302                          ASL     R2
2131 024476  001345                          BNE     30$             ;CONTINUE IF TEST NOT DONE
2132
2133                                 ;VERIFY THAT COMPOSITE ERROR SETS FOR EACH BIT OF RMER2
2134 024500                          50$:
2135 024500  012702  000001                    MOV     #1,R2           ;INITIALIZE TEST PATTERN
2136 024504  012737  010000  001444            MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT
2137
2138                                 ;WRITE THE TEST PATTERN AND VERIFY THAT ERR IS SET
2139 024512                          60$:
2140 024512  004737  054674                    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2141 024516  012760  000001  000024            MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2142 024524  012760  000000  000014            MOV     #0,RMER1(R0)    ;LOAD RMER1
2143 024532  012760  000000  000042            MOV     #0,RMER2(R0)    ;LOAD RMER2
2144 024540  010260  000042                    MOV     R2,RMER2(R0)    ;LOAD RMER2
2145 024544  013760  001444  000032            MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
2146 024552  016037  000012  001142            MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2147 024560  042737  137777  001142            BIC     #^CERR,$BDDAT
2148 024566  012737  040000  001140            MOV     #ERR,$GDDAT     ;SETUP EXPECTED VALUE FOR COMP ERROR
2149 024574  032702  001567                    BIT     #XNUER2,R2
2150 024600  001402                          BEQ     65$             ;BRANCH IF TEST BIT IS A USED BIT
2151 024602  005037  001140                    CLR     $GDDAT          ;TEST BIT IS NOT USED - ERR SHOULD BE 0
2152 024606  023737  001140  001142  65$:     CMP     $GDDAT,$BDDAT
2153 024614  001405                          BEQ     70$             ;BRANCH IF COMP ERROR IS OK
2154 024616  005037  001174                    CLR     $TMP0           ;SAVE RMER1 TEST PATTERN
2155 024622  010237  001176                    MOV     R2,$TMP1        ;SAVE RMER2 TEST PATTERN
2156 024626  104145                          EMT     145
2157
2158                                 ;ADVANCE THE TEST PATTERN FOR RMER2
2159 024630                          70$:
2160 024630  006302                          ASL     R2
2161 024632  001327                          BNE     60$             ;CONTINUE IF TEST NOT DONE
2162 024634                          80$:
2163
2164                                 ;;*****************************************************************
                                     ;*TEST 43        WRITE GO TEST

                                     ;;*****************************************************************
     024634                          TST43:
     024634  000004                          SCOPE                   ;SCOPE CALL
     024636  000240                          NOP
     024640  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     024644  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     024650  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     024654  012737  000043  001226          MOV     #43,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2165
2166 024662  010037  001136                  MOV     R0,$BDADR       ;COPY RMCS1 ADDRESS
2167 024666  005002                          CLR     R2              ;INITIALIZE FUNCTION CODE
2168
2169                                 ;CLEAR THE MASSBUS, SET DIAGNOSTIC MODE AND ENABLE DEBUG CLOCK
2170 024670                          10$:
2171 024670  004737  054674                    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2172 024674  012760  000001  000024            MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
```

```
2173 024702  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
2174 024710  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
2175 024716  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
2176
2177                                 ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1, VERIFY GO IS SET
2178 024724  010203                          MOV     R2,R3           ;SETUP FUNCTION CODE
2179 024726  052703  000001                  BIS     #GO,R3
2180 024732  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2181 024736  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
2182 024744  032737  000001  001142          BIT     #GO,$BDDAT
2183 024752  001007                          BNE     20$             ;BRANCH IF GO IS SET
2184
2185                                 ;REPORT THE ERROR-CANT SET GO WITH THIS FUNCTION CODE
2186 024754  042737  177700  001142          BIC     #^CFNCMSK,$BDDAT
2187 024762  010337  001140                  MOV     R3,$GDDAT       ;SAVE FUNCTION CODE
2188 024766  104146                          EMT     146
2189 024770  000405                          BR      30$
2190
2191                                 ;ADVANCE R2 TO THE NEXT FUNCTION CODE
2192 024772                          20$:
2193 024772  062702  000002                  ADD     #2,R2
2194 024776  022702  000076                  CMP     #ILF76,R2
2195 025002  103332                          BHIS    10$
2196
2197 025004                          30$:                            ;END OF TEST
2198
2199                                 ;;*****************************************************************
                                     ;*TEST 44        BRANCH MULTIP_EXOR TEST

                                     ;;*****************************************************************
     025004                          TST44:
     025004  000004                          SCOPE                   ;SCOPE CALL
     025006  000240                          NOP
     025010  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     025014  013700  001276                  MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
     025020  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     025024  012737  000044  001226          MOV     #44,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2200
2201 025032  010037  001136                  MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
2202 025036  062737  000040  001136          ADD     #RMMR2,$BDADR
2203 025044  012702  025270                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
2204
2205                                 ;CLEAR THE MASSBUS AND SET DEBUG CLOCK ENABLE
2206 025050                          10$:
2207 025050  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2208 025054  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2209 025062  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
2210 025070  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
2211 025076  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
2212
2213                                 ;THE TEST BIT SHOULD BE ONE BECAUSE THE ADDRESS IS ALL ONES WHEN
2214                                 ;THE COMMAND SEQUENCER IS INITIALIZED.
2215 025104  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
2216 025112  032737  010000  001142          BIT     #TST,$BDDAT
2217 025120  001010                          BNE     15$             ;BRANCH IF TEST BIT IS ON
2218 025122  042737  167777  001142          BIC     #^CTST,$BDDAT   ;SETUP FOR ERROR TYPE
2219 025130  012737  010000  001140          MOV     #TST,$GDDAT
```

```
2220 025136  104147                          EMT     147
2221 025140  000452                          BR      40$             ;SKIP REST OF TEST
2222
2223                                 ;GET THE FUNCTION CODE FROM THE TABLE AND TRANSFER IT TO THE DEVICE,
2224                                 ;THEN STEP THE COMMAND SEQUENCER ACCORDING TO THE TABLE.
2225 025142                         15$:
2226 025142  111203                          MOVB    (R2),R3
2227 025144  052703  000001                 BIS     #GO,R3
2228 025150  042703  177700                 BIC     #^CFNCMSK,R3    ;R3=FUNCTION CODE, GO BIT
2229 025154  010360  000000                 MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2230 025160  010337  001174                 MOV     R3,$TMP0        ;SAVE R3 FOR ERROR MSG
2231
2232 025164  116203  000001                 MOVB    1(R2),R3        ;GET CLOCK COUNT IN R3
2233 025170  042703  177400                 BIC     #^C377,R3
2234 025174                          20$:
2235 025174  012760  141001  000024         MOV     #DMD!DBEN!MUR.DBCK,RMMR1(R0)    ;LOAD RMMR1
2236 025202  012760  041001  000024         MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
2237 025210  005303                         DEC     R3              ;DECREMENT CLOCK COUNT
2238 025212  001370                         BNE     20$             ;ISSUE CLOCKS TILL ZERO
2239
2240                                 ;GET THE TEST BIT AND COMPARE IT WITH THE TABLE ENTRY
2241 025214  016037  000040  001142         MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
2242 025222  042737  167777  001142         BIC     #^CTST,$BDDAT
2243 025230  016237  000002  001140         MOV     2(R2),$GDDAT
2244 025236  023737  001140  001142         CMP     $GDDAT,$BDDAT
2245 025244  001402                         BEQ     30$             ;BRANCH IF TEST BIT OK
2246 025246  104150                         EMT     150
2247 025250  000406                         BR      40$             ;SKIP REST OF TEST
2248
2249                                 ;MOVE THE TABLE POINTER AND CONTINUE IF NEXT ENTRY POSITIVE
2250 025252                          30$:
2251 025252  062702  000004                 ADD     #4,R2
2252 025256  105762  000001                 TSTB    1(R2)
2253 025262  100401                         BMI     40$             ;BRANCH IF DONE TEST
2254 025264  000671                         BR      10$             ;REPEAT TEST
2255 025266  000436                  40$:   BR      200$            ;JUMP OVER TABLE
2256
2257                                 ;TABLE OF FUNCTION CODES, CLOCK COUNTS, AND TEST BITS
2258 025270                          100$:
2259 025270     000                         .BYTE   NOP             ;MUX ADDRESS=DATA COMMAND
2260 025271     001                         .BYTE   1
2261 025272  000000                         .WORD   0               ;TEST BIT 0
2262
2263 025274     000                         .BYTE   NOP             ;MUX ADDRESS=UNIT READY
2264 025275     002                         .BYTE   2
2265 025276  000000                         .WORD   0               ;TEST BIT=0
2266
2267 025300     010                         .BYTE   DRVCLR          ;MUX ADDRESS=F4
2268 025301     001                         .BYTE   1
2269 025302  010000                         .WORD   TST             ;TEST BIT=1
2270
2271 025304     050                         .BYTE   WCD             ;MUX ADDRESS-F4
2272 025305     001                         .BYTE   1
2273 025306  000000                         .WORD   0               ;TEST BIT=0
2274
2275 025310     012                         .BYTE   RLEASE          ;MUX ADDRESS-F4
2276 025311     001                         .BYTE   1
```

```
2277 025312  010000                        .WORD   TST             ;TEST BIT=1
2278
2279 025314      052                       .BYTE   WCH             ;MUX ADDRESS=F4
2280 025315      001                       .BYTE   1
2281 025316  000000                        .WORD   0               ;TEST BIT=0
2282
2283 025320      020                       .BYTE   RIP             ;MUX ADDRESS=F4
2284 025321      001                       .BYTE   1
2285 025322  010000                        .WORD   TST             ;TEST BIT=1
2286
2287 025324      060                       .BYTE   WD              ;MUX ADDRESS=F4
2288 025325      001                       .BYTE   1
2289 025326  000000                        .WORD   0               ;TEST BIT=0
2290
2291 025330      022                       .BYTE   PAKACK          ;MUX ADDRESS=F4
2292 025331      001                       .BYTE   1
2293 025332  010000                        .WORD   TST             ;TEST BIT=1
2294
2295 025334      062                       .BYTE   WH              ;MUX ADDRESS=F4
2296 025335      001                       .BYTE   1
2297 025336  000000                        .WORD   0               ;TEST BIT=0
2298
2299 025340      030                       .BYTE   SEARCH          ;MUX ADDRESS=F4
2300 025341      001                       .BYTE   1
2301 025342  010000                        .WORD   TST             ;TEST BIT=1
2302
2303 025344      070                       .BYTE   RD              ;MUX ADDRESS=F4
2304 025345      001                       .BYTE   1
2305 025346  000000                        .WORD   0               ;TEST BIT=0
2306
2307 025350      032                       .BYTE   ILF32           ;MUX ADDRESS=F4
2308 025351      001                       .BYTE   1
2309 025352  010000                        .WORD   TST             ;TEST BIT=1
2310
2311 025354      072                       .BYTE   RH              ;MUX ADDRESS=F4
2312 025355      001                       .BYTE   1
2313 025356  000000                        .WORD   0               ;TEST BIT=0
2314
2315 025360      000                       .BYTE                   ;END OF TABLE
2316 025361      377                       .BYTE   -1
2317 025362  000000                        .WORD
2318 025364                          200$:                          ;END OF TEST
2319
2320
                                    ;;*******************************************************************
                                    ;*TEST 45        SET/RESET GO TEST
                                    ;;*******************************************************************
     025364                         TST45:
     025364  000004                        SCOPE                   ;SCOPE CALL
     025366  000240                        NOP
     025370  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     025374  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     025400  013701  001466                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     025404  012737  000045  001226        MOV     #45,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2321
2322 025412  012702  026026                MOV     #200$,R2        ;INITIALIZE FUNCTION CODE POINTER
2323
```

```
2324                                    ;CLEAR, THEN SET DIAGNOSTIC MODE, CLEAR COMPOSITE ERROR, SET MEDIUM
2325                                    ;ON LINE AND ENABLE DEBUG CLOCK
2326 025416                            10$:
2327 025416  004737  054674                    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2328 025422  012760  000001  000024            MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2329 025430  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2330 025436  012760  000000  000014            MOV     #0,RMER1(R0)    ;LOAD RMER1
2331 025444  012760  000000  000042            MOV     #0,RMER2(R0)    ;LOAD RMER2
2332
2333                                    ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1 AND VERIFY GO IS SET
2334 025452  111203                            MOVB    (R2),R3         ;GET FUNCTION CODE
2335 025454  042703  177701                    BIC     #^C1F76,R3      ;CLEAR UNUSED BITS
2336 025460  052703  000001                    BIS     #GO,R3          ;SET GO
2337 025464  010360  000000                    MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2338 025470  016037  000000  001142            MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
2339 025476  032737  000001  001142            BIT     #GO,$BDDAT
2340 025504  001011                            BNE     20$             ;BRANCH IF GO IS SET
2341 025506  042737  177700  001142            BIC     #^CFNCMSK,$BDDAT
2342 025514  010337  001140                    MOV     R3,$GDDAT       ;SAVE EXPECTED RESULT
2343 025520  010037  001136                    MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
2344 025524  104151                            EMT     151
2345 025526  000536                            BR      100$
2346
2347                                    ;GET READY STATUS AND VERIFY THAT IT IS THE COMPLEMENT OF GO
2348 025530                            20$:
2349 025530  005037  001140                    CLR     $GDDAT          ;EXPECT DRY TO BE OFF
2350 025534  032737  000001  001142            BIT     #GO,$BDDAT      ;WAS GO SET??
2351 025542  001003                            BNE     30$             ;YES.!
2352 025544  012737  000200  001140            MOV     #DRY,$GDDAT     ;GO WAS NOT SET, DRY SHOULD BE
2353 025552                            30$:
     025552  016037  000012  001142            MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2354 025560  042737  177577  001142            BIC     #^CDRY,$BDDAT
2355 025566  023737  001140  001142            CMP     $GDDAT,$BDDAT
2356 025574  001406                            BEQ     40$             ;BRANCH IF DRY IS OK
2357 025576  010037  001136                    MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
2358 025602  062737  000012  001136            ADD     #RMDS,$BDADR
2359 025610  104152                            EMT     152
2360
2361                                    ;STEP THE DEBUG CLOCK AND VERIFY THAT GO REMAINS SET
2362 025612                            40$:
2363 025612  116204  000001                    MOVB    1(R2),R4        ;GET NUMBER OF CLOCK CYCLES
2364 025616  042704  177400                    BIC     #^C377,R4
2365 025622                            50$:
2366 025622  012760  141001  000024            MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
2367 025630  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2368 025636  016037  000000  001142            MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
2369 025644  042737  177700  001142            BIC     #^CFNCMSK,$BDDAT  ;CLEAR UNUSED BITS
2370 025652  010037  001136                    MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
2371 025656  010337  001140                    MOV     R3,$GDDAT       ;SAVE EXPECTED RESULT
2372
2373                                    ;DECREMENT CLOCK COUNT AND EXIT LOOP IF ZERO
2374 025662  005304                            DEC     R4
2375 025664  001406                            BEQ     60$
2376 025666  032737  000001  001142            BIT     #GO,$BDDAT      ;IS GO STILL SET??
2377 025674  001352                            BNE     50$             ;YES!!
2378 025676  104153                            EMT     153
2379 025700  000451                            BR      100$            ;OUT OF SYNC-SKIP TO NEXT
```

```
2380
2381                                        ;GO SHOULD NOW BE RESET AND DRY SHOULD BE SET
2382 025702                                 60$:
2383 025702  032737  000001  001142                BIT     #GO,$BDDAT      ;IS GO RESET??
2384 025710  001405                                BEQ     70$             ;YES..
2385 025712  042737  000001  001140                BIC     #GO,$GDDAT      ;SETUP EXPECTED RESULT
2386 025720  104154                                EMT     154
2387 025722  000440                                BR      100$
2388 025724                                 70$:
2389 025724  012737  000200  001140                MOV     #DRY,$GDDAT     ;EXPECT DRIVE READY TO BE SET
2390 025732  032737  000001  001142                BIT     #GO,$BDDAT      ;DID GO RESET??
2391 025740  001402                                BEQ     80$             ;YES!!
2392 025742  005037  001140                        CLR     $GDDAT          ;GO IS SET-
2393 025746                                 80$:
     025746  016037  000012  001142                MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2394 025754  042737  177577  001142                BIC     #^CDRY,$BDDAT
2395 025762  010037  001136                        MOV     R0,$BDADR       ;COPY REGISTER ADDRESS
2396 025766  062737  000012  001136                ADD     #RMDS,$BDADR
2397 025774  023737  001140  001142                CMP     $GDDAT,$BDDAT   ;IS DRIVE READY OK??
2398 026002  001401                                BEQ     90$             ;YES!!
2399 026004  104152                                EMT     152
2400
2401                                        ;ADVANCE TO THE NEXT FUNCTION CODE TO BE TESTED-EXIT IF DONE
2402 026006                                 90$:
2403 026006  062702  000002                        ADD     #2,R2           ;MOVE TABLE POINTER
2404 026012  105762  000001                        TSTB    1(R2)           ;END OF TABLE??
2405 026016  100402                                BMI     100$            ;YES!!
2406 026020  000137  025416                        JMP     10$             ;TEST THIS FUNCTION CODE
2407 026024  000423                         100$:  BR      300$            ;GO TO NEXT TEST
2408
2409                                        ;TABLE OF FUNCTION CODES AND CLOCK COUNTS USED DURING TEST
2410 026026                                 200$:
2411 026026      002                               .BYTE   ILF02           ;ILLEGAL FUNCTION CODE #2
2412 026027      001                               .BYTE   1
2413
2414 026030      004                               .BYTE   SEEK            ;SEEK COMMAND
2415 026031      001                               .BYTE   1
2416
2417 026032      006                               .BYTE   RECAL           ;RECALIBRATE COMMAND
2418 026033      001                               .BYTE   1
2419
2420 026034      014                               .BYTE   OFFSET          ;OFFSET COMMAND
2421 026035      001                               .BYTE   1
2422
2423 026036      016                               .BYTE   RTC             ;RETURN TO CENTER LINE COMMAND
2424 026037      001                               .BYTE   1
2425
2426 026040      024                               .BYTE   ILF24           ;ILLEGAL FUNCTION CODE #24
2427 026041      001                               .BYTE   1
2428
2429 026042      026                               .BYTE   ILF26           ;ILLEGAL FUNCTION CODE #26
2430 026043      001                               .BYTE   1
2431
2432 026044      034                               .BYTE   ILF34           ;ILLEGAL FUNCTION CODE #34
2433 026045      001                               .BYTE   1
2434
2435 026046      036                               .BYTE   ILF36           ;ILLEGAL FUNCTION CODE #36
```

```
2436 026047  001                        .BYTE   1
2437
2438 026050  042                        .BYTE   ILF42           ;ILLEGAL FUNCTION CODE #42
2439 026051  001                        .BYTE   1
2440
2441 026052  044                        .BYTE   ILF44           ;ILLEGAL FUNCTION CODE #44
2442 026053  001                        .BYTE   1
2443
2444 026054  046                        .BYTE   ILF46           ;ILLEGAL FUNCTION CODE #46
2445 026055  001                        .BYTE   1
2446
2447 026056  054                        .BYTE   ILF54           ;ILLEGAL FUNCTION CODE #54
2448 026057  001                        .BYTE   1
2449
2450 026060  056                        .BYTE   ILF56           ;ILLEGAL FUNCTION CODE #56
2451 026061  001                        .BYTE   1
2452
2453 026062  064                        .BYTE   ILF64           ;ILLEGAL FUNCTION CODE #64
2454 026063  001                        .BYTE   1
2455
2456 026064  066                        .BYTE   ILF66           ;ILLEGAL FUNCTION CODE #66
2457 026065  001                        .BYTE   1
2458
2459 026066  074                        .BYTE   ILF74           ;ILLEGAL FUNCTION CODE #74
2460 026067  001                        .BYTE   1
2461
2462 026070  076                        .BYTE   ILF76           ;ILLEGAL FUNCTION CODE #76
2463 026071  001                        .BYTE   1
2464
2465 026072  000                        .BYTE                   ;END OF TABLE
2466 026073  377                        .BYTE   -1
2467
2468 026074                     300$:                           ;END OF TEST
2469
2470                             ;;*******************************************************************
                                ;*TEST 46        END 1 RESET GO TEST
                                ;;*******************************************************************
     026074                     TST46:
     026074  000004                     SCOPE                   ;SCOPE CALL
     026076  000240                     NOP
     026100  012706  001100             MOV     #STACK,SP       ;LOAD THE STACK POINTER
     026104  013700  001276             MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     026110  013701  001466             MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     026114  012737  000046  001226     MOV     #46,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2471
2472 026122  012702  026360             MOV     #100$,R2        ;INITIALIZE TABLE POINTER
2473 026126  010037  001136             MOV     R0,$BDADR       ;COPY RMCS1 ADDRESS
2474
2475                             ;CLEAR MASSBUS, THEN SET MEDIUM ON LINE AND ENABLE DEBUG CLOCK
2476 026132                     10$:
2477 026132  004737  054674             JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2478 026136  012760  000001  000024     MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2479 026144  012760  041001  000024     MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2480 026152  012760  000000  000014     MOV     #0,RMER1(R0)    ;LOAD RMER1
2481 026160  012760  000000  000042     MOV     #0,RMER2(R0)    ;LOAD RMER2
2482
```

```
2483                                        ;TRANSFER THE FUNCTION CODE AND GO BIT TO RMCS1, VERIFY GO IS SET
2484 026166  111203                                MOVB    (R2),R3         ;GET FUNCTION CODE FROM
2485 026170  042703  177701                        BIC     #^C1F76,R3      ;TABLE AND SET GO
2486 026174  052703  000001                        BIS     #GO,R3
2487 026200  010337  001140                        MOV     R3,$GDDAT       ;SAVE FUNCTION CODE FOR MSG
2488 026204  010360  000000                        MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2489 026210  016037  000000  001142                MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
2490 026216  032737  000001  001142                BIT     #GO,$BDDAT
2491 026224  001005                                BNE     20$             ;BRANCH IF GO IS SET
2492 026226  042737  177700  001142                BIC     #^CFNCMSK,$BDDAT
2493 026234  104151                                EMT     151
2494 026236  000447                                BR      60$             ;OUT OF SYNC-SKIP
2495
2496                                        ;GET THE NUMBER OF CLOCK CYCLES FROM THE TABLE, SAVE EXPECTED STATUS
2497 026240                                 20$:
2498 026240  116204  000001                        MOVB    1(R2),R4        ;R4=CLOCK COUNT
2499 026244  042704  177400                        BIC     #^C377,R4
2500
2501                                        ;STEP THE DEBUG CLOCK AND VERIFY GO STATUS ON UNTIL CLOCK COUNT EXPIRES.
2502 026250                                 30$:
2503 026250  012760  141001  000024                MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
2504 026256  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2505 026264  016037  000000  001142                MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
2506 026272  042737  177700  001142                BIC     #^CFNCMSK,$BDDAT
2507 026300  005304                                DEC     R4
2508 026302  001406                                BEQ     40$             ;BRANCH IF GO SHOULD BE OFF
2509 026304  032737  000001  001142                BIT     #GO,$BDDAT
2510 026312  001356                                BNE     30$             ;CONTINUE IF GO IS ON
2511 026314  104153                                EMT     153
2512 026316  000417                                BR      60$             ;OUT OF SYNC-SKIP
2513
2514                                        ;VERIFY THAT GO RESET AT END1
2515 026320                                 40$:
2516 026320  032737  000001  001142                BIT     #GO,$BDDAT      ;DID GO RESET??
2517 026326  001405                                BEQ     50$             ;YES!!
2518 026330  042737  000001  001140                BIC     #GO,$GDDAT
2519 026336  104154                                EMT     154
2520 026340  000406                                BR      60$
2521
2522                                        ;GET THE NEXT FUNCTION CODE FROM THE TABLE
2523 026342                                 50$:
2524 026342  062702  000002                        ADD     #2,R2
2525 026346  105762  000001                        TSTB    1(R2)
2526 026352  100401                                BMI     60$             ;BRANCH IF END OF TABLE
2527 026354  000666                                BR      10$             ;TEST THIS FUNCTION CODE
2528 026356  000404                         60$:   BR      200$            ;JUMP OVER TABLE
2529
2530                                        ;TABLE OF FUNCTION CODES AND CLOCK COUNTS USED DURING TEST
2531 026360                                 100$:
2532 026360     012                                .BYTE   RLEASE          ;RELEASE COMMAND
2533 026361     002                                .BYTE   2
2534
2535 026362     030                                .BYTE   SEARCH          ;SEARCH COMMAND
2536 026363     002                                .BYTE   2
2537
2538 026364     032                                .BYTE   ILF32           ;ILLEGAL FUNCTON #32
2539 026365     002                                .BYTE   2
```

```
2540
2541 026366  000                        .BYTE                   ;END OF TABLE
2542 026367  377                        .BYTE   -1
2543 026370                     200$:                           ;END OF TEST
2544
2545                            ;;*******************************************************************
                                ;*TEST 47        SET PULSE TEST
                                ;;*******************************************************************
     026370                     TST47:
     026370  000004                     SCOPE                   ;SCOPE CALL
     026372  000240                     NOP
     026374  012706  001100             MOV     #STACK,SP       ;LOAD THE STACK POINTER
     026400  013700  001276             MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     026404  013701  001466             MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
     026410  012737  000047  001226     MOV     #47,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2546
2547 026416  010037  001136             MOV     R0,$BDADR       ;COPY REG ADDRESS FOR MSG
2548 026422  062737  000024  001136     ADD     #RMMR1,$BDADR
2549 026430  012702  026666             MOV     #100$,R2        ;INITIALIZE TABLE POINTER
2550
2551                            ;CLEAR THE MASS BUS, ENABLE DEBUG CLOCK, AND RESET ERROR REGISTERS
2552 026434                     10$:
2553 026434  004737  054674             JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2554 026440  012760  000001  000024     MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2555 026446  012760  041001  000024     MOV     #DMD!DBEN.MUR,RMMR1(R0) ;LOAD RMMR1
2556 026454  012760  000000  000014     MOV     #0,RMER1(R0)    ;LOAD RMER1
2557 026462  012760  000000  000042     MOV     #0,RMER2(R0)    ;LOAD RMER2
2558
2559                            ;VERIFY THAT CONTINUE, ''CONT'' IS RESET AFTER CLEAR
2560 026470  016037  000024  001142     MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
2561 026476  042737  177677  001142     BIC     #^CCONT,$BDDAT
2562 026504  001404                     BEQ     20$             ;BRANCH IF CONT WAS CLEARED
2563 026506  005037  001140             CLR     $GDDAT          ;FOR ERROR MSG
2564 026512  104155                     EMT     155
2565 026514  000463                     BR      70$
2566
2567                            ;GET THE FUNCTION CODE FROM THE TABLE AND TRANSFER IT TO RMCS1
2568 026516                     20$:
2569 026516  111203                     MOVB    (R2),R3
2570 026520  052703  000001             BIS     #GO,R3
2571 026524  042703  177700             BIC     #^CFNCMSK,R3    ;R3=FUNCTION CODE AND GO
2572 026530  010360  000000             MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2573 026534  010337  001174             MOV     R3,$TMP0        ;SAVE FUNCTION CODE FOR MSG
2574
2575                            ;GET THE CLOCK COUNT FROM THE TABLE
2576 026540  116203  000001             MOVB    1(R2),R3
2577 026544  042703  177400             BIC     #^C377,R3
2578
2579                            ;GET THE BIT STREAM FOR CONTINUE FROM THE TABLE
2580 026550  016204  000002             MOV     2(R2),R4
2581
2582                            ;STEP THE COMMAND SEQUENCER AND VERIFY CONTINUE STATUS
2583 026554                     30$:
2584 026554  012760  141001  000024     MOV     #DMD!DBEN!MUR.DBCK,RMMR1(R0)    ;LOAD RMMR1
2585 026562  012760  041001  000024     MOV     #DMD.DBEN.MUR,RMMR1(R0) ;LOAD RMMR1
2586 026570  016037  000024  001142     MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
```

```
2587 026576  042737  177677  001142          BIC     #^CCONT,$BDDAT
2588 026604  005037  001140                  CLR     $GDDAT          ;GENERATE EXPECTED CONTINUE
2589 026610  032704  000001                  BIT     #BIT0,R4
2590 026614  001403                          BEQ     40$
2591 026616  012737  000100  001140          MOV     #CONT,$GDDAT
2592 026624  023737  001140  001142  40$:    CMP     $GDDAT,$BDDAT
2593 026632  001402                          BEQ     50$             ;BRANCH IF CONTINUE IS OK
2594 026634  104156                          EMT     156
2595 026636  000412                          BR      70$             ;SKIP
2596
2597                                 ;DECREMENT CLOCK COUNT AND SHIFT BIT STREAM
2598 026640                          50$:
2599 026640  005303                          DEC     R3
2600 026642  001402                          BEQ     60$             ;BRANCH IF CLOCK COUNT EXPIRED
2601 026644  006204                          ASR     R4              ;SHIFT TO NEXT CONTINUE BIT
2602 026646  000742                          BR      30$             ;TEST NEXT CLOCK CYCLE
2603
2604                                 ;ADVANCE TABLE POINTER-EXIT IF DONE
2605 026650                          60$:
2606 026650  062702  000004                  ADD     #4,R2
2607 026654  105762  000001                  TSTB    1(R2)
2608 026660  100401                          BMI     70$             ;EXIT IF CLOCK COUNT NEGATIVE
2609 026662  000664                          BR      10$             ;CONTINUE TEST
2610 026664  000442                  70$:    BR      200$            ;JUMP OVER TABLE
2611
2612                                 ;TABLE OF FUNCTION CODES, CLOCK COUNTS AND CONTINUE BITS FOR TEST
2613 026666                          100$:
2614 026666     000                          .BYTE   NOP             ;NOP COMMAND
2615 026667     004                          .BYTE   4               ;4 CLOCKS
2616 026670  000000                          .WORD   ^B0000          ;CONTINUE-0000
2617
2618 026672     002                          .BYTE   ILF02           ;ILLEGAL FUNCTION 2
2619 026673     002                          .BYTE   2
2620 026674  000000                          .WORD   ^B00
2621
2622 026676     004                          .BYTE   SEEK            ;SEEK COMMAND
2623 026677     002                          .BYTE   2
2624 026700  000000                          .WORD   ^B00
2625
2626 026702     006                          .BYTE   RECAL           ;RECALIBRATE COMMAND
2627 026703     002                          .BYTE   2
2628 026704  000000                          .WORD   ^B00
2629
2630 026706     010                          .BYTE   DRVCLR          ;DRIVE CLEAR COMMAND
2631 026707     002                          .BYTE   2
2632 026710  000001                          .WORD   ^B01
2633
2634 026712     012                          .BYTE   RLEASE          ;RELEASE COMMAND
2635 026713     003                          .BYTE   3
2636 026714  000000                          .WORD   ^B000
2637
2638 026716     014                          .BYTE   OFFSET          ;OFFSET COMMAND
2639 026717     002                          .BYTE   2
2640 026720  000000                          .WORD   ^B00
2641
2642 026722     016                          .BYTE   RTC             ;RETURN TO CENTER COMMAND
2643 026723     002                          .BYTE   2
```

```
 2644 026724  000000                        .WORD   ^B00
 2645
 2646 026726     020                        .BYTE   RIP             ;READ IN PRESET COMMAND
 2647 026727     004                        .BYTE   4
 2648 026730  000016                        .WORD   ^B1110
 2649
 2650 026732     022                        .BYTE   PAKACK          ;PACK ACKNOWLEDGE
 2651 026733     004                        .BYTE   4
 2652 026734  000016                        .WORD   ^B1110
 2653
 2654 026736     024                        .BYTE   ILF24           ;ILLEGAL FUNCTION 24
 2655 026737     002                        .BYTE   2
 2656 026740  000000                        .WORD   ^B00
 2657
 2658 026742     026                        .BYTE   ILF26           ;ILLEGAL FUNCTION 26
 2659 026743     002                        .BYTE   2
 2660 026744  000000                        .WORD   ^B00
 2661
 2662 026746     030                        .BYTE   SEARCH          ;SEARCH COMMAND
 2663 026747     003                        .BYTE   3
 2664 026750  000000                        .WORD   ^B000
 2665
 2666 026752     032                        .BYTE   ILF32           ;ILLEGAL FUNCTION 32
 2667 026753     003                        .BYTE   3
 2668 026754  000000                        .WORD   ^B000
 2669
 2670 026756     034                        .BYTE   ILF34           ;ILLEGAL FUNCTION 34
 2671 026757     002                        .BYTE   2
 2672 026760  000000                        .WORD   ^B00
 2673
 2674 026762     036                        .BYTE   ILF36           ;ILLEGAL FUNCTION 36
 2675 026763     002                        .BYTE   2
 2676 026764  000000                        .WORD   ^B00
 2677
 2678 026766     000                        .BYTE                   ;END OF TABLE
 2679 026767     377                        .BYTE   -1
 2680 026770  000000                        .WORD
 2681
 2682 026772                        200$:                           ;END OF TEST
 2683
 2684                               ;;*******************************************************************
                                    ;*TEST 50        SET/RESET IVC TEST
                                    ;;*******************************************************************
      026772                        TST50:
      026772  000004                       SCOPE                   ;SCOPE CALL
      026774  000240                       NOP
      026776  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
      027002  013700  001276                MOV     $BASE,R0        ;R0   UNIBUS ADDRESS
      027006  013701  001466                MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
      027012  012737  000050  001226        MOV     #50,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
 2685
 2686 027020  010037  001136                MOV     R0,$BDADR       ;SETUP REG ADDRESS
 2687 027024  062737  000042  001136        ADD     #RMER2,$BDADR
 2688 027032  005002                        CLR     R2              ;R2=FUNCTION CODE
 2689
 2690                               ;INITIALIZE AND VERIFY THAT IVC STATUS IS ZERO.
```

```
2691 027034                                    10$:
2692 027034  004737  054674                          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2693 027040  012760  000001  000024                  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2694 027046  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2695 027054  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
2696 027062  012760  000000  000042                  MOV     #0,RMER2(R0)    ;LOAD RMER2
2697 027070  016037  000042  001142                  MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
2698 027076  042737  167777  001142                  BIC     #^CIVC,$BDDAT
2699 027104  001404                                  BEQ     20$             ;BRANCH IF IVC IS ZERO
2700 027106  005037  001140                          CLR     $GDDAT
2701 027112  104157                                  EMT     157
2702 027114  000444                                  BR      40$             ;SKIP REST OF TEST
2703
2704                                           ;LOAD THE FUNCTION CODE WITH GO BIT, STEP THE COMMAND SEQUENCER OFF
2705                                           ;ADDRESS 0 AND VERIFY IVC STATUS.
2706 027116                                    20$:
2707 027116  010203                                  MOV     R2,R3           ;SETUP FUNCTION CODE
2708 027120  052703  000001                          BIS     #GO,R3
2709 027124  010360  000000                          MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2710 027130  012760  141001  000024                  MOV     #DMD!MUR.DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
2711 027136  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2712 027144  016037  000042  001142                  MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
2713 027152  042737  167777  001142                  BIC     #^CIVC,$BDDAT   ;SET ACTUAL STATUS
2714 027160  016237  063416  001140                  MOV     FNCDTB(R2),$GDDAT  ;SETUP EXPECTED STATUS FROM
2715 027166  042737  167777  001140                  BIC     #^CIVC,$GDDAT   ;FUNCTION CODE TABLE
2716 027174  023737  001140  001142                  CMP     $GDDAT,$BDDAT
2717 027202  001403                                  BEQ     30$             ;BRANCH IF IVC IS OK
2718 027204  010237  001174                          MOV     R2,$TMP0        ;SAVE FUNCTION CODE FOR MSG
2719 027210  104160                                  EMT     160
2720
2721                                           ;ADVANCE FUNCTION CODE AND REPEAT TEST IF NOT DONE
2722 027212                                    30$:
2723 027212  062702  000002                          ADD     #2,R2
2724 027216  022702  000076                          CMP     #ILF76,R2
2725 027222  103401                                  BLO     40$             ;BRANCH IF DONE TEST
2726 027224  000703                                  BR      10$
2727
2728 027226                                    40$:                            ;END OF TEST
2729
2730                                           ;;*******************************************************************
                                               ;*TEST 51        SET LSC TEST

                                               ;;*******************************************************************
     027226                                    TST51:
     027226  000004                                  SCOPE                   ;SCOPE CALL
     027230  000240                                  NOP
     027232  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
     027236  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     027242  013701  001466                          MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
     027246  012737  000051  001226                  MOV     #51,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2731
2732 027254  010037  001136                          MOV     R0,$BDADR
2733 027260  062737  000042  001136                  ADD     #RMER2,$BDADR
2734
2735                                           ;INITIALIZE AND VERIFY THAT LOSS OF SYSTEM CLOCK, "LSC", IS RESET
2736 027266  004737  054674                          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2737 027272  012760  000001  000024                  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
```

```
2738 027300  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
2739 027306  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
2740 027314  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
2741 027322  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
2742 027330  042737  173777  001142          BIC     #^CLSC,$BDDAT
2743 027336  001403                          BEQ     10$             ;BRANCH IF LSC IS ZERO
2744 027340  005037  001140                  CLR     $GDDAT
2745 027344  104161                          EMT     161
2746
2747                                 ;WITH DEBUG CLOCK ENABLED, SET GO AND WAIT FOR ONE SHOT TO SET
2748 027346                          10$:
     027346  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
2749 027354  012737  000001  001534          MOV     #1,WATCH        ;SET WATCHDOG TIMER VALUE
     027362  004777  152150                  JSR     PC,@CLOCK       ;START THE CLOCK
2750 027366  005737  001534          20$:    TST     WATCH
2751 027372  001375                          BNE     20$             ;WAIT FOR WATCH ZERO
2752 027374  004777  152140                  JSR     PC,@STOPCL      ;STOP THE CLOCK
2753
2754                                 ;ONE SHOT SHOULD BE SET-DISABLE DIAGNOSTIC CLOCK AND LSC SHOULD SET.
2755 027400  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2756 027406  016037  000042  001142          MOV     RMER2(R0),$BDDAT        ;STORE RMER2 AT $BDDAT
2757 027414  042737  173777  001142          BIC     #^CLSC,$BDDAT
2758 027422  001004                          BNE     30$             ;BRANCH IF LSC SET
2759 027424  012737  004000  001140          MOV     #LSC,$GDDAT
2760 027432  104162                          EMT     162
2761
2762 027434                          30$:                            ;END OF TEST
2763
2764                                 ;;*******************************************************************
                                     ;*TEST 52        DECODE TEST

                                     ;;*******************************************************************
     027434                          TST52:
     027434  000004                          SCOPE                   ;SCOPE CALL
     027436  000240                          NOP
     027440  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     027444  013700  001276                  MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
     027450  013701  001466                  MOV     TSTQUE,R1       ;R1   POINTER TO DEVICE
     027454  012737  000052  001276          MOV     #52,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2765
2766 027462  005037  001426                  CLR     RMER10          ;NO ERROR FIRST TEST
2767 027466                          5$:
2768 027466  004737  030046                  JSR     PC,100$         ;INITIALIZE
2769
2770                                 ;EXECUTE A PACK ACKNOWLEDGE AND CHECK VOLUME VALID
2771 027472  013760  001426  000014          MOV     RMER10,RMER1(R0)        ;LOAD RMER1
2772 027500  012760  000023  000000          MOV     #PACACK!GO,RMCS1(R0)    ;LOAD RMCS1
2773 027506  012703  000003                  MOV     #3,R3
2774 027512                          10$:
     027512  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
2775 027520  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2776 027526  005303                          DEC     R3
2777 027530  001370                          BNE     10$             ;ISSUE NEXT CLOCK IF COUNT NOT 0
2778 027532  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2779 027540  042737  177677  001142          BIC     #^CVV,$BDDAT
2780 027546  001414                          BEQ     20$             ;BRANCH IF VV IS ZERO
2781 027550  005737  001426                  TST     RMER10
```

```
2782 027554  001527                        BEQ     70$             ;BRANCH IF VV SHOULD BE SET
2783 027556  005037  001140                CLR     $GDDAT          ;SETUP ERROR MESSAGE
2784 027562  010037  001136                MOV     R0,$BDADR
2785 027566  062737  000012  001136        ADD     #RMDS,$BDADR
2786 027574  104163                        EMT     163
2787 027576  000522                        BR      80$             ;SKIP
2788 027600                          20$:
2789 027600  004737  030046                JSR     PC,100$         ;INITIALIZE AND SET DIAGNOSTIC MODE
2790
2791                                 ;EXECUTE A READ IN PRESET AND CHECK VOLUME VALID
2792 027604  013760  001426  000014        MOV     RMER10,RMER1(R0)        ;LOAD RMER1
2793 027612  012760  000021  000000        MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
2794 027620  012703  000003                MOV     #3,R3           ;R3=CLOCK COUNT
2795 027624                          30$:
     027624  012760  141001  000024        MOV     #DMD!MUR!DBEN.DBCK,RMMR1(R0)    ;LOAD RMMR1
2796 027632  012760  041001  000024        MOV     #DMD.MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2797 027640  005303                        DEC     R3
2798 027642  001370                        BNE     30$             ;ISSUE NEXT CLOCK IF COUNT NOT ZERO
2799 027644  016037  000012  001142        MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2800 027652  042737  177677  001142        BIC     #^CVV,$BDDAT
2801 027660  001414                        BEQ     40$             ;BRANCH IF VOLUME VALID NOT SET
2802 027662  005737  001426                TST     RMER10
2803 027666  001462                        BEQ     70$             ;BRANCH IF VOLUME VALID SHOULD BE SET
2804 027670  005037  001140                CLR     $GDDAT          ;SETUP ERROR MESSAGE
2805 027674  010037  001136                MOV     R0,$BDADR
2806 027700  062737  000012  001136        ADD     #RMDS,$BDADR
2807 027706  104163                        EMT     163
2808 027710  000455                        BR      80$             ;SKIP
2809 027712                          40$:
2810 027712  004737  030046                JSR     PC,100$         ;INITIALIZE AND SET DIAGNOSTIC MODE
2811
2812                                 ;EXECUTE A WRITE CHECK DATA AND CHECK OCCUPIED
2813 027716  013760  001426  000014        MOV     RMER10,RMER1(R0)        ;LOAD RMER1
2814 027724  012760  000051  000000        MOV     #WCD!GO,RMCS1(R0)       ;LOAD RMCS1
2815 027732  012703  000002                MOV     #2,R3           ;R3 CLOCK COUNT
2816 027736                          50$:
     027736  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
2817 027744  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2818 027752  005303                        DEC     R3
2819 027754  001370                        BNE     50$             ;ISSUE NEXT CLOCK IF COUNT NOT ZERO
2820 027756  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
2821 027764  042737  077777  001142        BIC     #^COCC,$BDDAT
2822 027772  001414                        BEQ     60$             ;BRANCH IF OCCUPIED IS RESET
2823 027774  005737  001426                TST     RMER10
2824 030000  001415                        BEQ     70$             ;BRANCH IF OCCUPIED SHOULD BE SET
2825 030002  005037  001140                CLR     $GDDAT          ;SETUP ERROR MESSAGE
2826 030006  010037  001136                MOV     R0,$BDADR
2827 030012  062737  000024  001136        ADD     #RMMR1,$BDADR
2828 030020  104164                        EMT     164
2829 030022  000410                        BR      80$
2830
2831                                 ;VOLUME VALID AND OCCUPIED DID NOT SET-SEE IF COMP ERROR WAS ACTIVE
2832 030024                          60$:
2833 030024  005737  001426                TST     RMER10
2834 030030  001005                        BNE     80$             ;BRANCH IF COMP ERROR WAS SET
2835
2836                                 ;COULD NOT SET VV OR OCCUPIED-SUSPECT DECODE FLOP NOT SETTING
```

```
030032  104165                          EMT     165

                                ;REPEAT TEST WITH COMPOSITE ERROR ACTIVE-VERIFY THAT DECODE FLOP
                                ;DOES NOT SET, AS INDICATED BY VOLUME VALID AND OCCUPIED.
030034                          70$:
030034  012737  040000  001426          MOV     #UNS,RMER10     ;USE UNSAFE TO SET COMP ERROR
030042  000611                          BR      5$

030044  000510                  80$:    BR      200$            ;END OF TEST


                                ;;*************************************************************
                                ;SUBROUTINE USED DURING TEST
                                ;;*************************************************************

                                ;USING DIAGNOSTIC MODE, RESET VOLUME VALID AND COMPOSITE ERROR.
                                ;VERIFY THAT VV, ERR, AND OCC ARE ZERO.
030046                          100$:
030046  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
030052  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
030060  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
030066  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
030074  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
030102  005037  001140                  CLR     $GDDAT          ;SETUP FOR ERROR MSG
030106  010037  001136                  MOV     R0,$BDADR
030112  062737  000012  001136          ADD     #RMDS,$BDADR
030120  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
030126  042737  137777  001142          BIC     #^CERR,$BDDAT
030134  001402                          BEQ     110$            ;BRANCH IF COMP ERROR ZERO
030136  104143                          EMT     143
030140  000447                          BR      140$            ;SKIP TEST
030142                          110$:
030142  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
030150  042737  177677  001142          BIC     #^CVV,$BDDAT
030156  001402                          BEQ     120$            ;BRANCH IF VOLUME VALID ZERO.
030160  104135                          EMT     135
030162  000436                          BR      140$            ;SKIP TEST
030164                          120$:
030164  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
030172  042737  077777  001142          BIC     #^COCC,$BDDAT
030200  001407                          BEQ     130$            ;BRANCH IF OCCUPIED ZERO
030202  010037  001136                  MOV     R0,$BDADR       ;SETUP ERROR MESSAGE
030206  062737  000024  001136          ADD     #RMMR1,$BDADR
030214  104166                          EMT     166
030216  000420                          BR      140$            ;SKIP TEST

                                ;TO VERIFY THAT THE DECODE FLOP IS RESET, LOAD AN ILLEGAL FUNCTION
                                ;IN RMCS1 AND VERIFY THAT ILF DOES NOT SET.
030220                          130$:
030220  012760  000024  000000          MOV     #ILF24,RMCS1(R0)        ;LOAD RMCS1
030226  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
030234  042737  177776  001142          BIC     #^CILF,$BDDAT
030242  001410                          BEQ     150$            ;BRANCH IF ILF IS ZERO
030244  010037  001136                  MOV     R0,$BDADR       ;SETUP ERROR MESSAGE
030250  062737  000014  001136          ADD     #RMER1,$BDADR
030256  104167                          EMT     167
030260  012716  030266          140$:   MOV     #200$,(SP)      ;DONT GO BACK TO TEST
```

```
2892 030264  000207                    150$:   RTS     PC              ;RETURN TO TEST OR EXIT TEST
2893
2894 030266                             200$:
2895
2896                                    ;;*****************************************************************
                                        ;*TEST 53       SET/RESET VOLUME VALID TEST
                                        ;;*****************************************************************
     030266                             TST53:
     030266  000004                             SCOPE                   ;SCOPE CALL
     030270  000240                             NOP
     030272  012706  001100                     MOV     #STACK,SP       ;LOAD THE STACK POINTER
     030276  013700  001276                     MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     030302  013701  001466                     MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     030306  012737  000053  001226             MOV     #53,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2897
2898 030314  010037  001136                     MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS
2899 030320  062737  000012  001136             ADD     #RMDS,$BDADR
2900 030326  012702  030532                     MOV     #100$,R2        ;R2-TABLE POINTER
2901
2902                                    ;INITIALIZE AND USE DIAGNOSTIC MODE TO RESET VOLUME VALID
2903 030332                             10$:
2904 030332  004737  054674                     JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
2905 030336  012760  000001  000024             MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
2906 030344  012760  041001  000024             MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2907 030352  012760  000000  000014             MOV     #0,RMER1(R0)    ;LOAD RMER1
2908 030360  012760  000000  000042             MOV     #0,RMER2(R0)    ;LOAD RMER2
2909 030366  016037  000012  001142             MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2910 030374  042737  177677  001142             BIC     #^CVV,$BDDAT
2911 030402  001403                             BEQ     20$             ;BRANCH IF VOLUME VALID ZERO
2912 030404  005037  001140                     CLR     $GDDAT
2913 030410  104135                             EMT     135
2914
2915                                    ;EXECUTE THE FUNCTION CODE IN THE TABLE
2916 030412                             20$:
2917 030412  111203                             MOVB    (R2),R3         ;GET FUNCTION CODE
2918 030414  042703  177701                     BIC     #^C1F76,R3
2919 030420  052703  000001                     BIS     #GO,R3
2920 030424  010360  000000                     MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2921 030430  116204  000001                     MOVB    1(R2),R4        ;GET CLOCK COUNT
2922 030434  042704  177400                     BIC     #^C377,R4
2923 030440                             30$:
     030440  012760  141001  000024             MOV     #DMD!DBEN!MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
2924 030446  012760  041001  000024             MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
2925 030454  005304                             DEC     R4
2926 030456  001370                             BNE     30$             ;ISSUE CLOCKS TIL R4 ZERO
2927 030460  016037  000012  001142             MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
2928 030466  042737  177677  001142             BIC     #^CVV,$BDDAT
2929 030474  001007                             BNE     40$             ;BRANCH IF VOLUME VALID SET
2930 030476  010337  001174                     MOV     R3,$TMP0        ;SAVE FUNCTION CODE FOR MSG
2931 030502  012737  000100  001140             MOV     #VV,$GDDAT
2932 030510  104170                             EMT     170
2933 030512  000406                             BR      50$
2934
2935                                    ;ADVANCE THE TABLE POINTER, EXIT IF DONE
2936 030514                             40$:
2937 030514  062702  000002                     ADD     #2,R2
```

```
2938 030520  105762  000001                TSTB    1(R2)
2939 030524  100401                       BMI     50$             ;EXIT IF COUNT IS NEGATIVE
2940 030526  000701                       BR      10$
2941 030530  000403                50$:    BR      200$            ;JUMP OVER TABLE
2942
2943                                    ;TABLE OF FUNCTION CODES AND CLOCK COUNTS
2944 030532                             100$:
2945 030532      020                           .BYTE   RIP             ;READ IN PRESET COMMAND
2946 030533      003                           .BYTE   3
2947
2948 030534      022                           .BYTE   PAKACK          ;PACK ACKNOWLEDGE COMMAND
2949 030535      003                           .BYTE   3
2950
2951 030536      000                           .BYTE                   ;END OF TABLE
2952 030537      377                           .BYTE   -1
2953
2954 030540                             200$:                           ;END OF TEST
2955
2956                                    ;;*****************************************************************
                                       ;*TEST 54        ILLEGAL FUNCTION TEST

                                       ;;*****************************************************************
     030540                            TST54:
     030540  000004                            SCOPE                   ;SCOPE CALL
     030542  000240                            NOP
     030544  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
     030550  013700  001276                    MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     030554  013701  001466                    MOV     TSTCUE,R1       ;R1 = POINTER TO DEVICE
     030560  012737  000054  001226            MOV     #54,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
2957
2958 030566  005002                            CLR     R2              ;INITIALIZE FUNCTION CODE VALUE
2959 030570                             10$:
2960 030570  004737  054450                    JSR     PC,SETVV        ;GO SET VOLUME VALID
     030574  000402                            BR      20$             ;BRANCH TO 20$ IF NO ERROR
     030576  104000                            EMT
2961 030600  000460                            BR      50$             ;SKIP TEST IF ERROR
2962 030602  012704  000002                20$: MOV     #2,R4           ;R4=CLOCK COUNT
2963
2964                                    ;EXECUTE THE TEST FUNCTION CODE AND VERIFY ILF
2965 030606  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2966 030614  010203                            MOV     R2,R3           ;SETUP FUNCTION CODE IN R3
2967 030616  052703  000001                    BIS     #GO,R3
2968 030622  010360  000000                    MOV     R3,RMCS1(R0)    ;LOAD RMCS1
2969 030626                             30$:
     030626  012760  141001  000024            MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
2970 030634  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
2971 030642  005304                            DEC     R4
2972 030644  001370                            BNE     30$
2973 030646  016037  000014  001142            MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
2974 030654  042737  177776  001142            BIC     #^CILF,$BDDAT   ;SETUP ACTUAL ILF STATUS
2975 030662  016237  063416  001140            MOV     FNCDTB(R2),$GDDAT  ;GET EXPECTED ILF STATUS
2976 030670  042737  177776  001140            BIC     #^CILF,$GDDAT
2977 030676  023737  001140  001142            CMP     $GDDAT,$BDDAT
2978 030704  001410                            BEQ     40$             ;BRANCH IF ILF IS OK
2979 030706  010037  001136                    MOV     R0,$BDADR       ;SETUP FOR ERROR MSG
2980 030712  062737  000014  001136            ADD     #RMER1,$BDADR
2981 030720  010237  001174                    MOV     R2,$TMP0
```

```
2982 030724  104171                        EMT     171
2983
2984                                    ;ADVANCE TO THE NEXT FUNCTION CODE AND REPEAT TEST
2985 030726                             40$:
2986 030726  062702  000002                ADD     #2,R2
2987 030732  022702  000076                CMP     #ILF76,R2
2988 030736  103401                        BLO     50$
2989 030740  000713                        BR      10$
2990 030742                             50$:                            ;END OF TEST
2991
2992                                    ;;*****************************************************************
                                        ;*TEST 55        OCCUPIED TEST
                                        ;;*****************************************************************
     030742                             TST55:
     030742  000004                        SCOPE                       ;SCOPE CALL
     030744  000240                        NOP
     030746  012706  001100                MOV     #STACK,SP           ;LOAD THE STACK POINTER
     030752  013700  001276                MOV     $BASE,R0            ;R0 = UNIBUS ADDRESS
     030756  013701  001466                MOV     TSTQUE,R1           ;R1 = POINTER TO DEVICE
     030762  012737  000055  001226        MOV     #55,$TESTN          ;;SET TEST NUMBER IN APT MAIL BOX
2993
2994 030770  005002                        CLR     R2                          ;INITIALIZE FUNCTION CODE
2995
2996                                    ;GET THE DEVICE READY
2997 030772                             10$:
2998 030772  004737  054450                JSR     PC,SETVV            ;GO SET VOLUME VALID
     030776  000402                        BR      20$                 ;BRANCH TO 20$ IF NO ERROR
     031000  104000                        EMT
2999 031002  000464                        BR      50$
3000
3001                                    ;ENABLE DEBUG CLOCK AND LOAD THE FUNCTION CODE
3002 031004                             20$:
3003 031004  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3004 031012  010203                        MOV     R2,R3                       ;ASSEMBLE FUNCTION CODE AND
3005 031014  052703  000001                BIS     #GO,R3                      ;GO BIT IN R3
3006 031020  010360  000000                MOV     R3,RMCS1(R0)        ;LOAD RMCS1
3007 031024  012704  000002                MOV     #2,R4                       ;R4-CLOCK COUNT
3008
3009                                    ;STEP THE DEBUG CLOCK UNTIL SET PULSE IS ACTIVE
3010 031030                             30$:
3011 031030  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3012 031036  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3013 031044  005304                        DEC     R4
3014 031046  001370                        BNE     30$                         ;ISSUE NEXT CLOCK TIL R4 ZERO
3015
3016                                    ;VERIFY OCCUPIED STATUS
3017 031050  016037  000024  001142        MOV     RMMR1(R0),$BDDAT            ;STORE RMMR1 AT $BDDAT
3018 031056  042737  077777  001142        BIC     #^COCC,$BDDAT
3019 031064  005037  001140                CLR     $GDDAT              ;GENERATE OCC FROM AOE
3020 031070  032762  001000  063416        BIT     #AOE,FNCDTB(R2)
3021 031076  001403                        BEQ     35$
3022 031100  012737  100000  001140        MOV     #OCC,$GDDAT
3023 031106  023737  001140  001142  35$:  CMP     $GDDAT,$BDDAT
3024 031114  001411                        BEQ     40$                         ;BRANCH IF OCC IS OK
3025 031116  010237  001174                MOV     R2,$TMP0                    ;SAVE FUNCTION CODE
3026 031122  010037  001136                MOV     R0,$BDADR                   ;SETUP REGISTER ADDRESS
```

```
3027 031126  062737  000024  001136          ADD     #RMMR1,$BDADR
3028 031134  104173                          EMT     173
3029 031136  000406                          BR      50$
3030
3031                                  ;ADVANCE TO NEXT FUNCTIONCODE, EXIT IF DONE
3032 031140                           40$:
3033 031140  062702  000002                    ADD     #2,R2
3034 031144  022702  000076                    CMP     #ILF76,R2
3035 031150  103401                          BLO     50$                     ;EXIT IF DONE
3036 031152  000707                          BR      10$
3037
3038 031154                           50$:                                    ;END OF TEST
3039
3040                                  ;;*****************************************************************
                                      ;*TEST 56        READ IN PRESET TEST
                                      ;;*****************************************************************
     031154                           TST56:
     031154  000004                          SCOPE                   ;SCOPE CALL
     031156  000240                          NOP
     031160  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     031164  013700  001276                  MOV     $BASE,R0        ;R0   UNIBUS ADDRESS
     031170  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     031174  012737  000056  001226          MOV     #56,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3041
3042                                  ;CLEAR AND ENABLE DEBUG CLOCK - LEAVE VOLUME VALID RESET
3043 031202  004737  054674                    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3044 031206  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3045 031214  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3046 031222  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
3047 031230  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
3048
3049                                  ;LOAD ALL ONES IN RMDA, RMDC AND RMOF
3050 031236  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
3051 031244  012760  177777  000034          MOV     #-1,RMDC(R0)    ;LOAD RMDC
3052 031252  012760  177777  000032          MOV     #-1,RMOF(R0)    ;LOAD RMOF
3053
3054                                  ;LOAD READ IN PRESET COMMAND AND STEP THE CLOCK TILL SET PULSE
3055 031260  012760  000021  000000          MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
3056 031266  012702  000003                  MOV     #3,R2                   ;R2-CLOCK COUNT
3057 031272                           10$:
3058 031272  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3059 031300  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3060 031306  005302                          DEC     R2
3061 031310  001370                          BNE     10$                     ;ISSUE 3 CLOCKS
3062
3063                                  ;SEE IF RMDA OR RMDC OR RMOF IS ZERO
3064 031312  016002  000006                  MOV     RMDA(R0),R2     ;STORE RMDA AT R2
3065 031316  005702                          TST     R2
3066 031320  001413                          BEQ     20$                     ;BRANCH IF RMDA IS ZERO
3067 031322  016002  000034                  MOV     RMDC(R0),R2     ;STORE RMDC AT R2
3068 031326  042702  176000                  BIC     #XNUDC,R2               ;CLEAR UNUSED BITS
3069 031332  001406                          BEQ     20$                     ;BRANCH IF RMDC IS ZERO
3070 031334  016002  000032                  MOV     RMOF(R0),R2     ;STORE RMOF AT R2
3071 031340  042702  161577                  BIC     #XNUOF,R2               ;CLEAR UNUSED BITS
3072 031344  001401                          BEQ     20$                     ;BRANCH IF RMOF IS ZERO
3073
```

```
3074
3075 031346  104174                    ;READ IN PRESET COMMAND DIDNT CLEAR ANY OF THE 3 REGISTERS
                                              EMT     174
3076
3077 031350                             20$:                                    ;END OF TEST
3078
3079                                    ;;*********************************************************************
                                        ;*TEST 57        RIP/RMOF TEST
                                        ;;*********************************************************************
     031350                             TST57:
     031350  000004                            SCOPE                   ;SCOPE CALL
     031352  000240                            NOP
     031354  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
     031360  013700  001276                    MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     031364  013701  001466                    MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     031370  012737  000057  001226            MOV     #57,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3080
3081 031376  010037  001136                    MOV     R0,$BDADR               ;SETUP REGISTER ADDRESS AND
3082 031402  062737  000032  001136            ADD     #RMOF,$BDADR
3083 031410  005037  001140                    CLR     $GDDAT                  ;EXPECTED RMOF
3084
3085                                    ;INITIALIZE AND SET BITS IN RMOF
3086 031414  004737  054674                    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3087 031420  012760  000001  000024            MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3088 031426  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3089 031434  012760  000000  000014            MOV     #0,RMER1(R0)    ;LOAD RMER1
3090 031442  012760  000000  000042            MOV     #0,RMER2(R0)    ;LOAD RMER2
3091 031450  012760  177777  000032            MOV     #-1,RMOF(R0)    ;LOAD RMOF
3092
3093                                    ;EXECUTE A READ IN PRESET IN DIAGNOSTIC MODE TILL SET PULSE
3094 031456  012760  000021  000000            MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
3095 031464  012702  000003                    MOV     #3,R2                   ;R2=CLOCK COUNT
3096 031470                             10$:
     031470  012760  141001  000024            MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3097 031476  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3098 031504  005302                            DEC     R2
3099 031506  001370                            BNE     10$                     ;ISSUE 3 CLOCKS
3100
3101                                    ;VERIFY THAT RMOF IS ZERO
3102 031510  016037  000032  001142            MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
3103 031516  042737  161577  001142            BIC     #XNUOF,$BDDAT
3104 031524  001401                            BEQ     20$                     ;BRANCH IF RMOF IS ZERO
3105 031526  104175                            EMT     175
3106
3107 031530                             20$:                                    ;END OF TEST
3108
3109                                    ;;*********************************************************************
                                        ;*TEST 60        RMDA/RMDC/RIP TEST
                                        ;;*********************************************************************
     031530                             TST60:
     031530  000004                            SCOPE                   ;SCOPE CALL
     031532  000240                            NOP
     031534  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
     031540  013700  001276                    MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
     031544  013701  001466                    MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
     031550  012737  000060  001226            MOV     #60,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
```

```

031556  005037  001140                          CLR     $GDDAT

                                        ;CLEAR, ENABLE DEBUG CLOCK, THEN PRESET RMDA AND RMDC
031562  004737  054674                          JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
031566  012760  000001  000024                  MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
031574  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
031602  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
031610  012760  000000  000014                  MOV     #0,RMER1(R0)    ;LOAD RMER1
031616  012760  177777  000006                  MOV     #-1,RMDA(R0)    ;LOAD RMDA
031624  012760  177777  000034                  MOV     #-1,RMDC(R0)    ;LOAD RMDC

                                        ;EXECUTE READ IN PRESET TILL SET PULSE
031632  012760  000021  000000                  MOV     #RIP!GO,RMCS1(R0)       ;LOAD RMCS1
031640  012702  000003                          MOV     #3,R2
031644                                  10$:
031644  012760  141001  000024                  MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
031652  012760  041001  000024                  MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
031660  005302                                  DEC     R2
031662  001370                                  BNE     10$                     ;ISSUE 3 CLOCKS

                                        ;VERIFY RMDA IS ZERO
031664  016037  000006  001142                  MOV     RMDA(R0),$BDDAT ;STORE RMDA AT $BDDAT
031672  005737  001142                          TST     $BDDAT
031676  001406                                  BEQ     20$                     ;BRANCH IF RMDA RESET
031700  010037  001136                          MOV     R0,$BDADR
031704  062737  000006  001176                  ADD     #RMDA,$BDADR
031712  104176                                  EMT     176

                                        ;VERIFY RMDC IS ZERO
031714                                  20$:
031714  016037  000034  001142                  MOV     RMDC(R0),$BDDAT ;STORE RMDC AT $BDDAT
031722  042737  176000  001142                  BIC     #XNUDC,$BDDAT
031730  001406                                  BEQ     30$                     ;BRANCH IF RMDC RESET
031732  010037  001136                          MOV     R0,$BDADR
031736  062737  000034  001136                  ADD     #RMDC,$BDADR
031744  104302                                  EMT     302
031746                                  30$:                                    ;END OF TEST


                                        ;;*****************************************************************
                                        ;*TEST 61         OFFSET COMMAND TEST

                                        ;;*****************************************************************
031746                                  TST61:
031746  000004                                  SCOPE                   ;SCOPE CALL
031750  000240                                  NOP
031752  012706  001100                          MOV     #STACK,SP       ;LOAD THE STACK POINTER
031756  013700  001276                          MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
031762  013701  001466                          MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
031766  012737  000061  001226                  MOV     #61,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX

031774  010037  001136                          MOV     R0,$BDADR
032000  062737  000012  001136                  ADD     #RMDS,$BDADR
032006  012737  000001  001140                  MOV     #OM,$GDDAT

032014  004737  054450                          JSR     PC,SETVV        ;GO SET VOLUME VALID
032020  000402                                  BR      10$             ;BRANCH TO 10$ IF NO ERROR
```

```
'E1     OFFSET COMMAND TEST

        032022  104000                          EMT
3155    032024  000453                          BR      40$
3156    032026                          10$:
        032026  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
3157
3158                                    ;ENABLE DEBUG CLOCK AND EXECUTE OFFSET COMMAND
3159    032034  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3160    032042  012760  000015  000000          MOV     #OFFSET!GO,RMCS1(R0)    ;LOAD RMCS1
3161    032050  012702  000002                  MOV     #2,R2                   ;R2=CLOCK COUNT
3162    032054                          20$:
        032054  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3163    032062  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3164    032070  005302                          DEC     R2
3165    032072  001370                          BNE     20$                     ;ISSUE 2 CLOCKS
3166
3167                                    ;VERIFY THAT OFFSET MODE IS SET
3168    032074  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3169    032102  042737  177776  001142          BIC     #^C<OM>,$BDDAT
3170    032110  001001                          BNE     40$                     ;BRANCH IF OM IS SET
3171    032112  104200                          EMT     200
3172    032114                          40$:                                    ;END OF TEST
3173
3174                                    ;;*******************************************************************
                                        ;*TEST 62        RETURN TO CENTER TEST
                                        ;;*******************************************************************
        032114                          TST62:
        032114  000004                          SCOPE                   ;SCOPE CALL
        032116  000240                          NOP
        032120  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        032124  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        032130  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        032134  012737  000062  001226          MOV     #62,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3175
3176    032142  010037  001136                  MOV     R0,$BDADR
3177    032146  062737  000012  001136          ADD     #RMDS,$BDADR
3178
3179    032154  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
        032160  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
        032162  104000                          EMT
3180    032164  000465                          BR      60$
3181
3182                                    ;SET OFFSET DIRECTION AND OFFSET
3183    032166                          10$:
3184    032166  012760  000200  000032          MOV     #OFD,RMOF(R0)   ;LOAD RMOF
3185    032174  004737  054572                  JSR     PC,SETOM        ;GO SET OFFSET MODE
        032200  000401                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
        032202  104000                          EMT
3186
3187                                    ;ENABLE DEBUG CLOCK AND EXECUTE RETURN TO CENTER COMMAND
3188    032204                          20$:
3189    032204  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
        032210  000402                          BR      30$             ;BRANCH TO 30$ IF NO ERROR
        032212  104000                          EMT
3190    032214  000451                          BR      60$
3191    032216                          30$:
3192    032216  012760  041001  000024          MOV     #DMD!MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
```

```
3193 032224  012760  000017  000000          MOV     #RTC!GO,RMCS1(R0)       ;LOAD RMCS1
3194 032232  012702  000002                  MOV     #2,R2
3195 032236                          40$:
     032236  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3196 032244  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3197 032252  005302                          DEC     R2
3198 032254  001370                          BNE     40$                     ;ISSUE 2 CLOCKS
3199
3200                                 ;VERIFY THAT OFFSET MODE IS RESET
3201 032256  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3202 032264  042737  177776  001142          BIC     #^COM,$BDDAT
3203 032272  001403                          BEQ     50$                     ;BRANCH IF OFFSET MODE RESET
3204 032274  005037  001140                  CLR     $GDDAT
3205 032300  104201                          EMT     201
3206
3207                                 ;VERIFY THAT OFFSET DIRECTION IS RESET
3208 032302                          50$:
3209 032302  016037  000032  001142          MOV     RMOF(R0),$BDDAT ;STORE RMOF AT $BDDAT
3210 032310  042737  177577  001142          BIC     #^COFD,$BDDAT
3211 032316  001410                          BEQ     60$                     ;BRANCH.IF OFD IS RESET
3212 032320  005037  001140                  CLR     $GDDAT
3213 032324  010037  001136                  MOV     R0,$BDADR
3214 032330  062737  000032  001136          ADD     #RMOF,$BDADR
3215 032336  104202                          EMT     202
3216 032340                          60$:                                    ;END OF TEST
3217
3218                                 ;;*****************************************************************
                                     ;*TEST 63        RMDC CLEAR OFFSET TEST

                                     ;;*****************************************************************
     032340                          TST63:
     032340  000004                          SCOPE                   ;SCOPE CALL
     032342  000240                          NOP
     032344  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     032350  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     032354  013701  001466                  MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
     032360  012737  000063  001226          MOV     #63,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3219
3220 032366  010037  001136                  MOV     R0,$BDADR
3221 032372  062737  000012  001136          ADD     #RMDS,$BDADR
3222
3223 032400  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     032404  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
     032406  104000                          EMT
3224 032410  000421                          BR      40$                     ;SKIP REST OF TEST
3225 032412                          10$:
3226 032412  004737  054572                  JSR     PC,SETOM        ;GO SET OFFSET MODE
     032416  000401                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
     032420  104000                          EMT
3227
3228                                 ;WRITE THE DESIRED CYLINDER REGISTER AND VERIFY THAT OFFSET IS ZERO
3229 032422                          20$:
3230 032422  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
3231 032430  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3232 032436  042737  177776  001142          BIC     #^COM,$BDDAT
3233 032444  001403                          BEQ     40$
3234 032446  005037  001140                  CLR     $GDDAT
```

```
3235 032452  104203                          EMT     203
3236
3237 032454                                 40$:                                    ;END OF TEST
3238
3239                                        ;;*************************************************************
                                            ;*TEST 64        EBL CLEAR OFFSET TEST
                                            ;;*************************************************************
     032454                                 TST64:
     032454  000004                                SCOPE                   ;SCOPE CALL
     032456  000240                                NOP
     032460  012706  001100                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
     032464  013700  001276                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     032470  013701  001466                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     032474  012737  000064  001226                MOV     #64,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3240
3241 032502  013737  001334  001420                MOV     LSTRK,RMDAO     ;SETUP LAST TRACK AND
3242 032510  112737  000037  001420                MOVB    #31.,RMDAO      ;LAST SECTOR
3243 032516  010037  001136                        MOV     R0,$BDADR       ;SETUP REGISTER FOR ERROR MSG
3244 032522  062737  000012  001136                ADD     #RMDS,$BDADR
3245
3246 032530  004737  054450                        JSR     PC,SETVV        ;GO SET VOLUME VALID
     032534  000402                                BR      10$             ;BRANCH TO 10$ IF NO ERROR
     032536  104000                                EMT
3247 032540  000440                                BR      30$                     ;SKIP REST OF TEST IF ERROR
3248 032542                                 10$:
3249 032542  004737  054572                        JSR     PC,SETOM        ;GO SET OFFSET MODE
     032546  000401                                BR      20$             ;BRANCH TO 20$ IF NO ERROR
     032550  104000                                EMT
3250 032552                                 20$:
3251 032552  012760  010000  000032                MOV     #FMT16,RMOF(R0) ;LOAD RMOF
3252 032560  013760  001420  000006                MOV     RMDAO,RMDA(R0)  ;LOAD RMDA
3253
3254                                        ;FORCE END OF BLOCK AND VERIFY THAT OFFSET MODE IS CLEARED
3255 032566  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3256 032574  012760  000001  000000                MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
3257 032602  012760  061001  000024                MOV     #DMD!MUR!DBEN!DEBL,RMMR1(R0)    ;LOAD RMMR1
3258 032610  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3259 032616  016037  000012  001142                MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3260 032624  042737  177776  001142                BIC     #^COM,$BDDAT
3261 032632  001403                                BEQ     30$                     ;BRANCH IF OFFSET IS ZERO
3262 032634  005037  001140                        CLR     $GDDAT
3263 032640  104204                                EMT     204
3264 032642                                 30$:                                    ;END OF TEST
3265
3266                                        ;;*************************************************************
                                            ;*TEST 65        RUN AND GO TEST
                                            ;;*************************************************************
     032642                                 TST65:
     032642  000004                                SCOPE                   ;SCOPE CALL
     032644  000240                                NOP
     032646  012706  001100                        MOV     #STACK,SP       ;LOAD THE STACK POINTER
     032652  013700  001276                        MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     032656  013701  001466                        MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     032662  012737  000065  001226                MOV     #65,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3267
```

```
3268 032670  005037  001140              CLR     $GDDAT                  ;INITIALIZE EXPECTED RESULT
3269 032674  005002                      CLR     R2                      ;INITIALIZE FUNCTION CODE
3270 032676  010037  001136              MOV     R0,$BDADR
3271 032702  062737  000024  001136      ADD     #RMMR1,$BDADR
3272
3273                                  ;CLEAR THE MASSBUS AND ENABLE DEBUG CLOCK
3274 032710                           10$:
3275 032710  004737  054674              JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3276 032714  012760  000001  000024      MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3277 032722  012760  040001  000024      MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
3278 032730  012760  000000  000014      MOV     #0,RMER1(R0)    ;LOAD RMER1
3279 032736  012760  000000  000042      MOV     #0,RMER2(R0)    ;LOAD RMER2
3280
3281                                  ;LOAD THE FUNCTION CODE AND VERIFY RUN AND GO FLOP
3282 032744  010203                      MOV     R2,R3                   ;ASSEMBLE FUNCTION CODE AND GO
3283 032746  052703  000001              BIS     #GO,R3
3284 032752  012737  000200  001534      MOV     #200,WATCH      ;SET WATCHDOG TIMER VALUE
     032760  004777  146552              JSR     PC,@CLOCK       ;START THE CLOCK
3285 032764  010360  000000              MOV     R3,RMCS1(R0)    ;LOAD RMCS1
3286 032770                           15$:
     032770  016037  000024  001142      MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
3287 032776  042737  137777  001142      BIC     #^CRG,$BDDAT
3288 033004  023737  001140  001142      CMP     $GDDAT,$BDDAT
3289 033012  001411                      BEQ     20$                     ;BRANCH IF RUN AND GO FLOP OK
3290 033014  005737  001534              TST     WATCH           ;TAKE ANOTHER SAMPLE IF CLOCK NOT ZERO
3291 033020  001363                      BNE     15$
3292 033022  004777  146512              JSR     PC,@STOPCL      ;STOP THE CLOCK
3293 033026  010237  001174              MOV     R2,$TMP0                ;SAVE FUNCTION CODE FOR MSG
3294 033032  104205                      EMT     205
3295 033034  000416                      BR      40$                     ;SKIP REST OF
3296 033036                           20$:
3297 033036  004777  146476              JSR     PC,@STOPCL      ;STOP THE CLOCK
3298
3299                                  ;ADVNACE TO NEXT FUNCTION CODE - EXIT IF DONE
3300 033042  062702  000002              ADD     #2,R2
3301 033046  022702  000076              CMP     #ILF76,R2
3302 033052  103407                      BLO     40$                     ;EXIT IF DONE
3303 033054  020227  000050              CMP     R2,#WCD                 ;CHANGE EXPECTED RESULTI IF
3304 033060  103403                      BLO     30$                     ;DATA COMMAND
3305 033062  012737  040000  001140      MOV     #RG,$GDDAT
3306 033070  000707                  30$:BR      10$                     ;REPEAT TEST
3307
3308 033072                           40$:                            ;END OF TEST
3309
3310                                  ;;*****************************************************************
                                      ;*TEST 66        SET IAE TEST

                                      ;;*****************************************************************
     033072                           TST66:
     033072  000004                      SCOPE                   ;SCOPE CALL
     033074  000240                      NOP
     033076  012706  001100              MOV     #STACK,SP       ;LOAD THE STACK POINTER
     033102  013700  001276              MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
     033106  013701  001466              MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
     033112  012737  000066  001226      MOV     #66,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3311
3312 033120  012702  033266              MOV     #100$,R2                ;R2-TABLE POINTER
```

```
3313 033124                          10$:
3314 033124  004737  C>4450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     033130  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
     033132  104000                          EMT
3315 033134  000453                          BR      50$                     ;SKIP REST OF TEST
3316
3317                                 ;LOAD INVALID TRACK, SECTOR AND CYLINDER ADDRESS AND SET FORMAT TO 18
3318 033136                          20$:
3319 033136  012760  177777  000006          MOV     #-1,RMDA(R0)    ;LOAD RMDA
3320 033144  012760  177777  000034          MOV     #-1,RMDC(R0)    ;LOAD RMDC
3321 033152  012760  000000  000032          MOV     #0,RMOF(R0)     ;LOAD RMOF
3322
3323                                 ;ENABLE DEBUG CLOCK AND LOAD FUNCTION CODE IN RMCSI WITH GO ON
3324 033160  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3325 033166  111203                          MOVB    (R2),R3
3326 033170  042703  177701                  BIC     #^C1LF76,R3
3327 033174  052703  000001                  BIS     #GO,R3
3328 033200  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
3329 033204  116204  000001                  MOVB    1(R2),R4                ;GET CLOCK COUNT
3330 033210  042704  177400                  BIC     #^C377,R4
3331
3332                                 ;CLOCK THE COMMAND SEQUENCER
3333 033214                          30$:
3334 033214  012760  141001  000024          MOV     #DMD.MUR!DBEN'DBCK,RMMR1(R0)    ;LOAD RMMR1
3335 033222  012760  041001  000024          MOV     #DMD MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
3336 033230  005304                          DEC     R4
3337 033232  001370                          BNE     30$
3338
3339                                 ;SEE IF IAE HAS SET
3340 033234  016004  000014                  MOV     RMER1(R0),R4    ;STORE RMER1 AT R4
3341 033240  042704  175777                  BIC     #^CIAE,R4
3342 033244  001007                          BNE     50$                     ;BRANCH IF IAE SET
3343
3344                                 ;IAE DID NOT SET - TRY ANOTHER FUNCTION CODE
3345 033246  062702  000002                  ADD     #2,R2
3346 033252  105762  000001                  TSTB    1(R2)
3347 033256  100401                          BMI     40$                     ;BRANCH IF ALL CODES TRIED
3348 033260  000721                          BR      10$
3349
3350                                 ;CANNOT SET IAE WITH ANY COMBINATION OF ADDRESS AND FUNCTION CODE
3351 033262                          40$:
3352 033262  104206                          EMT     206
3353
3354 033264  000411                  50$:    BR      200$                    ;JUMP OVER TABLE
3355
3356                                 ;TABLE OF FUNCTION CODES AND CLOCK COUNTS FOR TEST
3357 033266                          100$:
3358 033266       030                        .BYTE   SEARCH                  ;SEARCH COMMAND
3359 033267       002                        .BYTE   2
3360
3361 033270       004                        .BYTE   SEEK                    ;SEEK COMMAND
3362 033271       002                        .BYTE   2
3363
3364 033272       062                        .BYTE   WH                      ;WRITE HEADER COMMAND
3365 033273       002                        .BYTE   2
3366
3367 033274       052                        .BYTE   WCH                     ;WRITE CHECK HEADER COMMAND
```

```
3368 033275  002                        .BYTE   2
3369
3370 033276  072                        .BYTE   RH                      ;READ HEADER COMMAND
3371 033277  002                        .BYTE   2
3372
3373 033300  060                        .BYTE   WD                      ;WRITE DATA COMMAND
3374 033301  002                        .BYTE   2
3375
3376 033302  050                        .BYTE   WCD                     ;WRITE CHECK DATA COMMAND
3377 033303  002                        .BYTE   2
3378
3379 033304  070                        .BYTE   RD                      ;READ DATA COMMAND
3380 033305  002                        .BYTE   2
3381
3382 033306  000                        .BYTE                           ;END OF TABLE
3383 033307  377                        .BYTE   -1
3384
3385 033310                        200$:                                ;END OF TEST
3386
3387                                ;;*****************************************************************
                                    ;*TEST 67        SEARCH, SEEK, READ, WRITE TEST
                                    ;;*****************************************************************
     033310                         TST67:
     033310  000004                         SCOPE                      ;SCOPE CALL
     033312  000240                         NOP
     033314  012706  001100                 MOV     #STACK,SP          ;LOAD THE STACK POINTER
     033320  013700  001276                 MOV     $BASE,R0           ;R0 = UNIBUS ADDRESS
     033324  013701  001466                 MOV     TSTQUE,R1          ;R1 = POINTER TO DEVICE
     033330  012737  000067  001226         MOV     #67,$TESTN         ;;SET TEST NUMBER IN APT MAIL BOX
3388
3389 033336  005002                         CLR     R2                         ;INITIALIZE FUNCTION CODE
3390 033340                         10$:
3391 033340  004737  054450                JSR     PC,SETVV           ;GO SET VOLUME VALID
     033344  000402                         BR      20$                ;BRANCH TO 20$ IF NO ERROR
     033346  104000                         EMT
3392 033350  000472                         BR      50$
3393
3394                                ;LOAD INVALID TRACK, SECTOR AND CYLINDER ADDRESS AND SET FORMAT
3395                                ;TO 18 BIT MODE
3396 033352                         20$:
3397 033352  012760  177777  000006        MOV     #-1,RMDA(R0)       ;LOAD RMDA
3398 033360  012760  177777  000034        MOV     #-1,RMDC(R0)       ;LOAD RMDC
3399 033366  012760  000000  000032        MOV     #0,RMOF(R0)        ;LOAD RMOF
3400
3401                                ;ENABLE DEBUG CLOCK AND LOAD FUNCTION CODE IN RMCS1 WITH GO ON
3402 033374  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3403 033402  010203                         MOV     R2,R3                      ;ASSEMBLE CODE AND GO
3404 033404  052703  000001                 BIS     #GO,R3
3405 033410  010360  000000                 MOV     R3,RMCS1(R0)       ;LOAD RMCS1
3406
3407                                ;CLOCK THE COMMAND SEQUENCER TO SET PULSE
3408 033414  012704  000002                 MOV     #2,R4
3409 033420                         30$:
     033420  012760  141001  000024        MOV     #DMD.MUR.DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3410 033426  012760  041001  000024        MOV     #DMD MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
3411 033434  005304                         DEC     R4
```

```
3412 033436  001370                        BNE     30$
3413
3414                                ;VERIFY IAE ACCORDING TO FUNCTION CODE TABLE
3415 033440  016037  000014  001142        MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
3416 033446  042737  175777  001142        BIC     #^CIAE,$BDDAT
3417 033454  016237  063416  001140        MOV     FNCDTB(R2),$GDDAT       ;ASSEMBLE EXPECTED IAE
3418 033462  042737  175777  001140        BIC     #^CIAE,$GDDAT
3419 033470  023737  001140  001142        CMP     $GDDAT,$BDDAT
3420 033476  001411                        BEQ     40$                     ;BRANCH IF IAE OK
3421 033500  010037  001136                MOV     R0,$BDADR               ;SET UP ERROR MSG
3422 033504  062737  000014  001136        ADD     #RMER1,$BDADR
3423 033512  010237  001174                MOV     R2,$TMP0
3424 033516  104207                        EMT     207
3425 033520  000406                        BR      50$                     ;SKIP REST OF TEST
3426
3427                                ;ADVANCE TO NEXT FUNCTION CODE - EXIT IF DONE
3428 033522                         40$:
3429 033522  062702  000002                ADD     #2,R2
3430 033526  023702  000076                CMP     ILF76,R2
3431 033532  103401                        BLO     50$
3432 033534  000701                        BR      10$
3433 033536                         50$:                                    ;END OF TEST
3434
3435                                ;;*****************************************************************
                                    ;*TEST 70        INVALID TRACK/SECTOR TEST

                                    ;;*****************************************************************
     033536                         TST70:
     033536  000004                        SCOPE                   ;SCOPE CALL
     033540  000240                        NOP
     033542  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     033546  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     033552  013701  001466                MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     033556  012737  000070  001226        MOV     #70,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3436
3437 033564  013737  001334  001420        MOV     LSTRK,RMDA0             ;INITIALIZE TRACK ADDRESS
3438 033572  105237  001421                INCB    RMDA0+1                 ;SETUP FIRST INVALID ADDRESS
3439 033576                         10$:
3440 033576  004737  054450                JSR     PC,SETVV        ;GO SET VOLUME VALID
     033602  000402                        BR      20$             ;BRANCH TO 20$ IF NO ERROR
     033604  104000                        EMT
3441 033606  000477                        BR      100$                    ;SKIP REST OF TEST
3442
3443                                ;CLEAR DESIRED CYLINDER, LOAD INVALID TRACK ADDRESS, AND SET
3444                                ;18 BIT FORMAT
3445 033610                         20$:
3446 033610  012760  000000  000034        MOV     #0,RMDC(R0)     ;LOAD RMDC
3447 033616  013760  001420  000006        MOV     RMDA0,RMDA(R0)  ;LOAD RMDA
3448 033624  012760  000000  000032        MOV     #0,RMOF(R0)     ;LOAD RMOF
3449
3450                                ;EXECUTE A SEARCH COMMAND TO WHERE "SET PULSE" IS ACTIVE
3451 033632  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3452 033640  012760  000031  000000        MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
3453 033646  012703  000002                MOV     #2,R3
3454 033652                         30$:
3455 033652  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3456 033660  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
```

```
3457 033666  005303                        DEC     R3
3458 033670  001370                        BNE     30$                     ;ISSUE 2 CLOCKS
3459
3460                                ;VERIFY IAE IS SET
3461 033672  016037  000014  001142        MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
3462 033700  042737  175777  001142        BIC     #^CIAE,$BDDAT
3463 033706  001015                        BNE     40$                     ;BRANCH IF IAE IS ON
3464 033710  012737  002000  001140        MOV     #IAE,$GDDAT             ;SETUP ERROR MESSAGE
3465 033716  010037  001136                MOV     R0,$BDADR
3466 033722  062737  000014  001136        ADD     #RMER1,$BDADR
3467 033730  013737  001420  001174        MOV     RMDA0,$TMP0
3468 033736  104210                        EMT     210
3469 033740  000422                        BR      100$
3470
3471                                ;ADVANCE TO NEXT RMDA ADDRESS
3472 033742                         40$:
3473 033742  105737  001421                TSTB    RMDA0+1                 ;TESTING INVALID SECTORS ?
3474 033746  001411                        BEQ     50$                     ;YES !.
3475 033750  105237  001421                INCB    RMDA0+1                 ;INCREMENT TRACK ADDRESS
3476 033754  123727  001421  000200        CMPB    RMDA0+1,#128.           ;DONE ?
3477 033762  101705                        BLOS    10$                     ;NO, TEST NEXT TRACK ADDRESS
3478 033764  012737  000035  001420        MOV     #29.,RMDA0              ;LOAD LOAD SECTOR ADDRESS AND
3479                                                                        ;TRACK 0.
3480 033772  005237  001420          50$:  INC     RMDA0                   ;INCREMENT SECTOR ADDRESS
3481 033776  123727  001420  000200        CMPB    RMDA0,#128.             ;DONE ?
3482 034004  101674                        BLOS    10$                     ;NO, TEST NEXT SECTOR ADDRESS
3483 034006                         100$:
3484
3485                                ;;*******************************************************************
                                    ;*TEST 71        INVALID CYLINDER TEST

                                    ;;*******************************************************************
     034006                         TST71:
     034006  000004                        SCOPE                           ;SCOPE CALL
     034010  000240                        NOP
     034012  012706  001100                MOV     #STACK,SP               ;LOAD THE STACK POINTER
     034016  013700  001276                MOV     $BASE,R0                ;R0 - UNIBUS ADDRESS
     034022  013701  001466                MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
     034026  012737  000071  001226        MOV     #71,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
3486
3487 034034  012737  001467  001446        MOV     #823.,RMDC0             ;SET FIRST INVALID CYLINDER
3488 034042                         10$:
3489 034042  004737  054450                JSR     PC,SETVV                ;GO SET VOLUME VALID
     034046  000402                        BR      20$                     ;BRANCH TO 20$ IF NO ERROR
     034050  104000                        EMT
3490 034052  000460                        BR      50$                     ;SKIP IF ERROR
3491 034054                         20$:
3492 034054  013760  001446  000034        MOV     RMDC0,RMDC(R0)          ;LOAD RMDC
3493 034062  012760  000000  000006        MOV     #0,RMDA(R0)             ;LOAD RMDA
3494
3495                                ;ENABLE DEBUG CLOCK AND EXECUTE SEARCH COMMAND
3496 034070  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3497 034076  012760  000031  000000        MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
3498 034104  012703  000002                MOV     #2,R3
3499 034110                         30$:
     034110  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3500 034116  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
```

```
3501 034124  005303                          DEC     R3
3502 034126  001370                          BNE     30$                     ;ISSUE 2 CLOCKS
3503
3504                                 ;VERIFY IAE IS SET
3505 034130  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
3506 034136  042737  175777  001142          BIC     #^CIAE,$BDDAT
3507 034144  001015                          BNE     40$                     ;BRANCH IF IAE IS SET
3508 034146  012737  002000  001140          MOV     #IAE,$GDDAT             ;SETUP ERROR MESSAGE
3509 034154  010037  001136                  MOV     R0,$BDADR
3510 034160  062737  000014  001136          ADD     #RMER1,$BDADR
3511 034166  013737  001446  001174          MOV     RMDCO,$TMP0
3512 034174  104211                          EMT     211
3513 034176  000406                          BR      50$
3514
3515                                 ;ADVANCE CYLINDER ADDRESS
3516 034200                          40$:
3517 034200  005237  001446                  INC     RMDCO           ;INCREMENT CYLINDER ADDRESS
3518 034204  023727  001446  002000          CMP     RMDCO,#1024.    ;DONE ?
3519 034212  103713                          BLO     10$             ;NO, TEST AGAIN
3520 034214                          50$:
3521
3522                                 ;;*****************************************************************
                                     ;*TEST 72        SET AOE TEST

                                     ;;*****************************************************************
     034214                          TST72:
     034214  000004                          SCOPE                   ;SCOPE CALL
     034216  000240                          NOP
     034220  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     034224  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     034230  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     034234  012737  000072  001226          MOV     #72,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3523
3524 034242  005037  001444                  CLR     RMOFO           ;18 BIT FORMAT MODE
3525 034246  013737  001334  001420          MOV     LSTRK,RMDAO     ;SETUP LAST TRACK AND
3526 034254  112737  000035  001420          MOVB    #29.,RMDAO      ;LAST SECTOR
3527
3528                                 ;ENABLE DEBUG CLOCK AND LOAD LAST SECTOR ADDRESS, MEMORY ADDRESS AND
3529                                 ;WORD COUNT, THEN LOAD WRITE DATA COMMAND WITH GO SET
3530 034262                          10$:
3531 034262  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     034266  000402                          BR      15$             ;BRANCH TO 15$ IF NO ERROR
     034270  104000                          EMT
3532 034272  000504                          BR      40$             ;SKIP TEST IF ERROR
3533 034274                          15$:
3534 034274  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3535 034302  013760  001444  000032          MOV     RMOFO,RMOF(R0)  ;LOAD RMOF
3536 034310  016037  000032  001174          MOV     RMOF(R0),$TMP0  ;STORE RMOF AT $TMP0
3537 034316  012760  001466  000034          MOV     #822.,RMDC(R0)  ;LOAD RMDC
3538 034324  013760  001420  000006          MOV     RMDAO,RMDA(R0)  ;LOAD RMDA
3539 034332  012760  177000  000002          MOV     #-512.,RMWC(R0) ;LOAD RMWC
3540 034340  012760  104312  000004          MOV     #BUFFER,RMBA(R0)        ;LOAD RMBA
3541 034346  012760  000061  000000          MOV     #WD!GO,RMCS1(R0)        ;LOAD RMCS1
3542 034354  012702  000014                  MOV     #14,R2          ;R2 = CLOCK COUNT
3543                                                                 ;CLOCK COUNT 2-05 3-04 14-60...
3544                                 ;CLOCK THE COMMAND SEQUENCER TO GENERATE SET PULSE
3545 034360                          20$:
```

```
3546 034360  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
3547 034366  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3548 034374  005302                          DEC     R2
3549 034376  001370                          BNE     20$             ;ISSUE 2 CLOCKS
3550
3551                                 ;FORCE EBL TO GET TO OVERFLOW ADDRESS
3552 034400  012760  061001  000024          MOV     #DMD!MUR!DBEN!DEBL,RMMR1(R0)    ;LOAD RMMR1
3553 034406  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3554
3555                                 ;VERIFY THAT ADDRESS OVERFLOW ERROR IS SET
3556 034414  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
3557 034422  042737  176777  001142          BIC     #^CAOE,$BDDAT
3558 034430  001012                          BNE     30$             ;BRANCH IF AOE IS SET
3559 034432  010037  001136                  MOV     R0,$BDADR       ;SETUP ERROR MESSAGE
3560 034436  062737  000014  001136          ADD     #RMER1,$BDADR
3561 034444  012737  001000  001140          MOV     #AOE,$GDDAT
3562 034452  104212                          EMT     212
3563 034454  000413                          BR      40$
3564 034456                          30$:                            ;END OF TEST
3565 034456  032737  010000  001444          BIT     #FMT16,RMOFO    ;DONE 16 BIT FORMAT TEST ?
3566 034464  001007                          BNE     40$             ;YES !.
3567 034466  012737  010000  001444          MOV     #FMT16,RMOFO    ;SET 16 BIT FORMAT MODE AND
3568 034474  112737  000037  001420          MOVB    #31.,RMDAO      ;LAST SECTOR
3569 034502  000667                          BR      10$
3570 034504                          40$:
3571
3572                                 ;;*****************************************************************
                                     ;*TEST 73        SET RMR TEST

                                     ;;*****************************************************************
     034504                          TST73:
     034504  000004                          SCOPE                   ;SCOPE CALL
     034506  000240                          NOP
     034510  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     034514  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     034520  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     034524  012737  000073  001226          MOV     #73,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3573
3574 034532  010037  001136                  MOV     R0,$BDADR       ;SETUP REGISTER ADDRESS FOR MSG
3575 034536  062737  000014  001136          ADD     #RMER1,$BDADR
3576 034544  012702  034712                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
3577
3578                                 ;CLEAR THE DEVICE AND ENABLE DEBUG CLOCK
3579 034550                          10$:
3580 034550  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3581 034554  012760  000001  000024          MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3582 034562  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3583 034570  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
3584 034576  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
3585
3586                                 ;SET GO THEN WRITE THE REGISTER SPECIFIED BY THE TABLE
3587 034604  012760  000001  000000          MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
3588 034612  011203                          MOV     (R2),R3                 ;GENERATE REGISTER ADDRESS
3589 034614  060003                          ADD     R0,R3
3590 034616  012713  041001                  MOV     #DMD!MUR!DBEN,(R3)      ;WRITE THE REGISTER
3591
3592                                 ;VERIFY RMR ACCORDING TO TABLE
```

```
3593 034622 016037 000014 001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
3594 034630 042737 177773 001142          BIC     #^C<RMR>,$BDDAT
3595 034636 016237 000002 001140          MOV     2(R2),$GDDAT            ;GET EXPECTED RESULT FROM TABLE
3596 034644 023737 001140 001142          CMP     $GDDAT,$BDDAT
3597 034652 001411                        BEQ     30$                     ;BRANCH IF RMR IS OK
3598 034654 011237 001174                 MOV     (R2),$TMP0              ;SAVE TEST REGISTER
3599 034660 032762 000004 000002          BIT     #RMR,2(R2)
3600 034666 001002                        BNE     20$                     ;BRANCH IF ERROR SHOULD BE ONE
3601 034670 104213                        EMT     213
3602 034672 000401                        BR      30$
3603 034674                         20$:
     034674 104214                        EMT     214
3604
3605                                ;ADVANCE TABLE POINTER, EXIT IF DONE
3606 034676                         30$:
3607 034676 062702 000004                 ADD     #4,R2
3608 034702 005712                        TST     (R2)
3609 034704 100401                        BMI     40$                     ;EXIT IF ENTRY NEGATIVE
3610 034706 000720                        BR      10$
3611 034710 000410                  40$:  BR      200$                    ;JUMP OVER TABLE
3612
3613                                ;TABLE OF REGISTER ADDRESSES AND RMR VALUES
3614 034712                         100$:
3615 034712 000016                        .WORD   RMAS                    ;ATTENTION SUMMARY REG
3616 034714 000000                        .WORD   0                       ;RMR = 0
3617
3618 034716 000024                        .WORD   RMMR1                   ;MAINTENANCE REG
3619 034720 000000                        .WORD   0                       ;RMR = 0
3620
3621 034722 000006                        .WORD   RMDA                    ;DISK ADDRESS REG
3622 034724 000004                        .WORD   RMR                     ;RMR - 1
3623
3624 034726 177777                        .WORD   -1                      ;END OF TABLE
3625 034730 000000                        .WORD   0
3626
3627 034732                         200$:                                 ;END OF TEST
3628
3629                                ;;*************************************************** **********
                                    ;*TEST 74        PGM STATUS CHECK
                                    ;;*************************************************** **********
     034732                         TST74:
     034732 000004                        SCOPE                          ;SCOPE CALL
     034734 000240                        NOP
     034736 012706 001100                 MOV     #STACK,SP              ;LOAD THE STACK POINTER
     034742 013700 001276                 MOV     $BASE,R0               ;R0 = UNIBUS ADDRESS
     034746 013701 001466                 MOV     TSTQUE,R1              ;R1 = POINTER TO DEVICE
     034752 012737 000074 001226          MOV     #74,$TESTN             ;;SET TEST NUMBER IN APT MAIL BOX
3630
3631                                ;CLEAR AND READ DRIVE TYPE AND DRIVE STATUS
3632 034760 004737 054674                 JSR     PC,CN1CLR              ;GO CLEAR CONTROLLER
3633 034764 016037 000026 001174          MOV     RMDT(R0),$TMP0         ;STORE RMDT AT $TMP0
3634 034772 016037 000012 001142          MOV     RMDS(R0),$BDDAT        ;STORE RMDS AT $BDDAT
3635
3636                                ;OMIT TEST IF DRQ IS ON - ELSE VERIFY THAT PGM IS OFF
3637 035000 032737 004000 001174          BIT     #DRQ,$TMP0
3638 035006 001014                        BNE     10$                    ;BRANCH IF DRQ IS ON
```

```
3639 035010  042737  176777  001142          BIC     #^CPGM,$BDDAT
3640 035016  001410                          BEQ     10$                     ;BRANCH IF PGM IS OFF
3641 035020  005037  001140                  CLR     $GDDAT
3642 035024  010037  001136                  MOV     R0,$BDADR
3643 035030  062737  000012  001134          ADD     #RMDS,$GDADR
3644 035036  104215                          EMT     215
3645 035040                          10$:                                    ;END OF TEST
3646
3647                                 ;;*******************************************************************
                                     ;*TEST 75        DVA/DPR STATUS CHECK
                                     ;;*******************************************************************
     035040                          TST75:
     035040  000004                          SCOPE                           ;SCOPE CALL
     035042  000240                          NOP
     035044  012706  001100                  MOV     #STACK,SP               ;LOAD THE STACK POINTER
     035050  013700  001276                  MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
     035054  013701  001466                  MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
     035060  012737  000075  001226          MOV     #75,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
3648
3649                                 ;CLEAR AND VERIFY THAT DVA IS SET
3650 035066  004737  054674                  JSR     PC,CNTCLR               ;GO CLEAR CONTROLLER
3651 035072  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
3652 035100  042737  173777  001142          BIC     #^CDVA,$BDDAT
3653 035106  001006                          BNE     10$                     ;BRANCH IF DVA IS ON
3654 035110  012737  004000  001140          MOV     #DVA,$GDDAT
3655 035116  010037  001136                  MOV     R0,$BDADR
3656 035122  104216                          EMT     216
3657
3658                                 ;VERIFY THAT DPR IS SET
3659 035124                          10$:
3660 035124  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3661 035132  042737  177377  001142          BIC     #^CDPR,$BDDAT
3662 035140  001011                          BNE     20$
3663 035142  012737  000400  001140          MOV     #DPR,$GDDAT
3664 035150  010037  001136                  MOV     R0,$BDADR
3665 035154  062737  000012  001136          ADD     #RMDS,$BDADR
3666 035162  104217                          EMT     217
3667
3668 035164                          20$:                                    ;END OF TEST
3669
3670                                 ;;*******************************************************************
                                     ;*TEST 76        PORT REQUEST TEST, PART 1
                                     ;;*******************************************************************
     035164                          TST76:
     035164  000004                          SCOPE                           ;SCOPE CALL
     035166  000240                          NOP
     035170  012706  001100                  MOV     #STACK,SP               ;LOAD THE STACK POINTER
     035174  013700  001276                  MOV     $BASE,R0                ;R0 = UNIBUS ADDRESS
     035200  013701  001466                  MOV     TSTQUE,R1               ;R1 = POINTER TO DEVICE
     035204  012737  000076  001226          MOV     #76,$TESTN              ;;SET TEST NUMBER IN APT MAIL BOX
3671
3672 035212  004737  054450                  JSR     PC,SETVV                ;GO SET VOLUME VALID
     035216  000402                          BR      10$                     ;BRANCH TO 10$ IF NO ERROR
     035220  104000                          EMT
3673 035222  000434                          BR      20$
```

```
3674
3675                                    ;EXECUTE A RELEASE TO RESET REQUEST FLOP
3676 035224                             10$:
3677 035224  012760  000013  000000            MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
3678
3679                                    ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
3680 035232  016002  000040                    MOV     RMMR2(R0),R2    ;STORE RMMR2 AT R2
3681 035236  042702  037777                    BIC     #^C<RQA!RQB>,R2
3682 035242  001024                            BNE     20$
3683
3684                                    ;READ RMCS1 TO SET REQUEST FLOP
3685 035244  016002  000000                    MOV     RMCS1(R0),R2    ;STORE RMCS1 AT R2
3686
3687                                    ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
3688 035250  016005  000012                    MOV     RMDS(R0),R5     ;STORE RMDS AT R5
3689 035254  032705  001000                    BIT     #PGM,R5         ;SEE IF PGM IS SET
3690 035260  001415                            BEQ     20$             ;DONT TEST REQUEST IF PGM IS ZERO
3691 035262  016037  000040  001142            MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
3692 035270  042737  037777  001142            BIC     #^C<RQA.RQB>,$BDDAT
3693 035276  001006                            BNE     20$                     ;BRANCH IF REQUEST IS SET
3694 035300  010037  001136                    MOV     R0,$BDADR
3695 035304  062737  000040  001136            ADD     #RMMR2,$BDADR
3696 035312  104220                            EMT     220
3697 035314                             20$:
3698
3699                                    ;;*****************************************************************
                                        ;*TEST 77        PORT REQUEST TEST, PART 2

                                        ;;*****************************************************************
     035314                             TST77:
     035314  000004                            SCOPE                   ;SCOPE CALL
     035316  000240                            NOP
     035320  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
     035324  013700  001276                    MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     035330  013701  001466                    MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     035334  012737  000077  001226            MOV     #77,$TESTN      ;;SET TEST NUMBER IN APT MAIL BOX
3700
3701 035342  004737  054450                    JSR     PC,SETVV        ;GO SET VOLUME VALID
     035346  000402                            BR      10$             ;BRANCH TO 10$ IF NO ERROR
     035350  104000                            EMT
3702 035352  000435                            BR      20$
3703
3704                                    ;EXECUTE A RELEASE TO RESET REQUEST FLOP
3705 035354                             10$:
3706 035354  012760  000013  000000            MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
3707
3708                                    ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
3709 035362  016002  000040                    MOV     RMMR2(R0),R2    ;STORE RMMR2 AT R2
3710 035366  042702  037777                    BIC     #^C<RQA!RQB>,R2
3711 035372  001025                            BNE     20$
3712
3713                                    ;WRITE THE ATTENTION SUMMARY REGISTER TO SET REQUEST FLOP
3714 035374  012760  177777  000016            MOV     #-1,RMAS(R0)    ;LOAD RMAS
3715
3716                                    ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
3717 035402  016005  000012                    MOV     RMDS(R0),R5     ;STORE RMDS AT R5
3718 035406  032705  001000                    BIT     #PGM,R5         ;SEE IF PGM IS SET
```

```
3719 035412  001415                          BEQ     20$             ;DONT TEST REQUEST IF PGM IS ZERO
3720 035414  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
3721 035422  042737  037777  001142          BIC     #^C<RQA!RQB>,$BDDAT
3722 035430  001006                          BNE     20$
3723 035432  010037  001136                  MOV     R0,$BDADR
3724 035436  062737  000040  001136          ADD     #RMMR2,$BDADR
3725 035444  104221                          EMT     221
3726 035446                          20$:
3727
3728                                 ;;*****************************************************************
                                     ;*TEST 100       PORT REQUEST TEST, PART 3

                                     ;;*****************************************************************
     035446                          TST100:
     035446  000004                          SCOPE                   ;SCOPE CALL
     035450  000240                          NOP
     035452  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     035456  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     035462  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     035466  012737  000100  001226          MOV     #100,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
3729
3730 035474  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     035500  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
     035502  104000                          EMT
3731 035504  000435                          BR      20$
3732
3733                                 ;EXECUTE A RELEASE COMMAND TO RESET REQUEST FLOP
3734 035506                          10$:
3735 035506  012760  000013  000000          MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
3736
3737                                 ;READ RMMR2 AND SKIP TEST IF REQUEST FLOPS ARENT RESET
3738 035514  016002  000040                  MOV     RMMR2(R0),R2    ;STORE RMMR2 AT R2
3739 035520  042702  037777                  BIC     #^C<RQA!RQB>,R2
3740 035524  001025                          BNE     20$
3741
3742                                 ;WRITE RMDA TO SET REQUEST FLOP
3743 035526  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
3744
3745                                 ;VERIFY THAT REQUEST FLOP IS SET IF PGM IS SET
3746 035534  016005  000012                  MOV     RMDS(R0),R5     ;STORE RMDS AT R5
3747 035540  032705  001000                  BIT     #PGM,R5         ;SEE IF PGM IS SET
3748 035544  001415                          BEQ     20$             ;DONT TEST REQUEST IF PGM IS ZERO
3749 035546  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
3750 035554  042737  037777  001142          BIC     #^C<RQA!RQB>,$BDDAT
3751 035562  001006                          BNE     20$
3752 035564  010037  001136                  MOV     R0,$BDADR
3753 035570  062737  000040  001136          ADD     #RMMR2,$BDADR
3754 035576  104222                          EMT     222
3755
3756 035600                          20$:
3757
3758                                 ;;*****************************************************************
                                     ;*TEST 101       RELEASE TEST

                                     ;;*****************************************************************
     035600                          TST101:
     035600  000004                          SCOPE                   ;SCOPE CALL
```

```
                035602  000240                          NOP
                035604  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
                035610  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                035614  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
                035620  012737  000101  001226          MOV     #101,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
    3759
    3760                                        ;REQUEST FLOP SHOULD SET WHEN WRITTING RMCS1
    3761
    3762        035626  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
                035632  000402                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
                035634  104000                          EMT
    3763        035636  000454                          BR      40$
    3764
    3765                                        ;EXECUTE A RELEASE COMMAND
    3766        035640                          10$:
    3767        035640  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
    3768        035646  012760  000013  000000          MOV     #RLEASE!GO,RMCS1(R0)    ;LOAD RMCS1
    3769        035654  012702  000002                  MOV     #2,R2
    3770        035660                          20$:
                035660  012760  141001  000024          MOV     #DMD!DBEN.MUR!DBCK,RMMR1(R0)    ;LOAD RMMR1
    3771        035666  012760  041001  000024          MOV     #DMD!DBEN!MUR,RMMR1(R0) ;LOAD RMMR1
    3772        035674  005302                          DEC     R2
    3773        035676  001370                          BNE     20$                     ;ISSUE 2 CLOCKS
    3774
    3775                                        ;VERIFY REQUEST FLOPS ARE RESET
    3776        035700  016037  000026  001174          MOV     RMDT(R0),$TMP0  ;STORE RMDT AT $TMP0
    3777        035706  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
    3778        035714  042737  037777  001142          BIC     #^C<RQA.RQB>,$BDDAT
    3779        035722  001422                          BEQ     40$                     ;BRANCH IF REQUESTS ARE RESET
    3780        035724  032737  004000  001174          BIT     #DRQ,$TMP0
    3781        035732  001410                          BEQ     30$                     ;BRANCH IF SINGLE PORT DEVICE
    3782        035734  032737  100000  001142          BIT     #RQA,$BDDAT
    3783        035742  001412                          BEQ     40$                     ;BRANCH IF RQA IS RESET
    3784        035744  032737  040000  001142          BIT     #RQB,$BDDAT
    3785        035752  001406                          BEQ     40$                     ;BRANCH IF RQB IS RESET
    3786
    3787                                        ;DRQ IS ZERO AND A REQUEST FLOP IS ON, OR, DRQ IS ONE AND
    3788                                        ;BOTH REQUEST FLOPS ARE ON
    3789        035754                          30$:
    3790        035754  010037  001136                  MOV     R0,$BDADR
    3791        035760  062737  000040  001136          ADD     #RMMR2,$BDADR
    3792        035766  104223                          EMT     223
    3793        035770                          40$:                            ;END OF TEST
    3794
    3795                                        ;;*****************************************************************
                                                ;*TEST 102       WRITE ATA TEST

                                                ;;*****************************************************************
                035770                          TST102:
                035770  000004                          SCOPE                   ;SCOPE CALL
                035772  000240                          NOP
                035774  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
                036000  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
                036004  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
                036010  012737  000102  001226          MOV     #102,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
    3796
    3797                                        ;CLEAR THE DEVICE, SET DIAGNOSTIC MODE THEN SET UNIT READY
```

```
3798 036016  004737  054674                 JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3799 036022  012760  000001  000024         MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3800 036030  012760  000000  000014         MOV     #0,RMER1(R0)    ;LOAD RMER1
3801 036036  012760  000000  000042         MOV     #0,RMER2(R0)    ;LOAD RMER2
3802 036044  012760  001001  000024         MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
3803
3804                                 ;WRITE THE ATTENTION SUMMARY REGISTER
3805 036052  111102                         MOVB    (R1),R2                 ;ASSEMBLE THE ATA BIT IN R3
3806 036054  042702  177770                 BIC     #^CUNTMSK,R2
3807 036060  116203  063516                 MOVB    ATNTBL(R2),R3
3808 036064  042703  177400                 BIC     #^CATNMSK,R3
3809 036070  010360  000016                 MOV     R3,RMAS(R0)     ;LOAD RMAS
3810
3811                                 ;READ RMDS AND VERIFY THAT ATA IS RESET
3812 036074  016037  000012  001142         MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3813 036102  042737  077777  001142         BIC     #^CATA,$BDDAT
3814 036110  001411                         BEQ     10$                     ;BRANCH IF ATA IS RESET
3815 036112  010037  001136                 MOV     R0,$BDADR
3816 036116  062737  000012  001136         ADD     #RMDS,$BDADR
3817 036124  005037  001140                 CLR     $GDDAT
3818 036130  104224                         EMT     224
3819 036132  000417                         BR      20$
3820
3821                                 ;READ RMAS AND VERIFY ATA IS RESET
3822 036134                         10$:
3823 036134  016037  000016  001142         MOV     RMAS(R0),$BDDAT ;STORE RMAS AT $BDDAT
3824 036142  005103                         COM     R3
3825 036144  040337  001142                 BIC     R3,$BDDAT
3826 036150  001410                         BEQ     20$                     ;BRANCH IF ATA IS RESET
3827 036152  005037  001140                 CLR     $GDDAT
3828 036156  010037  001136                 MOV     R0,$BDADR
3829 036162  062737  000016  001136         ADD     #RMAS,$BDADR
3830 036170  104225                         EMT     225
3831 036172                         20$:                                    ;END OF TEST
3832
3833                                 ;;*******************************************************************
                                    ;*TEST 103       RESET ATA BY GO TEST

                                    ;;*******************************************************************
     036172                         TST103:
     036172  000004                         SCOPE                   ;SCOPE CALL
     036174  000240                         NOP
     036176  012706  001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
     036202  013700  001276                 MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     036206  013701  001466                 MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     036212  012737  000103  001226         MOV     #103,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
3834
3835 036220  004737  054674                 JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3836 036224  012760  000001  000024         MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3837 036232  012760  000000  000014         MOV     #0,RMER1(R0)    ;LOAD RMER1
3838 036240  012760  000000  000042         MOV     #0,RMER2(R0)    ;LOAD RMER2
3839 036246  012760  041001  000024         MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3840
3841                                 ;WITH DEBUG CLOCK ENABLED, SET GO AND VERIFY THAT ATA IS RESET
3842 036254  012760  000001  000000         MOV     #GO,RMCS1(R0)   ;LOAD RMCS1
3843 036262  016037  000012  001142         MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3844 036270  042737  077777  001142         BIC     #^CATA,$BDDAT
```

```
3345 036276  001410                        BEQ     10$                     ;BRANCH IF ATA IS RESET
3846 036300  005037  001140                CLR     $GDDAT
3847 036304  010037  001136                MOV     R0,$BDADR
3848 036310  062737  000012  001136        ADD     #RMDS,$BDADR
3849 036316  104226                        EMT     226
3850
3851 036320                        10$:                                    ;END OF TEST
3852
3853                               ;;*******************************************************************
                                   ;*TEST 104      UNIT READY ATA TEST

                                   ;;*******************************************************************
     036320                        TST104:
     036320  000004                       SCOPE                   ;SCOPE CALL
     036322  000240                       NOP
     036324  012706  001100               MOV     #STACK,SP       ;LOAD THE STACK POINTER
     036330  013700  001276               MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
     036334  013701  001466               MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
     036340  012737  000104  001226       MOV     #104,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
3854
3855                               ;SET DIAGNOSTIC MODE AND CLEAR ATA
3856 036346  004737  054674               JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
3857 036352  012760  000001  000024       MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3858 036360  012760  000000  000014       MOV     #0,RMER1(R0)    ;LOAD RMER1
3859 036366  012760  000000  000042       MOV     #0,RMER2(R0)    ;LOAD RMER2
3860 036374  012760  177777  000016       MOV     #-1,RMAS(R0)    ;LOAD RMAS
3861
3862                               ;SET UNIT READY AND VERIFY ATA IS SET
3863 036402  012760  001001  000024       MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
3864 036410  016037  000012  001142       MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3865 036416  042737  077777  001142       BIC     #^CATA,$BDDAT
3866 036424  001011                       BNE     10$                     ;BRANCH IF ATA IS SET
3867 036426  010037  001136               MOV     R0,$BDADR
3868 036432  062737  000012  001136       ADD     #RMDS,$BDADR
3869 036440  012737  100000  001140       MOV     #ATA,$GDDAT
3870 036446  104227                       EMT     227
3871
3872                               ;CLEAR ATA, RESET UNIT READY, AND VERIFY ATA IS SET
3873 036450                        10$:
3874 036450  012760  177777  000016       MOV     #-1,RMAS(R0)    ;LOAD RMAS
3875 036456  012760  000001  000024       MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
3876 036464  016037  000012  001142       MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
3877 036472  042737  077777  001142       BIC     #^CATA,$BDDAT
3878 036500  001011                       BNE     20$
3879 036502  010037  001136               MOV     R0,$BDADR
3880 036506  062737  000012  001136       ADD     #RMDS,$BDADR
3881 036514  012737  100000  001140       MOV     #ATA,$GDDAT
3882 036522  104230                       EMT     230
3883
3884 036524                        20$:                                    ;END OF TEST
3885
3886                               ;;*******************************************************************
                                   ;*TEST 105      ERROR ATA TEST

                                   ;;*******************************************************************
     036524                        TST105:
     036524  000004                       SCOPE                   ;SCOPE CALL
```

```
     036526  000240                                  NOP
     036530  012706  001100                          MOV     #STACK,SP         ;LOAD THE STACK POINTER
     036534  013700  001276                          MOV     $BASE,R0          ;R0 = UNIBUS ADDRESS
     036540  013701  001466                          MOV     TSTQUE,R1         ;R1 = POINTER TO DEVICE
     036544  012737  000105  001226                  MOV     #105,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

                                             ;CLEAR THE DEVICE AND RESET ATTENTION
     036552  004737  054674                          JSR     PC,CNTCLR         ;GO CLEAR CONTROLLER
     036556  012760  000001  000024                  MOV     #DMD,RMMR1(R0)    ;LOAD RMMR1
     036564  012760  000000  000014                  MOV     #0,RMER1(R0)      ;LOAD RMER1
     036572  012760  000000  000042                  MOV     #0,RMER2(R0)      ;LOAD RMER2
     036600  012760  177777  000016                  MOV     #-1,RMAS(R0)      ;LOAD RMAS

                                             ;WRITE ONES IN ERROR REGISTER 1 AND VERIFY ATA IS SET
     036606  012760  177777  000014                  MOV     #-1,RMER1(R0)     ;LOAD RMER1
     036614  016037  000012  001142                  MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
     036622  042737  077777  001142                  BIC     #^CATA,$BDDAT
     036630  001011                                  BNE     10$
     036632  012737  100000  001140                  MOV     #ATA,$GDDAT
     036640  010037  001136                          MOV     R0,$BDADR
     036644  062737  000012  001136                  ADD     #RMDS,$BDADR
     036652  104231                                  EMT     231

     036654                                  10$:                                        ;END OF TEST


                                             ;;*****************************************************************
                                             ;*TEST 106          REGISTER TRANSFER ATA TEST

                                             ;;*****************************************************************
     036654                                  TST106:
     036654  000004                                  SCOPE                     ;SCOPE CALL
     036656  000240                                  NOP
     036660  012706  001100                          MOV     #STACK,SP         ;LOAD THE STACK POINTER
     036664  013700  001276                          MOV     $BASE,R0          ;R0 = UNIBUS ADDRESS
     036670  013701  001466                          MOV     TSTQUE,R1         ;R1 = POINTER TO DEVICE
     036674  012737  000106  001226                  MOV     #106,$TESTN       ;;SET TEST NUMBER IN APT MAIL BOX

     036702  012702  037030                          MOV     #100$,R2                    ;INITIALIZE TABLE ADDRESS


                                             ;FORCE COMPOSITE ERROR AND RESET ATA
     036706                                  5$:
     036706  004737  054674                          JSR     PC,CNTCLR         ;GO CLEAR CONTROLLER
     036712  012760  000001  000024                  MOV     #DMD,RMMR1(R0)    ;LOAD RMMR1
     036720  012760  177777  000014                  MOV     #-1,RMER1(R0)     ;LOAD RMER1
     036726  012760  177777  000016                  MOV     #-1,RMAS(R0)      ;LOAD RMAS

                                             ;WRITE THE TEST REGISTER AND VERIFY ATA
     036734  011203                                  MOV     (R2),R3                     ;GENERATE REGISTER ADDRESS
     036736  060003                                  ADD     R0,R3
     036740  005013                                  CLR     (R3)
     036742  016037  000012  001142                  MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
     036750  042737  077777  001142                  BIC     #^CATA,$BDDAT
     036756  016237  000002  001140                  MOV     2(R2),$GDDAT
     036764  023737  001140  001142                  CMP     $GDDAT,$BDDAT
     036772  001410                                  BEQ     10$                         ;BRANCH IF ATA IS OK
     036774  010337  001174                          MOV     R3,$TMP0
     037000  010037  001136                          MOV     R0,$BDADR
```

```
3929 037004  062737  000012  001136          ADD     #RMDS,$BDADR
3930 037012  104232                          EMT     232
3931
3932                                 ;MOVE TABLE POINTER - EXIT IF DONE
3933 037014                          10$:
3934 037014  062702  000004                  ADD     #4,R2
3935 037020  005712                          TST     (R2)
3936 037022  100401                          BMI     20$                     ;EXIT IF ENTRY MINUS
3937 037024  000730                          BR      5$
3938 037026  000410                  20$:    BR      200$                    ;JUMP OVER TABLE
3939
3940                                 ;TABLE OF REGISTER ADDRESSES AND ATA BITS
3941 037030                          100$:
3942 037030  000016                          .WORD   RMAS
3943 037032  000000                          .WORD
3944
3945 037034  000006                          .WORD   RMDA
3946 037036  100000                          .WORD   ATA
3947
3948 037040  000000                          .WORD   RMCS1
3949 037042  000000                          .WORD
3950
3951 037044  177777                          .WORD   -1                      ;END OF TABLE
3952 037046  000000                          .WORD
3953
3954 037050                          200$:                                   ;END OF TEST
3955
3956                                 ;;*****************************************************************
                                     ;*TEST 107       P SET ATA TEST
                                     ;;*****************************************************************
     037050                          TST107:
     037050  000004                          SCOPE                   ;SCOPE CALL
     037052  000240                          NOP
     037054  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     037060  013700  001276                  MOV     $BASE,R0        ;R0   UNIBUS ADDRESS
     037064  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     037070  012737  000107  001226          MOV     #107,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
3957
3958 037076  012702  037240                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
3959 037102                          10$:
3960 037102  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     037106  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
     037110  104000                          EMT
3961 037112  000451                          BR      50$
3962
3963                                 ;EXECUTE THE COMMAND FROM THE TABLE
3964 037114                          20$:
3965 037114  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
3966 037122  011203                          MOV     (R2),R3         ;GET FUNCTION CODE
3967 037124  052703  000001                  BIS     #GO,R3
3968 037130  010360  000000                  MOV     R3,RMCS1(R0)    ;LOAD RMCS1
3969 037134  012703  000003                  MOV     #3,R3
3970 037140                          30$:
3971 037140  012760  141001  000024          MOV     #DMD.MUR.DBEN.DBCK,RMMR1(R0)    ;LOAD RMMR1
3972 037146  012760  041001  000024          MOV     #DMD.MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
3973 037154  005303                          DEC     R3
```

```
3974 037156  001370                          BNE     30$
3975
3976                                 ;VERIFY THAT ATA IS SET
3977 037160  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDA
3978 037166  042737  077777  001142          BIC     #^CATA,$BDDAT
3979 037174  001013                          BNE     40$
3980 037176  010037  001136                  MOV     R0,$BDADR
3981 037202  062737  000012  001136          ADD     #RMDS,$BDADR
3982 037210  012737  100000  001140          MOV     #ATA,$GDDAT
3983 037216  011237  001174                  MOV     (R2),$TMP0
3984 037222  104233                         EMT     233
3985
3986                                 ;ADVANCE TABLE POINTER-EXIT IF DONE
3987 037224                          40$::
3988 037224  062702  000002                  ADD     #2,R2
3989 037230  005712                          TST     (R2)
3990 037232  100401                          BMI     50$
3991 037234  000722                          BR      10$
3992 037236  000403                 50$:    BR      200$            ;JUMP OVER TABLE
3993
3994                                 ;TABLE OF FUNCTION CODES
3995 037240                          100$:
3996 037240  000014                          .WORD   OFFSET
3997 037242  000016                          .WORD   RTC
3998 037244  177777                          .WORD   -1              ;END OF TABLE
3999 037246                          200$:
4000
4001                                 ;;*****************************************************************
                                     ;*TEST 110       SET WLE TEST
                                     ;;*****************************************************************
     037246                          TST110:
     037246  000004                          SCOPE                   ;SCOPE CALL
     037250  000240                          NOP
     037252  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     037256  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     037262  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     037266  012737  000110  001226          MOV     #110,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
4002
4003 037274  012702  037470                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
4004 037300                          10$:
4005 037300  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     037304  000402                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
     037306  104000                          EMT
4006 037310  000466                          BR      50$
4007
4008                                 ;ENABLE DEBUG CLOCK AND SET WRITE PROTECT ACCORDING TO TABLE
4009 037312                          20$:
4010 037312  016203  000002                  MOV     2(R2),R3        ;GET WRITE PROTECT FROM TABLE
4011 037316  052703  041001                  BIS     #DMD!MUR.DBEN,R3        ;SET OTHER MAINT BITS
4012 037322  010360  000024                  MOV     R3,RMMR1(R0)    ;LOAD RMMR1
4013
4014                                 ;LOAD AND EXECUTE THE COMMAND FROM THE TABLE
4015 037326  011204                          MOV     (R2),R4         ;GET FUNCTION CODE FROM TABLE
4016 037330  052704  000001                  BIS     #GO,R4          ;SET GO
4017 037334  010460  000000                  MOV     R4,RMCS1(R0)    ;LOAD RMCS1
4018 037340  012705  000002                  MOV     #2,R5           ;R5-CLOCK COUNT
```

SEQ 0186

```
4019 037344  052703  100000          30$:    BIS     #DBCK,R3        ;SET CLOCK
4020 037350  010360  000024                  MOV     R3,RMMR1(R0)    ;LOAD RMMR1
4021 037354  042703  100000                  BIC     #DBCK,R3        ;RESET CLOCK
4022 037360  010360  000024                  MOV     R3,RMMR1(R0)    ;LOAD RMMR1
4023 037364  005305                          DEC     R5
4024 037366  001366                          BNE     30$             ;ISSUE 2 CLOCKS
4025
4026                                 ;VERIFY THAT WRITE LOCK ERROR IS ACCORDING TO TABLE
4027 037370  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
4028 037376  042737  173777  001142          BIC     #^CWLE,$BDDAT
4029 037404  026237  000004  001142          CMP     4(R2),$BDDAT
4030 037412  001417                          BEQ     40$             ;BRANCH IF WLE IS OK
4031 037414  016237  000004  001140          MOV     4(R2),$GDDAT    ;SAVE DATA FOR ERROR MSG
4032 037422  010037  001136                  MOV     R0,$BDADR
4033 037426  062737  000014  001136          ADD     #RMER1,$BDADR
4034 037434  011237  001174                  MOV     (R2),$TMP0      ;$TMP0=FUNCTION CODE
4035 037440  016237  000002  001176          MOV     2(R2),$TMP1     ;$TMP1=WRITE PROTECT
4036 037446  104234                          EMT     234
4037 037450  000406                          BR      50$
4038
4039                                 ;ADVANCE TABLE POINTER TO NEXT FUNCTION CODE-EXIT IF DONE
4040 037452                          40$:
4041 037452  062702  000006                  ADD     #6,R2
4042 037456  005762  000002                  TST     2(R2)
4043 037462  100401                          BMI     50$
4044 037464  000705                          BR      10$             ;REPEAT TEST
4045
4046 037466  000416                  50$:    BR      200$            ;JUMP OVER TABLE
4047
4048                                 ;TABLE OF FUNCTION CODES, WRITE PROTECT AND WRITE LOCK ERRORS
4049 037470                          100$:
4050 037470  000060                          .WORD   WD              ;WRITE DATA COMMAND
4051 037472  000010                          .WORD   MWP             ;WRITE PROTECT ON
4052 037474  004000                          .WORD   WLE             ;WRITE LOCK ERROR ONE
4053
4054 037476  000060                          .WORD   WD              ;WRITE DATA COMMAND
4055 037500  000000                          .WORD                   ;MWP OFF
4056 037502  000000                          .WORD                   ;WLE OFF
4057
4058 037504  000070                          .WORD   RD              ;READ DATA COMMAND
4059 037506  000010                          .WORD   MWP             ;MWP ON
4060 037510  000000                          .WORD                   ;WLE OFF
4061
4062 037512  000020                          .WORD   RIP             ;READ IN PRESET COMMAND
4063 037514  000010                          .WORD   MWP             ;MWP ON
4064 037516  000000                          .WORD                   ;WLE OFF
4065
4066 037520  000000                          .WORD                   ;END OF TABLE
4067 037522  177777                          .WORD   -1
4068
4069 037524                          200$:                           ;END OF TEST
4070
4071                                 ;;*******************************************************************
                                     ;*TEST 111       EXCEPTION TEST

                                     ;;*******************************************************************
037524                               TST111:
```

```
        037524  000004                          SCOPE                   ;SCOPE CALL
        037526  000240                          NOP
        037530  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
        037534  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
        037540  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
        037544  012737  000111  001226          MOV     #111,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 4072
 4073                                   ;WITH OCCUPIED SET, EACH OF THE FOLLOWING ERRORS SHOULD CAUSE AN
 4074                                   ;EXCEPTION:
 4075                                   ;
 4076                                   ;       PAR, IF RUN AND GO IS SET
 4077                                   ;       RMR, IF RUN AND GO IS SET
 4078                                   ;       ILR, IF RUN AND GO IS SET
 4079                                   ;       DCK
 4080                                   ;       HCE
 4081                                   ;       HCRC
 4082                                   ;       FER
 4083                                   ;       OPI
 4084
 4085 037552  012702  040024                  MOV     #100$,R2        ;INITIALIZE TABLE POINTER
 4086
 4087                                   ;INITIALIZE AND VERIFY THAT EXCEPTION IS RESET
 4088 037556                            10$:
 4089 037556  004737  054674                  JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 4090 037562  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 4091 037570  042737  167777  001142          BIC     #^CREX,$BDDAT
 4092 037576  001410                          BEQ     20$             ;BRANCH IF EXCEPTION IS 0
 4093 037600  010037  001136                  MOV     R0,$BDADR
 4094 037604  062737  000024  001136          ADD     #RMMR1,$BDADR
 4095 037612  005037  001140                  CLR     $GDDAT
 4096 037616  104235                          EMT     235
 4097 037620                            20$:
 4098 037620  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
        037624  000402                          BR      30$             ;BRANCH TO 30$ IF NO ERROR
        037626  104000                          EMT
 4099 037630  000474                          BR      60$
 4100
 4101                                   ;EXECUTE A WRITE DATA COMMAND TO SET OCCUPIED, RUN AND GO
 4102 037632                            30$:
 4103 037632  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 4104 037640  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 4105 037646  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4106 037654  012760  000061  000000          MOV     #WD!GO,RMCS1(R0)        ;LOAD RMCS1
 4107 037662  012703  000002                  MOV     #2,R3
 4108 037666                            40$:
        037666  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 4109 037674  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4110 037702  005303                          DEC     R3
 4111 037704  001370                          BNE     40$             ;ISSUE 2 CLOCKS
 4112
 4113                                   ;LOAD ERROR REGISTER WITH ENTRY FROM TABLE AND VERIFY THAT EXCEPTION IS SET
 4114 037706  011260  000014                  MOV     (R2),RMER1(R0)  ;LOAD RMER1
 4115 037712  012737  000200  001534          MOV     #200,WATCH      ;SET WATCHDOG TIMER VALUE
        037720  004777  141612                  JSR     PC,@CLOCK       ;START THE CLOCK
 4116 037724                            45$:
        037724  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 4117 037732  042737  167777  001142          BIC     #^CREX,$BDDAT
```

```
 4118 037740  001021                          BNE     50$
 4119 037742  005737  001534                  TST     WATCH           ;HAS CLOCK EXPIRED ??
 4120 037746  001366                          BNE     45$             ;NO - TAKE ANOTHER SAMPLE
 4121 037750  004777  141564                  JSR     PC,@STOPCL      ;STOP THE CLOCK
 4122 037754  012737  010000  001140          MOV     #REX,$GDDAT
 4123 037762  010037  001136                  MOV     R0,$BDADR
 4124 037766  062737  000024  001136          ADD     #RMMR1,$BDADR
 4125 037774  011237  001174                  MOV     (R2),$TMP0
 4126 040000  104236                          EMT     236
 4127 040002  000407                          BR      60$
 4128
 4129                                 ;ADVANCE TABLE POINTER-EXIT IF DONE
 4130 040004                          50$:
 4131 040004  004777  141530                  JSR     PC,@STOPCL      ;STOP THE CLOCK
 4132 040010  062702  000002                  ADD     #2,R2
 4133 040014  005712                          TST     (R2)
 4134 040016  001401                          BEQ     60$
 4135 040020  000656                          BR      10$
 4136 040022  000402                  60$:    BR      200$            ;JUMP OVER TABLE
 4137
 4138                                 ;PRESENTLY, THE TABLE HAS ONLY ONE ENTRY, THE TEST USING RMR.  THE
 4139                                 ;TABLE SHOULD BE EXPANDED TO TEST ALL THE CONDITIONS LISTED ABOVE IF
 4140                                 ;A HARDWARE CHANGE IS MADE SUCH THAT IT IS POSSIBLE TO WRITE ERROR
 4141                                 ;REGISTER 1 WITH GO SET.
 4142 040024                          100$:
 4143 040024  000004                          .WORD   RMR             ;RMR IS CAUSED BY HARDWARE
 4144
 4145 040026  000000                          .WORD                   ;END OF TABLE
 4146 040030                          200$:                           ;END OF TEST
 4147
 4148                                 ;;*****************************************************************
                                      ;*TEST 112       RECALIBRATE TEST

                                      ;;*****************************************************************
      040030                          TST112:
      040030  000004                          SCOPE                   ;SCOPE CALL
      040032  000240                          NOP
      040034  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
      040040  013700  001276                  MOV     $BASE,R0        ;R0 - UNIBUS ADDRESS
      040044  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      040050  012737  000112  001226          MOV     #112,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 4149
 4150 040056  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     '040062  000403                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
      040064  104000                          EMT
 4151 040066  000137  041364                  JMP     330$
 4152
 4153                                 ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
 4154 040072                          10$:
 4155 040072  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 4156 040100  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 4157 040106  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 4158 040114  012760  000007  000000          MOV     #RECAL.GO,RMCS1(R0)     ;LOAD RMCS1
 4159
 4160                                 ;STEP COMMAND SEQUENCER TO RECAL COM (2 CLOCKS)
 4161 040122  012702  000002                  MOV     #2,R2
 4162 040126                          20$:
```

```
      040126  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4163  040134  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4164  040142  005302                          DEC     R2
4165  040144  001370                          BNE     20$
4166
4167                                  ;DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
4168  040146  012760  040001  000024          MOV     #DMD.DBEN,RMMR1(R0)     ;LOAD RMMR1
4169  040154  012702  000002                  MOV     #2,R2
4170  040160                          30$:
      040160  012760  140001  000024          MOV     #DMD!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
4171  040166  012760  040001  000024          MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
4172  040174  005302                          DEC     R2
4173  040176  001370                          BNE     30$
4174
4175                                  ;VERIFY THAT OPI IS SET
4176  040200  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
4177  040206  042737  157777  001142          BIC     #^COPI,$BDDAT
4178  040214  001011                          BNE     40$
4179  040216  012737  020000  001140          MOV     #OPI,$GDDAT
4180  040224  010037  001136                  MOV     R0,$BDADR
4181  040230  062737  000014  001136          ADD     #RMER1,$BDADR
4182  040236  104241                          EMT     241
4183
4184  040240  012737  040246  001124  40$:    MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
4185
4186                                  ;VERIFY THAT RECALIBRATE ABORTS DURING EXECUTION
4187  040246                          50$:
4188  040246  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      040252  000403                          BR      60$             ;BRANCH TO 60$ IF NO ERROR
      040254  104000                          EMT
4189  040256  000137  041364                  JMP     330$
4190
4191                                  ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
4192  040262                          60$:
4193  040262  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4194  040270  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
4195  040276  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
4196  040304  012760  000007  000000          MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
4197
4198                                  ;STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
4199  040312  012702  000003                  MOV     #3,R2
4200  040316                          70$:
4201  040316  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4202  040324  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4203  040332  005302                          DEC     R2
4204  040334  001370                          BNE     70$
4205
4206                                  ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
4207  040336  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
4208
4209                                  ;STEP 2 CLOCKS AND VERIFY GO IS RESET
4210  040344  012702  000002                  MOV     #2,R2
4211  040350                          80$:
      040350  012760  141101  000024          MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
4212  040356  012760  041101  000024          MOV     #DMD!MUR!DBEN.MDF,RMMR1(R0)     ;LOAD RMMR1
4213  040364  005302                          DEC     R2
4214  040366  001370                          BNE     80$
```

```
4215 040370  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4216 040376  042737  177776  001142          BIC     #^CGO,$BDDAT
4217 040404  001405                          BEQ     90$
4218 040406  010037  001136                  MOV     R0,$BDADR
4219 040412  005037  001140                  CLR     $GDDAT
4220 040416  104242                          EMT     242
4221
4222 040420  012737  040426  001124  90$:    MOV     #100$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4223
4224                                 ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
4225 040426                          100$:
4226 040426  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     040432  000403                          BR      110$            ;BRANCH TO 110$ IF NO ERROR
     040434  104000                          EMT
4227 040436  000137  041364                  JMP     330$
4228
4229                                 ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
4230 040442                          110$:
4231 040442  012760  041001  000024          MOV     #DMD!MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
4232 040450  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
4233 040456  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
4234 040464  012760  000007  000000          MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
4235
4236                                 ;STEP THE COMMAND SEQUENCER
4237 040472  012702  000017                  MOV     #15.,R2
4238 040476                          120$:
4239 040476  012760  141001  000024          MOV     #DMD!MUR!DBEN.DBCK,RMMR1(R0)    ;LOAD RMMR1
4240 040504  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4241 040512  005302                          DEC     R2
4242 040514  001370                          BNE     120$
4243
4244                                 ;VERIFY THAT OPI IS SET
4245 040516  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
4246 040524  042737  157777  001142          BIC     #^COPI,$BDDAT
4247 040532  001011                          BNE     130$
4248 040534  010037  001136                  MOV     R0,$BDADR
4249 040540  062737  000014  001136          ADD     #RMER1,$BDADR
4250 040546  012737  020000  001140          MOV     #OPI,$GDDAT
4251 040554  104243                          EMT     243
4252
4253 040556  012737  040564  001124  130$:   MOV     #150$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4254
4255                                 ;VERIFY ATA SETS IF DRIVE COMPLETES RECALIBRATE (ON CYLINDER SETS)
4256 040564                          150$:
4257 040564  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     040570  000403                          BR      160$            ;BRANCH TO 160$ IF NO ERROR
     040572  104000                          EMT
4258 040574  000137  041364                  JMP     330$
4259
4260                                 ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
4261 040600                          160$:
4262 040600  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4263 040606  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
4264 040614  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
4265 040622  012760  000007  000000          MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
4266
4267                                 ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
```

```
4268 040630  012702  000015              MOV     #13.,R2
4269 040634                        170$:
     040634  012760  141401  000024      MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
4270 040642  012760  041401  000024      MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4271 040650  005302                      DEC     R2
4272 040652  001370                      BNE     170$
4273
4274                               ;DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER
4275 040654  012760  041001  000024      MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4276 040662  012760  041401  000024      MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4277
4278                               ;STEP COMMAND SEQUENCER TO SET ATTENTION (3 CLOCKS)
4279 040670  012702  000003              MOV     #3,R2
4280 040674                        180$:
4281 040674  012760  141401  000024      MOV     #DMD!MUR!DBEN!MOC.DBCK,RMMR1(R0)        ;LOAD RMMR1
4282 040702  012760  041401  000024      MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4283 040710  005302                      DEC     R2
4284 040712  001370                      BNE     180$
4285
4286                               ;VERIFY ATA IS SET
4287 040714  016037  000012  001142      MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
4288 040722  042737  077777  001142      BIC     #^CATA,$BDDAT
4289 040730  001011                      BNE     190$
4290 040732  012737  100000  001140      MOV     #ATA,$GDDAT
4291 040740  010037  001136              MOV     R0,$BDADR
4292 040744  062737  000012  001136      ADD     #RMDS,$BDADR
4293 040752  104244                      EMT     244
4294
4295 040754  012737  040762  001124  190$:  MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4296
4297                               ;VERIFY THAT RECALIBRATE ABORTS AFTER EXECUTION DURING WAIT LOOP
4298 040762                        200$:
4299 040762  004737  054450              JSR     PC,SETVV        ;GO SET VOLUME VALID
     040766  000402                      BR      210$            ;BRANCH TO 210$ IF NO ERROR
     040770  104000                      EMT
4300 040772  000574                      BR      330$
4301
4302                               ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
4303 040774                        210$:
4304 040774  012760  041401  000024      MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4305 041002  012760  000000  000014      MOV     #0,RMER1(R0)    ;LOAD RMER1
4306 041010  012760  000000  000042      MOV     #0,RMER2(R0)    ;LOAD RMER2
4307 041016  012760  000007  000000      MOV     #RECAL!GO,RMCS1(R0)     ;LOAD RMCS1
4308
4309                               ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
4310 041024  012702  000015              MOV     #13.,R2
4311 041030                        220$:
4312 041030  012760  141401  000024      MOV     #DMD.MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
4313 041036  012760  041401  000024      MOV     #DMD.MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4314 041044  005302                      DEC     R2
4315 041046  001370                      BNE     220$
4316
4317                               ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
4318 041050  012760  041001  000024      MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4319
4320                               ;STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
4321 041056  012702  000007              MOV     #7,R2
```

```
4322 041062                                    230$:
4323 041062  012760  141001  000024                MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4324 041070  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4325 041076  005302                                DEC     R2
4326 041100  001370                                BNE     230$
4327
4328                                           ;VERIFY THAT GO IS STILL SET
4329 041102  016037  000000  001142                MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4330 041110  042737  177776  001142                BIC     #^CGO,$BDDAT
4331 041116  001006                                BNE     240$
4332 041120  012737  000001  001140                MOV     #GO,$GDDAT
4333 041126  010037  001136                        MOV     R0,$BDADR
4334 041132  104245                                EMT     245
4335
4336                                           ;SET SEEK INCOMPLETE ERROR
4337 041134                                    240$:
     041134  012760  041201  000024                MOV     #DMD!MUR.DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
4338
4339                                           ;STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
4340 041142  012702  000003                        MOV     #3,R2
4341 041146                                    250$:
     041146  012760  141201  000024                MOV     #DMD!MUR!DBEN!MSER.DBCK,RMMR1(R0)       ;LOAD RMMR1
4342 041154  012760  041201  000024                MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
4343 041162  005302                                DEC     R2
4344 041164  001370                                BNE     250$
4345 041166  016037  000000  001142                MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4346 041174  042737  177776  001142                BIC     #^CGO,$BDDAT
4347 041202  001405                                BEQ     260$
4348 041204  010037  001136                        MOV     R0,$BDADR
4349 041210  005037  001140                        CLR     $GDDAT
4350 041214  104246                                EMT     246
4351
4352 041216  012737  041224  001124  260$:        MOV     #300$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4353
4354                                           ;VERIFY THE TAG BUS DURING RECALIBRATE
4355 041224                                    300$:
4356 041224  004737  054450                        JSR     PC,SETVV        ;GO SET VOLUME VALID
     041230  000402                                BR      310$            ;BRANCH TO 310$ IF NO ERROR
     041232  104000                                EMT
4357 041234  000453                                BR      330$
4358
4359                                           ;ENABLE DEBUG CLOCK AND LOAD RECALIBRATE COMMAND
4360 041236                                    310$:
4361 041236  012760  041401  000024                MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4362 041244  012760  000000  000014                MOV     #0,RMER1(R0)    ;LOAD RMER1
4363 041252  012760  000000  000042                MOV     #0,RMER2(R0)    ;LOAD RMER2
4364 041260  012760  000007  000000                MOV     #RECAL.GO,RMCS1(R0)     ;LOAD RMCS1
4365 041266  012702  041366                        MOV     #400$,R2        ;INITIALIZE TABLE POINTER
4366
4367                                           ;VERIFY TAG BUS ACCORDING TO TABLE
4368 041272                                    315$:
     041272  016037  000040  001142                MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
4369 041300  042737  150000  001142                BIC     #RQA!RQB!TST,$BDDAT
4370 041306  021237  001142                        CMP     (R2),$BDDAT
4371 041312  001411                                BEQ     320$
4372 041314  011237  001140                        MOV     (R2),$GDDAT
4373 041320  010037  001136                        MOV     R0,$BDADR
```

```
4374 041324  062737  000040  001136          ADD     #RMMR2,$BDADR
4375 041332  104247                          EMT     247
4376 041334  000413                          BR      330$
4377
4378                                 ;ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
4379 041336                          320$:
4380 041336  062702  000002                  ADD     #2,R2
4381 041342  005712                          TST     (R2)
4382 041344  100407                          BMI     330$            ;EXIT IF ENTRY NEGATIVE
4383
4384                                 ;STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
4385 041346  012760  151001  000024          MOV     #DMD!DBEN!MUR!MOV!DBCK,RMMR1(R0)        ;LOAD RMMR1
4386 041354  012760  041401  000024          MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
4387 041362  000743                          BR      315$
4388 041364  000416                  330$:   BR      500$            ;JUMP OVER TABLE
4389
4390                                 ;TABLE OF TAG BUS CONTROL AND BIT VALUES
4391 041366                          400$:
4392 041366  001777                          .WORD   1777            ;BUS BITS AT HIGH IMPEDANCE STATE
4393 041370  001777                          .WORD   1777
4394 041372  006100                          .WORD   CC!CH!BB06      ;CONTROL BITS ENABLED, BIT 6 ON
4395 041374  006100                          .WORD   CC!CH!BB06
4396 041376  026100                          .WORD   TAG!CC!CH!BB06  ;TAG COMES ON
4397 041400  026100                          .WORD   TAG!CC!CH!BB06
4398 041402  026100                          .WORD   TAG.CC!CH!BB06
4399 041404  026100                          .WORD   TAG!CC!CH.BB06
4400 041406  026100                          .WORD   TAG!CC!CH!BB06
4401 041410  026100                          .WORD   TAG.CC!CH!BB06
4402 041412  006100                          .WORD   CC!CH!BB06      ;TAG GOES OFF
4403 041414  006100                          .WORD   CC!CH!BB06
4404 041416  001777                          .WORD   1777            ;CONTROL BITS DISABLED
4405
4406 041420  177777                          .WORD   -1              ;END OF TABLE
4407
4408 041422                          500$:                           ;END OF TEST
4409
4410                                 ;;*****************************************************************
                                     ;*TEST 113       SEEK TEST

                                     ;;*****************************************************************
     041422                          TST113:
     041422  000004                          SCOPE                   ;SCOPE CALL
     041424  000240                          NOP
     041426  012706  001100                  MOV     #STACK,SP       ;LOAD THE STACK POINTER
     041432  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     041436  013701  001466                  MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     041442  012737  000113  001226          MOV     #113,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
4411
4412                                 ;VERIFY THAT OPI SETS IF UNIT READY DROPS DURING COMMAND EXECUTION
4413
4414 041450  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     041454  000403                          BR      10$             ;BRANCH TO 10$ IF NO ERROR
     041456  104000                          EMT
4415 041460  000137  043100                  JMP     330$
4416
4417                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
4418                                 ;ADDRESS,  AND LOAD SEEK COMMAND
```

```
4419 041464                                    10$:
4420 041464  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4421 041472  012760  000000  000006                MOV     #0,RMDA(R0)      ;LOAD RMDA
4422 041500  012760  000000  000034                MOV     #0,RMDC(R0)      ;LOAD RMDC
4423 041506  012760  000000  000014                MOV     #0,RMER1(R0)     ;LOAD RMER1
4424 041514  012760  000000  000042                MOV     #0,RMER2(R0)     ;LOAD RMER2
4425 041522  012760  000005  000000                MOV     #SEEK!GO,RMCS1(R0)       ;LOAD RMCS1
4426
4427                                           ;STEP COMMAND SEQUENCER TO SEEK COM (2 CLOCKS)
4428 041530  012702  000002                        MOV     #2,R2
4429 041534                                    20$:
     041534  012760  141001  000024                MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4430 041542  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4431 041550  005302                                DEC     R2
4432 041552  001370                                BNE     20$
4433
4434                                           ;DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
4435 041554  012760  040001  000024                MOV     #DMD.DBEN,RMMR1(R0)      ;LOAD RMMR1
4436 041562  012702  000002                        MOV     #2,R2
4437 041566                                    30$:
     041566  012760  140001  000024                MOV     #DMD.DBEN.DBCK,RMMR1(R0)         ;LOAD RMMR1
4438 041574  012760  040001  000024                MOV     #DMD.DBEN,RMMR1(R0)      ;LOAD RMMR1
4439 041602  005302                                DEC     R2
4440 041604  001370                                BNE     30$
4441
4442                                           ;VERIFY THAT OPI IS SET
4443 041606  016037  000014  001142                MOV     RMER1(R0),$BDDAT         ;STORE RMER1 AT $BDDAT
4444 041614  042737  157777  001142                BIC     #^COPI,$BDDAT
4445 041622  001011                                BNE     40$
4446 041624  012737  020000  001140                MOV     #OPI,$GDDAT
4447 041632  010037  001136                        MOV     R0,$BDADR
4448 041636  062737  000014  001136                ADD     #RMER1,$BDADR
4449 041644  104250                                EMT     250
4450
4451 041646  012737  041654  001124  40$:     MOV     #50$,$LPERR      ;CHANGE LOOP ON ERROR ADDRESS
4452
4453                                           ;VERIFY THAT SEEK ABORTS DURING EXECUTION
4454 041654                                    50$:
4455 041654  004737  054450                        JSR     PC,SETVV         ;GO SET VOLUME VALID
     041660  000403                                BR      60$              ;BRANCH TO 60$ IF NO ERROR
     041662  104000                                EMT
4456 041664  000137  043100                        JMP     330$
4457
4458                                           ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
4459                                           ;ADDRESS,  AND LOAD SEEK COMMAND
4460 041670                                    60$:
4461 041670  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4462 041676  012760  000000  000006                MOV     #0,RMDA(R0)      ;LOAD RMDA
4463 041704  012760  000000  000034                MOV     #0,RMDC(R0)      ;LOAD RMDC
4464 041712  012760  000000  000014                MOV     #0,RMER1(R0)     ;LOAD RMER1
4465 041720  012760  000000  000042                MOV     #0,RMER2(R0)     ;LOAD RMER2
4466 041726  012760  000005  000000                MOV     #SEEK!GO,RMCS1(R0)       ;LOAD RMCS1
4467
4468                                           ;STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
4469 041734  012702  000003                        MOV     #3,R2
4470 041740                                    70$:
4471 041740  012760  141001  000024                MOV     #DMD.MUR!DBEN.DBCK,RMMR1(R0)     ;LOAD RMMR1
```

```
4472 041746  012760  041001  000024          MOV     #DMD!MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
4473 041754  005302                          DEC     R2
4474 041756  001370                          BNE     70$
4475
4476                                 ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
4477 041760  012760  041101  000024          MOV     #DMD!MUR!DBEN.MDF,RMMR1(R0)     ;LOAD RMMR1
4478
4479                                 ;STEP 2 CLOCKS AND VERIFY GO IS RESET
4480 041766  012702  000002                  MOV     #2,R2
4481 041772                          80$:
     041772  012760  141101  000024          MOV     #DMD!MUR!DBEN!MDF.DBCK,RMMR1(R0)        ;LOAD RMMR1
4482 042000  012760  041101  000024          MOV     #DMD!MUR!DBEN.MDF,RMMR1(R0)     ;LOAD RMMR1
4483 042006  005302                          DEC     R2
4484 042010  001370                          BNE     80$
4485 042012  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4486 042020  042737  177776  001142          BIC     #^CGO,$BDDAT
4487 042026  001405                          BEQ     90$
4488 042030  010037  001136                  MOV     R0,$BDADR
4489 042034  005037  001140                  CLR     $GDDAT
4490 042040  104251                          EMT     251
4491
4492 042042  012737  042050  001124  90$:    MOV     #100$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4493
4494                                 ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
4495 042050                          100$:
4496 042050  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     042054  000403                          BR      110$            ;BRANCH TO 110$ IF NO ERROR
     042056  104000                          EMT
4497 042060  000137  043100                  JMP     330$
4498
4499                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
4500                                 ;ADDRESS,  AND LOAD SEEK COMMAND
4501 042064                          110$:
4502 042064  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4503 042072  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
4504 042100  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
4505 042106  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
4506 042114  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
4507 042122  012760  000005  000000          MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
4508
4509                                 ;STEP THE COMMAND SEQUENCER
4510 042130  012702  000017                  MOV     #15.,R2
4511 042134                          '20$:
4512 042134  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4513 042142  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4514 042150  005302                          DEC     R2
4515 042152  001370                          BNE     120$
4516
4517                                 ;VERIFY THAT OPI IS SET
4518 042154  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
4519 042162  042737  157777  001142          BIC     #^COPI,$BDDAT
4520 042170  001011                          BNE     130$
4521 042172  010037  001136                  MOV     R0,$BDADR
4522 042176  062737  000014  001136          ADD     #RMER1,$BDADR
4523 042204  012737  020000  001140          MOV     #OPI,$GDDAT
4524 042212  104252                          EMT     252
4525
```

```
4526 042214  012737  042222  001124  130$:   MOV     #150$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4527
4528                                 ;VERIFY ATA SETS IF DRIVE COMPLETES SEEK (ON CYLINDER SETS)
4529 042222                          150$:
4530 042222  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     042226  000403                          BR      160$            ;BRANCH TO 160$ IF NO ERROR
     042230  104000                          EMT
4531 042232  000137  043100                  JMP     330$
4532
4533                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
4534                                 ;ADDRESS,  AND LOAD SEEK COMMAND
4535 042236                          160$:
4536 042236  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4537 042244  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
4538 042252  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
4539 042260  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
4540 042266  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
4541 042274  012760  000005  000000          MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
4542
4543                                 ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
4544 042302  012702  000015                  MOV     #13.,R2
4545 042306                          170$:
     042306  012760  141401  000024          MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
4546 042314  012760  041401  000024          MOV     #DMD!MUR.DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4547 042322  005302                          DEC     R2
4548 042324  001370                          BNE     170$
4549
4550                                 ;DROP ON CYLINDER TO RESET LATCH, THEN SET ON CYLINDER
4551 042326  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4552 042334  012760  041401  000024          MOV     #DMD.MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4553
4554                                 ;STEP COMMAND SEQUENCER TO SET ATTENTION (3 CLOCKS)
4555 042342  012702  000003                  MOV     #3,R2
4556 042346                          180$:
4557 042346  012760  141401  000024          MOV     #DMD.MUR.DBEN.MOC.DBCK,RMMR1(R0)        ;LOAD RMMR1
4558 042354  012760  041401  000024          MOV     #DMD!MUR.DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4559 042362  005302                          DEC     R2
4560 042364  001370                          BNE     180$
4561
4562                                 ;VERIFY ATA IS SET
4563 042366  016037  000012  001142          MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
4564 042374  042737  077777  001142          BIC     #^CATA,$BDDAT
4565 042402  001011                          BNE     190$
4566 042404  012737  100000  001140          MOV     #ATA,$GDDAT
4567 042412  010037  001136                  MOV     R0,$BDADR
4568 042416  062737  000012  001136          ADD     #RMDS,$BDADR
4569 042424  104253                          EMT     253
4570
4571 042426  012737  042434  001124  190$:   MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4572
4573                                 ;VERIFY THAT SEEK ABORTS AFTER EXECUTION DURING WAIT LOOP
4574 042434                          200$:
4575 042434  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     042440  000403                          BR      210$            ;BRANCH TO 210$ IF NO ERROR
     042442  104000                          EMT
4576 042444  000137  043100                  JMP     330$
4577
```

```
4578                                    ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
4579                                    ;ADDRESS  AND LOAD SEEK COMMAND
4580 042450                             210$:
4581 042450  012760  041401  000024             MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4582 042456  012760  000000  000006             MOV     #0,RMDA(R0)     ;LOAD RMDA
4583 042464  012760  000000  000034             MOV     #0,RMDC(R0)     ;LOAD RMDC
4584 042472  012760  000000  000014             MOV     #0,RMER1(R0)    ;LOAD RMER1
4585 042500  012760  000000  000042             MOV     #0,RMER2(R0)    ;LOAD RMER2
4586 042506  012760  000005  000000             MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
4587
4588                                    ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (13 CLOCKS)
4589 042514  012702  000015                     MOV     #13.,R2
4590 042520                             220$:
4591 042520  012760  141401  000024             MOV     #DMD!MUR.DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
4592 042526  012760  041401  000024             MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4593 042534  005302                             DEC     R2
4594 042536  001370                             BNE     220$
4595
4596                                    ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
4597 042540  012760  041001  000024             MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4598
4599                                    ;STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
4600 042546  012702  000007                     MOV     #7,R2
4601 042552                             230$:
4602 042552  012760  141001  000024             MOV     #DMD.MUR!DBEN.DBCK,RMMR1(R0)    ;LOAD RMMR1
4603 042560  012760  041001  000024             MOV     #DMD.MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
4604 042566  005302                             DEC     R2
4605 042570  001370                             BNE     230$
4606
4607                                    ;VERIFY THAT GO IS STILL SET
4608 042572  016037  000000  001142             MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4609 042600  042737  177776  001142             BIC     #^CGO,$BDDAT
4610 042606  001006                             BNE     240$
4611 042610  012737  000001  001140             MOV     #GO,$GDDAT
4612 042616  010037  001136                     MOV     R0,$BDADR
4613 042622  104254                             EMT     254
4614
4615                                    ;SET SEEK INCOMPLETE ERROR
4616 042624                             240$:
     042624  012760  041201  000024             MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
4617
4618                                    ;STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
4619 042632  012702  000003                     MOV     #3,R2
4620 042636                             250$:
     042636  012760  141201  000024             MOV     #DMD.MUR!DBEN!MSER.DBCK,RMMR1(R0)       ;LOAD RMMR1
4621 042644  012760  041201  000024             MOV     #DMD'MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
4622 042652  005302                             DEC     R2
4623 042654  001370                             BNE     250$
4624 042656  016037  000000  001142             MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4625 042664  042737  177776  001142             BIC     #^CGO,$BDDAT
4626 042672  001405                             BEQ     260$
4627 042674  010037  001136                     MOV     R0,$BDADR
4628 042700  005037  001140                     CLR     $GDDAT
4629 042704  104255                             EMT     255
4630
4631 042706  012737  042720  001124  260$:      MOV     #300$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4632 042714  012703  000001                     MOV     #1,R3           ;INITIALIZE CYLINDER ADDRESS
```

```
4633
4634                                    ;VERIFY THE TAG BUS DURING SEEK
4635 042720                             300$:
4636 042720  004737  054450                    JSR     PC,SETVV        ;GO SET VOLUME VALID
     042724  000402                            BR      310$            ;BRANCH TO 310$ IF NO ERROR
     042726  104000                            EMT
4637 042730  000463                            BR      330$
4638
4639                                    ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
4640                                    ;ADDRESS  AND LOAD SEEK COMMAND
4641 042732                             310$:
4642 042732  012760  041401  000024            MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4643 042740  012760  000000  000006            MOV     #0,RMDA(R0)     ;LOAD RMDA
4644 042746  010360  000034                    MOV     R3,RMDC(R0)     ;LOAD RMDC
4645 042752  012760  000000  000014            MOV     #0,RMER1(R0)    ;LOAD RMER1
4646 042760  012760  000000  000042            MOV     #0,RMER2(R0)    ;LOAD RMER2
4647 042766  012760  000005  000000            MOV     #SEEK!GO,RMCS1(R0)      ;LOAD RMCS1
4648 042774  012702  043114                    MOV     #400$,R2        ;INITIALIZE TABLE POINTER
4649
4650                                    ;VERIFY TAG BUS ACCORDING TO TABLE AND CYLINDER IN R3
4651 043000                             315$:
     043000  016037  000040  001142            MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
4652 043006  042737  150000  001142            BIC     #RQA!RQB!TST,$BDDAT
4653 043014  011237  001140                    MOV     (R2),$GDDAT
4654 043020  050337  001140                    BIS     R3,$GDDAT       ;OR CYLINDER ADDRESS IN
4655 043024  023737  001140  001142            CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
4656 043032  001407                            BEQ     320$            ;BRANCH IF TAG BUS OK
4657 043034  010037  001136                    MOV     R0,$BDADR
4658 043040  062737  000040  001136            ADD     #RMMR2,$BDADR
4659 043046  104256                            EMT     256
4660 043050  000413                            BR      330$
4661
4662                                    ;ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
4663 043052                             320$:
4664 043052  062702  000002                    ADD     #2,R2
4665 043056  005712                            TST     (R2)
4666 043060  100407                            BMI     330$            ;EXIT IF ENTRY NEGATIVE
4667
4668                                    ;STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
4669 043062  012760  151001  000024            MOV     #DMD!DBEN!MUR!MOV!DBCK,RMMR1(R0)        ;LOAD RMMR1
4670 043070  012760  041401  000024            MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
4671 043076  000740                            BR      315$
4672
4673                                    ;REPEAT TAG BUS TEST FOR EACH PRIME CYLINDER, I.E., 1,2,4,...
4674 043100                             330$:
4675 043100  006303                            ASL     R3              ;SHIFT TO NEXT CYLINDER
4676 043102  020327  002000                    CMP     R3,#1024.
4677 043106  103001                            BHIS    340$            ;EXIT IF WAS DONE
4678 043110  000703                            BR      300$            ;TEST NEXT CYLINDER
4679 043112  000416                    340$:   BR      500$            ;JUMP OVER TABLE
4680
4681                                    ;TABLE OF TAG BUS CONTROL AND BIT VALUES
4682 043114                             400$:
4683 043114  001777                            .WORD   1777            ;BUS BITS AT HIGH IMPEDANCE STATE
4684 043116  001777                            .WORD   1777
4685 043120  004000                            .WORD   CC              ;CONTROL BITS ENABLED, BIT 6 ON
4686 043122  004000                            .WORD   CC
```

```
4687 043124  024000                    .WORD   TAG!CC          ;TAG COMES ON
4688 043126  024000                    .WORD   TAG!CC
4689 043130  024000                    .WORD   TAG!CC
4690 043132  024000                    .WORD   TAG!CC
4691 043134  024000                    .WORD   TAG!CC
4692 043136  024000                    .WORD   TAG!CC
4693 043140  004000                    .WORD   CC              ;TAG GOES OFF
4694 043142  004000                    .WORD   CC
4695 043144  001777                    .WORD   1777            ;CONTROL BITS DISABLED
4696
4697 043146  177777                    .WORD   -1              ;END OF TABLE
4698
4699 043150                        500$:                       ;END OF TEST
4700
4701                                ;;*****************************************************************
                                    ;*TEST 114       SEARCH TEST
                                    ;;*****************************************************************
     043150                         TST114:
     043150  000004                        SCOPE               ;SCOPE CALL
     043152  000240                        NOP
     043154  012706  001100                MOV     #STACK,SP   ;LOAD THE STACK POINTER
     043160  013700  001276                MOV     $BASE,R0    ;R0 = UNIBUS ADDRESS
     043164  013701  001466                MOV     TSTQUE,R1   ;R1 = POINTER TO DEVICE
     043170  012737  000114  001226        MOV     #114,$TESTN ;;SET TEST NUMBER IN APT MAIL BOX
4702
4703                                ;VERIFY THAT OPI SETS IF UNIT READY DROPS DURING COMMAND EXECUTION
4704
4705 043176  004737  054450                JSR     PC,SETVV    ;GO SET VOLUME VALID
     043202  000403                        BR      10$         ;BRANCH TO 10$ IF NO ERROR
     043204  104000                        EMT
4706 043206  000137  045244                JMP     330$
4707
4708                                ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
4709                                ;ADDRESS,  AND LOAD SEARCH COMMAND
4710 043212                        10$:
4711 043212  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4712 043220  012760  000000  000006        MOV     #0,RMDA(R0)     ;LOAD RMDA
4713 043226  012760  000000  000034        MOV     #0,RMDC(R0)     ;LOAD RMDC
4714 043234  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
4715 043242  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
4716 043250  012760  000031  000000        MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
4717
4718                                ;STEP COMMAND SEQUENCER TO SEARCH COM (2 CLOCKS)
4719 043256  012702  000002                MOV     #2,R2
4720 043262                        20$:
     043262  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4721 043270  012760  041001  000024        MOV     #DMD.MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
4722 043276  005302                        DEC     R2
4723 043300  001370                        BNE     20$
4724
4725                                ;DROP UNIT READY AND STEP COMMAND SEQUENCER (2 CLOCKS)
4726 043302  012760  040001  000024        MOV     #DMD!DBEN,RMMR1(R0)     ;LOAD RMMR1
4727 043310  012702  000002                MOV     #2,R2
4728 043314                        30$:
     043314  012760  140001  000024        MOV     #DMD!DBEN.DBCK,RMMR1(R0)        ;LOAD RMMR1
4729 043322  012760  040001  000024        MOV     #DMD.DBEN,RMMR1(R0)     ;LOAD RMMR1
```

```
4730 043330  005302                        DEC     R2
4731 043332  001370                        BNE     30$
4732
4733                                 ;VERIFY THAT OPI IS SET
4734 043334  016037  000014  001142        MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
4735 043342  042737  157777  001142        BIC     #^COPI,$BDDAT
4736 043350  001011                        BNE     40$
4737 043352  012737  020000  001140        MOV     #OPI,$GDDAT
4738 043360  010037  001136                MOV     R0,$BDADR
4739 043364  062737  000014  001136        ADD     #RMER1,$BDADR
4740 043372  104257                        EMT     257
4741
4742 043374  012737  043402  001124  40$:  MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
4743
4744                                 ;VERIFY THAT SEARCH ABORTS DURING EXECUTION
4745 043402                          50$:
4746 043402  004737  054450                JSR     PC,SETVV        ;GO SET VOLUME VALID
     043406  000403                        BR      60$             ;BRANCH TO 60$ IF NO ERROR
     043410  104000                        EMT
4747 043412  000137  045244                JMP     330$
4748
4749                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
4750                                 ;ADDRESS,  AND LOAD SEARCH COMMAND
4751 043416                          60$:
4752 043416  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4753 043424  012760  000000  000006        MOV     #0,RMDA(R0)     ;LOAD RMDA
4754 043432  012760  000000  000034        MOV     #0,RMDC(R0)     ;LOAD RMDC
4755 043440  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
4756 043446  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
4757 043454  012760  000031  000000        MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
4758
4759                                 ;STEP THE COMMAND SEQUENCER TO FIRST TEST FOR ABORT (3 CLOCKS)
4760 043462  012702  000003                MOV     #3,R2
4761 043466                          70$:
4762 043466  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4763 043474  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4764 043502  005302                        DEC     R2
4765 043504  001370                        BNE     70$
4766
4767                                 ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
4768 043506  012760  041101  000024        MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
4769
4770                                 ;STEP 2 CLOCKS AND VERIFY GO IS RESET
4771 043514  012702  000002                MOV     #2,R2
4772 043520                          80$:
     043520  012760  141101  000024        MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
4773 043526  012760  041101  000024        MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
4774 043534  005302                        DEC     R2
4775 043536  001370                        BNE     80$
4776 043540  016037  000000  001142        MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4777 043546  042737  177776  001142        BIC     #^CGO,$BDDAT
4778 043554  001405                        BEQ     90$
4779 043556  010037  001136                MOV     R0,$BDADR
4780 043562  005037  001140                CLR     $GDDAT
4781 043566  104260                        EMT     260
4782
4783                                 ;(THE OTHER TWO ABORT TESTS IN THE COMMAND SEQUENCER ARE TESTED
```

```
                                        ;DURING DATA COMMAND TESTS)
                                        
043570  012737  043576  001124  90$:    MOV     #100$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
                                        
                                        ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT CLEAR
043576                                  100$:
043576  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
043602  000403                          BR      110$            ;BRANCH TO 110$ IF NO ERROR
043604  104000                          EMT
043606  000137  045244                  JMP     330$
                                        
                                        ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
                                        ;ADDRESS,  AND LOAD SEARCH COMMAND
043612                                  110$:
043612  012760  041001  000024          MOV     #DMD'MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
043620  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
043626  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
043634  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
043642  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
043650  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
                                        
                                        ;STEP THE COMMAND SEQUENCER
043656  012702  000023                  MOV     #19.,R2
043662                                  120$:
043662  012760  141001  000024          MOV     #DMD!MUR.DBEN.DBCK,RMMR1(R0)    ;LOAD RMMR1
043670  012760  041001  000024          MOV     #DMD!MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
043676  005302                          DEC     R2
043700  001370                          BNE     120$
                                        
                                        ;VERIFY THAT OPI IS SET
043702  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
043710  042737  157777  001142          BIC     #^COPI,$BDDAT
043716  001011                          BNE     130$
043720  010037  001136                  MOV     R0,$BDADR
043724  062737  000014  001136          ADD     #RMER1,$BDADR
043732  012737  020000  001140          MOV     #OPI,$GDDAT
043740  104261                          EMT     261
                                        
043742  012737  043750  001124  130$:   MOV     #150$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
                                        
                                        ;VERIFY ATA SETS IF DRIVE COMPLETES SEARCH (ON CYLINDER AND
                                        ;SECTOR COMPARE SETS)
043750                                  150$:
043750  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
043754  000403                          BR      160$            ;BRANCH TO 160$ IF NO ERROR
043756  104000                          EMT
043760  000137  045244                  JMP     330$
                                        
                                        ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
                                        ;ADDRESS,  AND LOAD SEARCH COMMAND
043764                                  160$:
043764  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
043772  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
044000  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
044006  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
044014  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
044022  012760  000031  000000          MOV     #SEARCH'GO,RMCS1(R0)    ;LOAD RMCS1
```

```
 4837
 4838                                   ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
 4839 044030  012702  000021                    MOV     #17.,R2
 4840 044034                            170$:
      044034  012760  141401  000024            MOV     #DMD!MUR!DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
 4841 044042  012760  041401  000024            MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 4842 044050  005302                            DEC     R2
 4843 044052  001370                            BNE     170$
 4844
 4845                                   ;DROP ON CYLINDER TO RESET LATCH,AND RAISE INDEX PULSE
 4846                                   ;TO SET FORMAT CHANGE FLOP
 4847 044054  012760  041005  000024            MOV     #DMD!MUR!DBEN!MI,RMMR1(R0)      ;LOAD RMMR1
 4848
 4849                                   ;RAISE ON CYLINDER AND INHIBIT SEARCH TIMEOUT
 4850 044062  012760  051401  000024            MOV     #DMD!MUR!DBEN!MOC!MSEN,RMMR1(R0)        ;LOAD RMMR1
 4851
 4852                                   ;STEP COMMAND SEQUENCER TO SEARCH ENABLE (2 CLOCKS)
 4853 044070  012702  000002                    MOV     #2,R2
 4854 044074                            180$:
 4855 044074  012760  151401  000024            MOV     #DMD!MUR.DBEN!MOC!MSEN!DBCK,RMMR1(R0)   ;LOAD RMMR1
 4856 044102  012760  051401  000024            MOV     #DMD!MUR!DBEN!MOC!MSEN,RMMR1(R0)        ;LOAD RMMR1
 4857 044110  005302                            DEC     R2
 4858 044112  001370                            BNE     180$
 4859
 4860                                   ;FORCE SECTOR COMPARE BY CLOCKING SECTOR PULSE WITH SECTOR COMPARE
 4861                                   ;ACTIVE
 4862 044114  012760  051403  000024            MOV     #DMD!MUR!MOC!DBEN!MSEN!MSC,RMMR1(R0)    ;LOAD RMMR1
 4863 044122  012760  051443  000024            MOV     #DMD!MUR!MOC!DBEN!MSEN!MSC!MS,RMMR1(R0) ;LOAD RMMR1
 4864 044130  012760  051403  000024            MOV     #DMD!MUR!MOC!DBEN!MSEN!MSC,RMMR1(R0)    ;LOAD RMMR1
 4865
 4866                                   ;CLOCK SEQUENCER TO SET ATA (3 CLOCKS)
 4867 044136  012702  000003                    MOV     #3,R2           ;R2 = CLOCK COUNT
 4868 044142                            185$:
      044142  012760  151401  000024            MOV     #DMD!MUR!MOC!DBEN!MSEN!DBCK,RMMR1(R0)   ;LOAD RMMR1
 4869 044150  012760  051401  000024            MOV     #DMD!MUR!MOC!DBEN.MSEN,RMMR1(R0)        ;LOAD RMMR1
 4870 044156  005302                            DEC     R2
 4871 044160  001370                            BNE     185$
 4872
 4873                                   ;VERIFY ATA IS SET
 4874 044162  016037  000012  001142            MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 4875 044170  042737  077777  001142            BIC     #^CATA,$BDDAT
 4876 044176  001011                            BNE     190$
 4877 044200  012737  100000  001140            MOV     #ATA,$GDDAT
 4878 044206  010037  001136                    MOV     R0,$BDADR
 4879 044212  062737  000012  001136            ADD     #RMDS,$BDADR
 4880 044220  104262                            EMT     262
 4881
 4882 044222  012737  044230  001124  190$:    MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 4883
 4884                                   ;VERIFY THAT SEARCH ABORTS AFTER EXECUTION DURING SEARCH SEEK LOOP
 4885 044230                            200$:
 4886 044230  004737  054450                    JSR     PC,SETVV        ;GO SET VOLUME VALID
      044234  000403                            BR      210$            ;BRANCH TO 210$ IF NO ERROR
      044236  104000                            EMT
 4887 044240  000137  045244                    JMP     330$
 4888
 4889                                   ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
```

```
4890                                        ;ADDRESS  AND LOAD SEARCH COMMAND
4891 044244                                 210$:
4892 044244  012760  041401  000024                 MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4893 044252  012760  000000  000006                 MOV     #0,RMDA(R0)     ;LOAD RMDA
4894 044260  012760  000000  000034                 MOV     #0,RMDC(R0)     ;LOAD RMDC
4895 044266  012760  000000  000014                 MOV     #0,RMER1(R0)    ;LOAD RMER1
4896 044274  012760  000000  000042                 MOV     #0,RMER2(R0)    ;LOAD RMER2
4897 044302  012760  000031  000000                 MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
4898
4899                                        ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
4900 044310  012702  000021                 MOV     #17.,R2
4901 044314                                 220$:
4902 044314  012760  141401  000024                 MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
4903 044322  012760  041401  000024                 MOV     #DMD.MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4904 044330  005302                                 DEC     R2
4905 044332  001370                                 BNE     220$
4906
4907                                        ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER 0
4908 044334  012760  041001  000024                 MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4909
4910                                        ;STEP COMMAND SEQUENCER THROUGH WAIT LOOP (7 CLOCKS)
4911 044342  012702  000007                 MOV     #7,R2
4912 044346                                 230$:
4913 044346  012760  141001  000024                 MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
4914 044354  012760  041001  000024                 MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
4915 044362  005302                                 DEC     R2
4916 044364  001370                                 BNE     230$
4917
4918                                        ;VERIFY THAT GO IS STILL SET
4919 044366  016037  000000  001142                 MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4920 044374  042737  177776  001142                 BIC     #^CGO,$BDDAT
4921 044402  001006                                 BNE     240$
4922 044404  012737  000001  001140                 MOV     #GO,$GDDAT
4923 044412  010037  001136                         MOV     R0,$BDADR
4924 044416  104263                                 EMT     263
4925
4926                                        ;SET SEEK INCOMPLETE ERROR
4927 044420                                 240$:
     044420  012760  041201  000024                 MOV     #DMD.MUR!DBEN.MSER,RMMR1(R0)    ;LOAD RMMR1
4928
4929                                        ;STEP COMMAND SEQUENCER AND VERIFY GO RESETS (3 CLOCKS)
4930 044426  012702  000003                 MOV     #3,R2
4931 044432                                 250$:
     044432  012760  141201  000024                 MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)       ;LOAD RMMR1
4932 044440  012760  041201  000024                 MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
4933 044446  005302                                 DEC     R2
4934 044450  001370                                 BNE     250$
4935 044452  016037  000000  001142                 MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
4936 044460  042737  177776  001142                 BIC     #^CGO,$BDDAT
4937 044466  001405                                 BEQ     260$
4938 044470  010037  001136                         MOV     R0,$BDADR
4939 044474  005037  001140                         CLR     $GDDAT
4940 044500  104264                                 EMT     264
4941
4942 044502  012737  044510  001124  260$:  MOV     #265$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
4943
4944                                        ;VERIFY THAT SEARCH ABORTS DURING SECTOR COMPARE LOOP
```

```
4945 044510                          265$:
4946 044510  004737  054450  .              JSR     PC,SETVV        ;GO SET VOLUME VALID
     044514  000403                         BR      270$            ;BRANCH TO 270$ IF NO ERROR
     044516  104000                         EMT
4947 044520  000137  045244              JMP     330$
4948
4949                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
4950                                 ;ADDRESS,  AND LOAD SEARCH COMMAND
4951 044524                          270$:
4952 044524  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4953 044532  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
4954 044540  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
4955 044546  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
4956 044554  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
4957 044562  012760  000031  000000          MOV     #SEARCH.GO,RMCS1(R0)    ;LOAD RMCS1
4958
4959                                 ;STEP COMMAND SEQUENCER TO FIRST ON LATCH TEST (17 CLOCKS)
4960 044570  012702  000021              MOV     #17.,R2
4961 044574                          275$:
     044574  012760  141401  000024          MOV     #DMD!MUR.DBEN!DBCK!MOC,RMMR1(R0)        ;LOAD RMMR1
4962 044602  012760  041401  000024          MOV     #DMD.MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
4963 044610  005302                          DEC     R2
4964 044612  001370                          BNE     275$
4965
4966                                 ;DROP ON CYLINDER TO RESET LATCH, AND RAISE INDEX PULSE
4967                                 ;TO SET FORMAT CHANGE FLOP
4968 044614  012760  041005  000024          MOV     #DMD!MUR!DBEN!MI,RMMR1(R0)      ;LOAD RMMR1
4969
4970                                 ;RAISE ON CYLINDER AND INHIBIT SEARCH TIMEOUT
4971 044622  012760  051401  000024          MOV     #DMD!MUR!DBEN!MOC!MSEN,RMMR1(R0)        ;LOAD RMMR1
4972
4973                                 ;STEP COMMAND SEQUENCER TO SEARCH ENABLE (2 CLOCKS)
4974 044630  012702  000002              MOV     #2,R2
4975 044634                          280$:
4976 044634  012760  151401  000024          MOV     #DMD!MUR!DBEN!MOC!MSEN!DBCK,RMMR1(R0)   ;LOAD RMMR1
4977 044642  012760  051401  000024          MOV     #DMD!MUR!DBEN!MOC!MSEN,RMMR1(R0)        ;LOAD RMMR1
4978 044650  005302                          DEC     R2
4979 044652  001370                          BNE     280$
4980
4981                                 ;VERIFY THAT SEARCH ENABLE IS ON DURING SECTOR COMPARE LOOP
4982 044654  012702  000004              MOV     #4,R2           ;R2 - CLOCK COUNT
4983 044660                          281$:
4984 044660  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
4985 044666  042737  173777  001142          BIC     #^CESRC,$BDDAT
4986 044674  001411                          BEQ     282$            ;BRANCH IF SEARCH NOT ENABLED
4987 044676  012760  151401  000024          MOV     #DMD!MUR!MOC!DBEN!MSEN!DBCK,RMMR1(R0)   ;LOAD RMMR1
4988 044704  012760  051401  000024          MOV     #DMD!MUR!MOC!DBEN!MSEN,RMMR1(R0)        ;LOAD RMMR1
4989 044712  005302                          DEC     R2
4990 044714  001361                          BNE     281$
4991 044716  000411                          BR      283$
4992 044720  012737  004000  001140  282$:   MOV     #ESRC,$GDDAT
4993 044726  010037  001136                  MOV     R0,$BDADR
4994 044732  062737  000024  001136          ADD     #RMMR1,$BDADR
4995 044740  104265                          EMT     265
4996
4997                                 ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
4998 044742                          283$:
```

```
      044742  012760  051501  000024          MOV     #DMD!MUR!DBEN!MOC!MSEN!MDF,RMMR1(R0)    ;LOAD RMMR1
 4999
 5000                                 ;STEP 2 CLOCKS AND VERIFY GO IS RESET
 5001 044750  012702  000002          MOV     #2,R2           ;R2 - CLOCK COUNT
 5002 044754                          285$:
      044754  012760  151501  000024          MOV     #DMD!MUR!DBEN!MOC!MSEN!MDF!DBCK,RMMR1(R0)       ;LOAD RMMR1
 5003 044762  012760  051501  000024          MOV     #DMD!MUR!DBEN!MOC!MSEN!MDF,RMMR1(R0)    ;LOAD RMMR1
 5004 044770  005302                          DEC     R2
 5005 044772  001370                          BNE     285$
 5006 044774  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
 5007 045002  042737  177776  001142          BIC     #^CGO,$BDDAT
 5008 045010  001406                          BEQ     290$
 5009 045012  012737  000000  001140          MOV     #0,$GDDAT
 5010 045020  010037  001136                  MOV     R0,$BDADR
 5011 045024  104260                          EMT     260
 5012
 5013 045026  012737  045040  001124  290$:   MOV     #300$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 5014 045034  012703  000001                  MOV     #1,R3           ;INITIALIZE CYLINDER ADDRESS
 5015
 5016                                 ;VERIFY THE TAG BUS DURING SEARCH
 5017 045040                          300$:
 5018 045040  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      045044  000402                          BR      310$            ;BRANCH TO 310$ IF NO ERROR
      045046  104000                          EMT
 5019 045050  000475                          BR      330$
 5020
 5021                                 ;ENABLE DEBUG CLOCK, LOAD CYLINDER, TRACK AND SECTOR
 5022                                 ;ADDRESS  AND LOAD SEARCH COMMAND
 5023 045052                          310$:
 5024 045052  012760  041401  000024          MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 5025 045060  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 5026 045066  010360  000034                  MOV     R3,RMDC(R0)     ;LOAD RMDC
 5027 045072  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 5028 045100  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 5029 045106  012760  000031  000000          MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
 5030
 5031                                 ;;*****************************************************************
 5032                                 ;       MOV     #400$,R2        ;INITIALIZE TABLE POINTER
 5033
 5034                                                 ;HARDWARE ECO CHANGE TO THE PLA OF THE
 5035                                                 ;CS BOARD.
 5036 045114  012702  000011                  MOV     #9.,R2          ;CLOCK THE SEQUENCER THRU THE FIRST
 5037                                                 ;9. COMMAND SEQUENCES TO ALLOW THE PROGRAM
 5038                                                 ;TO RUN WITH OR WITHOUT THE ECO.
 5039 045120                          312$:
 5040 045120  012760  141401  000024          MOV     #DMD!DBEN!MUR!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5041 045126  012760  041401  000024          MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
 5042 045134  005302                          DEC     R2              ;DONE 9. CLOCKS ?
 5043 045136  001370                          BNE     312$            ;NO !!
 5044 045140  012702  045302                  MOV     #450$,R2        ;INITIALIZE NEW TABLE POINTER
 5045                                 ;;*****************************************************************
 5046
 5047                                 ;VERIFY TAG BUS ACCORDING TO TABLE AND CYLINDER IN R3
 5048 045144                          315$:
      045144  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
 5049 045152  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
 5050 045160  011237  001140                  MOV     (R2),$GDDAT
```

```
5051 045164  050337  001140                BIS     R3,$GDDAT       ;OR CYLINDER ADDRESS IN
5052 045170  023737  001140  001142        CMP     $GDDAT,$BDDAT   ;COMPARE EXPECTED AND RECEIVED
5053 045176  001407                        BEQ     320$            ;BRANCH IF TAG BUS OK
5054 045200  010037  001136                MOV     R0,$BDADR
5055 045204  062737  000040  001136        ADD     #RMMR2,$BDADR
5056 045212  104266                        EMT     266
5057 045214  000420                        BR      340$
5058
5059                                ;ADVANCE TO NEXT ENTRY IN TABLE-EXIT IF DONE
5060 045216                         320$:
5061 045216  062702  000002                ADD     #2,R2
5062 045222  005712                        TST     (R2)
5063 045224  100407                        BMI     330$            ;EXIT IF ENTRY NEGATIVE
5064
5065                                ;STEP THE COMMAND SEQUENCER AND REPEAT VERIFICATION
5066 045226  012760  141401  000024        MOV     #DMD!DBEN!MUR!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
5067 045234  012760  041401  000024        MOV     #DMD!DBEN!MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
5068 045242  000740                        BR      315$
5069
5070                                ;REPEAT TAG BUS TEST FOR EACH PRIME CYLINDER, I.E., 1,2,4,...
5071 045244                         330$:
5072 045244  006303                        ASL     R3              ;SHIFT TO NEXT CYLINDER
5073 045246  020327  002000                CMP     R3,#1024.
5074 045252  103001                        BHIS    340$            ;EXIT IF WAS DONE
5075 045254  000671                        BR      300$            ;TEST NEXT CYLINDER
5076 045256  000424                340$:   BR      500$            ;JUMP OVER TABLE
5077
5078                                ;TABLE OF TAG BUS CONTROL AND BIT VALUES
5079 045260                         400$:
5080 045260  001777                        .WORD   1777            ;BUS BITS AT HIGH IMPEDENCE STATE
5081 045262  001777                        .WORD   1777
5082 045264  001777                        .WORD   1777
5083 045266  001777                        .WORD   1777
5084 045270  001777                        .WORD   1777
5085 045272  004000                        .WORD   CC              ;CONTROL BITS ENABLED, BIT 6 ON
5086 045274  004000                        .WORD   CC
5087 045276  024000                        .WORD   TAG!CC          ;TAG COMES ON
5088 045300  024000                        .WORD   TAG!CC
5089 045302                         450$:                       ;START TABLE HERE FOR HARDWARE ECO CHANGE
5090 045302  024000                        .WORD   TAG!CC
5091 045304  024000                        .WORD   TAG.CC
5092 045306  024000                        .WORD   TAG!CC
5093 045310  024000                        .WORD   TAG.CC
5094 045312  177777                        .WORD   -1              ;END TABLE HERE FOR HARDWARE ECO CHANGE
5095
5096                                ;       .WORD   CC              ;TAG GOES OFF
5097 045314  004000                        .WORD   CC
5098 045316  001777                        .WORD   1777            ;CONTROL BITS DISABLED
5099 045320  001777                        .WORD   1777
5100 045322  001777                        .WORD   1777
5101 045324  001777                        .WORD   1777
5102
5103 045326  177777                        .WORD   -1              ;END OF TABLE
5104
5105 045330                         500$:                       ;END OF TEST
5106
5107                                ;;*****************************************************************
```

```
                                        ;*TEST 115      SEARCH TIMEOUT TEST
                                        ;;*****************************************************************
      045330                            TST115:
      045330  000004                            SCOPE                   ;SCOPE CALL
      045332  000240                            NOP
      045334  012706  001100                    MOV     #STACK,SP       ;LOAD THE STACK POINTER
      045340  013700  001276                    MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      045344  013701  001466                    MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
      045350  012737  000115  001226            MOV     #115,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 5108
 5109 045356  004737  054450                    JSR     PC,SETVV        ;GO SET VOLUME VALID
      045362  000402                            BR      10$             ;BRANCH TO 10$ IF NO ERROR
      045364  104000                            EMT
 5110 045366  000550                            BR      90$
 5111
 5112                                   ;ENABLE DEBUG CLOCK AND LOAD SEARCH COMMAND
 5113 045370                            10$:
 5114 045370  012760  041401  000024            MOV     #DMD.MUR.DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 5115 045376  012760  000000  000014            MOV     #0,RMER1(R0)    ;LOAD RMER1
 5116 045404  012760  000000  000042            MOV     #0,RMER2(R0)    ;LOAD RMER2
 5117 045412  012760  000000  000034            MOV     #0,RMDC(R0)     ;LOAD RMDC
 5118 045420  012760  000000  000006            MOV     #0,RMDA(R0)     ;LOAD RMDA
 5119 045426  012760  000031  000000            MOV     #SEARCH!GO,RMCS1(R0)    ;LOAD RMCS1
 5120
 5121                                   ;EXECUTE SEARCH TO TEST FOR ON LATCH RESET (17 CLOCKS)
 5122 045434  012702  000021                    MOV     #17.,R2
 5123 045440                            20$:
      045440  012760  141401  000024            MOV     #DMD!MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5124 045446  012760  041401  000024            MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 5125 045454  005302                            DEC     R2
 5126 045456  001370                            BNE     20$
 5127
 5128                                   ;DROP ON CYLINDER TO RESET LATCH, RAISE ON CYLINDER
 5129 045460  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 5130 045466  012760  041401  000024            MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
 5131
 5132                                   ;STEP COMMAND SEQUENCER TO SECTOR COMPARE LOOP (2 CLOCKS)
 5133 045474  012702  000002                    MOV     #2,R2
 5134 045500                            30$:
      045500  012760  151401  000024            MOV     #DMD!DBEN!MUR!MOC!DBCK!MSEN,RMMR1(R0)   ;LOAD RMMR1
 5135 045506  012760  051401  000024            MOV     #DMD!DBEN.MUR!MOC!MSEN,RMMR1(R0)        ;LOAD RMMR1
 5136 045514  005302                            DEC     R2
 5137 045516  001370                            BNE     30$
 5138
 5139                                   ;THE COMMAND SEQUENCER IS NOW IN THE SECTOR COMPARE LOOP.  STEP
 5140                                   ;THROUGH THE LOOP AND VERIFY SEARCH IS ENABLED.
 5141 045520  012702  000005                    MOV     #5,R2
 5142 045524                            40$:
      045524  012760  151401  000024            MOV     #DMD!DBEN!MUR!MOC!DBCK!MSEN,RMMR1(R0)   ;LOAD RMMR1
 5143 045532  012760  051401  000024            MOV     #DMD!DBEN!MUR!MOC!MSEN,RMMR1(R0)        ;LOAD RMMR1
 5144 045540  016037  000024  001142            MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 5145 045546  042737  173777  001142            BIC     #^CESRC,$BDDAT
 5146 045554  001403                            BEQ     50$             ;BRANCH IF SEARCH NOT ENABLED
 5147 045556  005302                            DEC     R2
 5148 045560  001361                            BNE     40$
 5149 045562  000412                            BR      60$
```

```
5150 045564  012737  004000  001140  50$:    MOV     #ESRC,$GDDAT    ;SETUP ERROR MSG
5151 045572  010037  001136           MOV     R0,$BDADR
5152 045576  062737  000024  001136   ADD     #RMMR1,$BDADR
5153 045604  104265                   EMT     265
5154 045606  000440                   BR      90$
5155
5156                                ;DROP MSEN TO ENABLE SEARCH TIMEOUT AND WAIT FOR OPI TO SET.
5157 045610                         60$:
5158 045610  012760  041401  000024   MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5159 045616  012737  070000  001534   MOV     #70000,WATCH    ;SET WATCHDOG TIMER VALUE
     045624  004777  133706           JSR     PC,@CLOCK       ;START THE CLOCK
5160 045630                         70$:
     045630  016037  000014  001142   MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
5161 045636  042737  157777  001142   BIC     #^COPI,$BDDAT
5162 045644  001017                   BNE     80$
5163 045646  005737  001534           TST     WATCH
5164 045652  001366                   BNE     70$
5165 045654  004777  133660           JSR     PC,@STOPCL      ;STOP THE CLOCK
5166 045660  012737  020000  001140   MOV     #OPI,$GDDAT     ;SETUP ERROR MSG
5167 045666  010037  001136           MOV     R0,$BDADR
5168 045672  062737  000014  001136   ADD     #RMER1,$BDADR
5169 045700  104267                   EMT     267
5170 045702  000402                   BR      90$
5171 045704                         80$:
     045704  004777  133630           JSR     PC,@STOPCL      ;STOP THE CLOCK
5172
5173 045710                         90$:                            ;END OF TEST
5174
5175                                ;;*****************************************************************
                                    ;*TEST 116       DATA COMMAND TESTS (1)

                                    ;;*****************************************************************
     045710                         TST116.
     045710  000004                        SCOPE                   ;SCOPE CALL
     045712  000240                        NOP
     045714  012706  001100                MOV     #STACK,SP       ;LOAD THE STACK POINTER
     045720  013700  001276                MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     045724  013701  001466                MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
     045730  012737  000116  001226        MOV     #116,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
5176
5177                                ;VERIFY DATA COMMAND SETS OPI IF DRIVE NOT READY
5178
5179 045736  004737  054450           JSR     PC,SETVV        ;GO SET VOLUME VALID
     045742  000402                   BR      10$             ;BRANCH TO 10$ IF NO ERROR
     045744  104000                   EMT
5180 045746  000471                   BR      40$
5181
5182                                ;ENABLE DEBUG CLOCK AND LOAD READ COMMAND
5183 045750                         10$:
5184 045750  012760  041401  000024   MOV     #DMD.MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5185 045756  012760  000000  000014   MOV     #0,RMER1(R0)    ;LOAD RMER1
5186 045764  012760  000000  000042   MOV     #0,RMER2(R0)    ;LOAD RMER2
5187 045772  012760  000000  000006   MOV     #0,RMDA(R0)     ;LOAD RMDA
5188 046000  012760  000000  000034   MOV     #0,RMDC(R0)     ;LOAD RMDC
5189 046006  012760  000071  000000   MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
5190
5191                                ;S EP COMMAND SEQUENCER TO UNIT READY TEST (3 CLOCKS)
```

```
 5192 046014  012702  000003            MOV     #3,R2
 5193 046020                       20$:
      046020  012760  141401  000024    MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5194 046026  012760  041401  000024    MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5195 046034  005302                    DEC     R2
 5196 046036  001370                    BNE     20$
 5197
 5198                               ;DROP UNIT READY
 5199 046040  012760  040401  000024    MOV     #DMD.MOC.DBEN,RMMR1(R0) ;LOAD RMMR1
 5200
 5201                               ;STEP SEQUENCER AND VERIFY OPI SETS (2 CLOCKS)
 5202 046046  012702  000002            MOV     #2,R2
 5203 046052                       30$:
      046052  012760  140401  000024    MOV     #DMD.MOC!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 5204 046060  012760  040401  000024    MOV     #DMD!MOC.DBEN,RMMR1(R0) ;LOAD RMMR1
 5205 046066  005302                    DEC     R2
 5206 046070  001370                    BNE     30$
 5207 046072  016037  000014  001142    MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
 5208 046100  042737  157777  001142    BIC     #^COPI,$BDDAT
 5209 046106  001011                    BNE     40$
 5210 046110  012737  020000  001140    MOV     #OPI,$GDDAT
 5211 046116  010037  001136            MOV     R0,$BDADR
 5212 046122  062737  000014  001136    ADD     #RMER1,$BDADR
 5213 046130  104270                    EMT     270
 5214
 5215 046132  012737  046140  001124  40$:    MOV     #50$,$LPERR     ;CHANGE LOOP ON ERROR TEST
 5216
 5217                               ;VERIFY DATA COMMAND ABORTS AT LOCATION 129
 5218 046140                       50$:
 5219 046140  004737  054450            JSR     PC,SETVV        ;GO SET VOLUME VALID
      046144  000402                    BR      60$             ;BRANCH TO 60$ IF NO ERROR
      046146  104000                    EMT
 5220 046150  000576                    BR      150$
 5221
 5222                               ;ENABLE DEBUG CLOCK AND LOAD READ COMMAND
 5223 046152                       60$:
 5224 046152  012760  041401  000024    MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5225 046160  012760  000000  000014    MOV     #0,RMER1(R0)    ;LOAD RMER1
 5226 046166  012760  000000  000042    MOV     #0,RMER2(R0)    ;LOAD RMER2
 5227 046174  012760  000000  000006    MOV     #0,RMDA(R0)     ;LOAD RMDA
 5228 046202  012760  000000  000034    MOV     #0,RMDC(R0)     ;LOAD RMDC
 5229 046210  012760  000071  000000    MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
 5230
 5231                               ;STEP COMMAND SEQUENCER TO ABORT TEST AT LOCATION 129 (4 CLOCKS)
 5232 046216  012702  000004            MOV     #4,R2
 5233 046222                       70$:
      046222  012760  141401  000024    MOV     #DMD!MUR.MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5234 046230  012760  041401  000024    MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5235 046236  005302                    DEC     R2
 5236 046240  001370                    BNE     70$
 5237
 5238                               ;SET DEVICE FAULT TO CAUSE ABORT CONDITION
 5239 046242  012760  041501  000024    MOV     #DMD.MUR!MOC.DBEN.MDF,RMMR1(R0) ;LOAD RMMR1
 5240
 5241                               ;STEP THE SEQUENCER THROUGH THE TEST FOR ABORT (1 CLOCK)
 5242 046250  012702  000001            MOV     #1,R2
 5243 046254                       80$:
```

```
                046254  012760  141501  000024          MOV     #DMD!MUR.MOC.DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
5244            046262  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5245            046270  005302                          DEC     R2
5246            046272  001370                          BNE     80$
5247
5248                                                ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
5249                                                ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
5250            046274  012702  000020                      MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
5251            046300                              85$:
                046300  012760  045501  000024          MOV     #DMD!MUR!MOC!DBEN.MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
5252            046306  012760  041501  000024          MOV     #DMD!MUR!MOC.DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5253            046314  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5254            046322  042737  157777  001142          BIC     #^CEBL,$BDDAT
5255            046330  001014                          BNE     90$             ;BRANCH IF EBL IS SET
5256
5257            046332  005302                          DEC     R2
5258            046334  001361                          BNE     85$             ;CONTINUE BIT CLOCKS IF COUNT NOT 0
5259            046336  012737  020000  001140          MOV     #EBL,$GDDAT
5260            046344  010037  001136                  MOV     R0,$BDADR
5261            046350  062737  000024  001136          ADD     #RMMR1,$BDADR
5262            046356  104271                          EMT     271
5263            046360  000472                          BR      150$
5264
5265                                                ;STEP THE SEQUENCER THROUGH ITS TEST FOR EBL (2 CLOCKS)
5266            046362                              90$:
5267            046362  012702  000002                  MOV     #2,R2
5268            046366                              100$:
5269            046366  012760  141501  000024          MOV     #DMD!MUR.MOC.DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
5270            046374  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5271            046402  005302                          DEC     R2
5272            046404  001370                          BNE     100$
5273
5274                                                ;ABORT EBL SHOULD NOW BE INACTIVE - FORCE BIT CLOCK USING THE
5275                                                ;MAINTENANCE REGISTER TO RESET EBL (16 BIT CLOCKS)
5276            046406  012702  000020                      MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
5277            046412                              110$:
                046412  012760  045501  000024          MOV     #DMD!MUR!MOC.DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
5278            046420  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5279            046426  005302                          DEC     R2
5280            046430  001370                          BNE     110$            ;ISSUE 16 BIT CLOCKS THEN TEST
5281
5282                                                ;VERIFY EBL IS NOW RESET
5283            046432                              120$:
5284            046432  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5285            046440  042737  157777  001142          BIC     #^CEBL,$BDDAT
5286            046446  001411                          BEQ     130$            ;BRANCH IF EBL IS RESET
5287            046450  005037  001140                  CLR     $GDDAT
5288            046454  010037  001136                  MOV     R0,$BDADR
5289            046460  062737  000024  001136          ADD     #RMMR1,$BDADR
5290            046466  104273                          EMT     273
5291            046470  000426                          BR      150$
5292
5293                                                ;VERIFY GO RESETS WITHIN 4 CLOCK CYCLES
5294            046472                              130$:
5295            046472  012702  000004                  MOV     #4,R2
5296            046476                              140$:
                046476  012760  141501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
```

```
5297 046504  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5298 046512  005302                          DEC     R2
5299 046514  001370                          BNE     140$
5300 046516  016037  000000  001142          MOV     RMCS1(R0),$BDDAT        ;STORE RMCS1 AT $BDDAT
5301 046524  042737  177776  001142          BIC     #^CGO,$BDDAT
5302 046532  001405                          BEQ     150$
5303 046534  005037  001140                  CLR     $GDDAT
5304 046540  010037  001136                  MOV     R0,$BDADR
5305 046544  104274                          EMT     274
5306
5307 046546  012737  046554  001124  150$:   MOV     #200$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
5308
5309                                  ;VERIFY SEQUENCER BRANCHES TO SEEK WHEN RUN AND GO FLOP SETS
5310 046554                          200$:
5311 046554  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
     046560  000402                          BR      210$            ;BRANCH TO 210$ IF NO ERROR
     046562  104000                          EMT
5312 046564  000512                          BR      250$
5313
5314                                  ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
5315 046566                          210$:
5316 046566  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5317 046574  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
5318 046602  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
5319 046610  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
5320 046616  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
5321 046624  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
5322
5323                                  ;MOVE SEQUENCER TO TEST FOR RUN AND GO AT LOCATION 130 (5 CLOCKS)
5324 046632  012702  000005                  MOV     #5,R2
5325 046636                          220$:
     046636  012760  141401  000024          MOV     #DMD!MUR!MOC.DBEN.DBCK,RMMR1(R0)        ;LOAD RMMR1
5326 046644  012760  041401  000024          MOV     #DMD!MUR!MOC.DBEN,RMMR1(R0)     ;LOAD RMMR1
5327 046652  005302                          DEC     R2
5328 046654  001370                          BNE     220$
5329
5330                                  ;VERIFY RUN AND GO IS SET
5331 046656  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5332 046664  042737  137777  001142          BIC     #^CRG,$BDDAT
5333 046672  001012                          BNE     230$
5334 046674  012737  040000  001140          MOV     #RG,$GDDAT
5335 046702  010037  001136                  MOV     R0,$BDADR
5336 046706  062737  000024  001136          ADD     #RMMR1,$BDADR
5337 046714  104275                          EMT     275
5338 046716  000435                          BR      250$
5339
5340                                  ;VERIFY THAT CYLINDER TAG COMES UP IN ONE CLOCK CYCLE
5341 046720                          230$:
5342 046720  012702  000001                  MOV     #1,R2
5343 046724                          240$:
     046724  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN.DBCK,RMMR1(R0)        ;LOAD RMMR1
5344 046732  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5345 046740  005302                          DEC     R2
5346 046742  001370                          BNE     240$
5347 046744  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
5348
5349                                  ;;*****************************************************************
```

```
                                        ;       BIC     #RQA!RQB!TAG,$BDDAT
                                                                        ;THE FOLLOWING CODE WAS ADDED
                                                                        ;TO ALLOW THE PROGRAM TO RUN WITH
                                                                        ;OR WITHOUT THE ECO TO THE CS BOARD.

046752  042737  162000  001142                BIC     #RQA.RQB!TAG!CH,$BDDAT  ;HARDWARE ECO CHANGE CAUSES CC AND
                                                                        ;CH TO SET AT THE SAME TIME
                                        ;;*****************************************************************

046760  012737  004000  001140                MOV     #CC,$GDDAT
046766  023737  001140  001142                CMP     $GDDAT,$BDDAT
046774  001406                               BEQ     250$
046776  010037  001136                       MOV     R0,$BDADR
047002  062737  000040  001136                ADD     #RMMR2,$BDADR
047010  104276                               EMT     276

047012  012737  047024  001124  250$:        MOV     #260$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS

                                        ;VERIFY DATA COMMAND ABORTS AT COMMAND SEQUENCER LOCATIONS 144, 145
047020  012702  000144                       MOV     #144,R2         ;INITIALIZE TEST LOCATION
047024                                  260$:
047024  004737  054450                       JSR     PC,SETVV        ;GO SET VOLUME VALID
047030  000402                               BR      270$            ;BRANCH TO 270$ IF NO ERROR
047032  104000                               EMT
047034  000553                               BR      320$

                                        ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
047036                                  270$:
047036  012760  041401  000024                MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
047044  012760  000000  000014                MOV     #0,RMER1(R0)    ;LOAD RMER1
047052  012760  000000  000042                MOV     #0,RMER2(R0)    ;LOAD RMER2
047060  012760  000000  000006                MOV     #0,RMDA(R0)     ;LOAD RMDA
047066  012760  000000  000034                MOV     #0,RMDC(R0)     ;LOAD RMDC
047074  012760  000071  000000                MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1

                                        ;WAIT FOR RUN AND GO TO SET
047102  012737  000310  001534                MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
047110  004777  132422                       JSR     PC,@CLOCK       ;START THE CLOCK
047114                                  280$:
047114  016037  000024  001142                MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
047122  042737  137777  001142                BIC     #^CRG,$BDDAT
047130  001017                               BNE     290$
047132  005737  001534                       TST     WATCH
047136  001366                               BNE     280$
047140  004777  132374                       JSR     PC,@STOPCL      ;STOP THE CLOCK
047144  012737  040000  001140                MOV     #RG,$GDDAT
047152  010037  001136                       MOV     R0,$BDADR
047156  062737  000024  001136                ADD     #RMMR1,$BDADR
047164  104275                               EMT     275
047166  000476                               BR      320$
047170                                  290$:
047170  004777  132344                       JSR     PC,@STOPCL      ;STOP THE CLOCK

                                        ;MOVE COMMAND SEQUENCER TO ABORT TEST (LOCATION 144 OR 145)
047174  012703  000006                       MOV     #6,R3           ;SETUP CLOCK COUNT
047200  022702  000144                       CMP     #144,R2
047204  001402                               BEQ     300$
```

```
5402 047206  012703  000007          MOV     #7,R3
5403 047212                  300$:
     047212  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
5404 047220  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5405 047226  005303                          DEC     R3
5406 047230  001370                          BNE     300$
5407
5408                         ;SET DRIVE FAULT TO FORCE ABORT CONDITION
5409 047232  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5410
5411                         ;CLOCK SEQUENCER THROUGH ITS TEST FOR ABORT (1 CLOCK)
5412 047240  012703  000001          MOV     #1,R3
5413 047244                  305$:
     047244  012760  141501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
5414 047252  012760  041501  000024          MOV     #DMD.MUR.MOC.DBEN.MDF,RMMR1(R0) ;LOAD RMMR1
5415 047260  005303                          DEC     R3
5416 047262  001370                          BNE     305$            ;ISSUE 2 CLOCKS
5417
5418                         ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
5419                         ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
5420 047264  012702  000020          MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
5421 047270                  306$:
     047270  012760  045501  000024          MOV     #DMD!MUR.MOC!DBEN.MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
5422 047276  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5423 047304  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5424 047312  042737  157777  001142          BIC     #^CEBL,$BDDAT
5425 047320  001013                          BNE     310$            ;BRANCH IF EBL IS SET
5426
5427 047322  005302                          DEC     R2
5428 047324  001361                          BNE     306$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
5429 047326  012737  020000  001140          MOV     #EBL,$GDDAT
5430 047334  010037  001136                  MOV     R0,$BDADR
5431 047340  062737  000024  001136          ADD     #RMMR1,$BDADR
5432 047346  104271                          EMT     271
5433 047350  022702  000144          310$:   CMP     #144,R2
5434 047354  001003                          BNE     320$
5435 047356  012702  000145                  MOV     #145,R2
5436 047362  000620                          BR      260$
5437
5438 047364  012737  047374  001124  320$:   MOV     #330$,$LPERR            ;CHANGE LOOP ON ERROR ADDRESS
5439
5440                         ;VERIFY HEAD TAG DURING DATA COMMAND
5441 047372  005002                  CLR     R2              ;INITIALIZE TRACK ADDRESS = 0
5442 047374                  330$:
5443 047374  004737  054450          JSR     PC,SETVV        ;GO SET VOLUME VALID
     047400  000402                  BR      340$            ;BRANCH TO 340$ IF NO ERROR
     047402  104000                  EMT
5444 047404  000570                  BR      400$
5445
5446                         ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
5447 047406                  340$:
5448 047406  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5449 047414  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
5450 047422  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
5451 047430  010260  000006                  MOV     R2,RMDA(R0)     ;LOAD RMDA
5452 047434  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
5453 047442  012760  000071  000000          MOV     #RD.GO,RMCS1(R0)        ;LOAD RMCS1
```

```
5454
5455                                    ;WAIT FOR RUN AND GO TO SET
5456 047450  012737  000310  001534            MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
     047456  004777  132054                    JSR     PC,@CLOCK       ;START THE CLOCK
5457 047462                             350$:
     047462  016037  000024  001142            MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5458 047470  042737  137777  001142            BIC     #^CRG,$BDDAT
5459 047476  001017                            BNE     360$
5460 047500  005737  001534                    TST     WATCH
5461 047504  001366                            BNE     350$
5462 047506  004777  132026                    JSR     PC,@STOPCL      ;STOP THE CLOCK
5463 047512  012737  040000  001140            MOV     #RG,$GDDAT
5464 047520  010037  001136                    MOV     R0,$BDADR
5465 047524  062737  000024  001136            ADD     #RMMR1,$BDADR
5466 047532  104275                            EMT     275
5467 047534  000514                            BR      400$
5468 047536                             360$:
     047536  004777  131776                    JSR     PC,@STOPCL      ;STOP THE CLOCK
5469
5470                                    ;STEP COMMAND SEQUENCER TO HEAD SEQUENCE, LOCATION 156 (17 CLOCKS)
5471 047542  012703  000021                    MOV     #17.,R3
5472 047546                             370$:
     047546  012760  141401  000024            MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
5473 047554  012760  041401  000024            MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5474 047562  005303                            DEC     R3
5475 047564  001370                            BNE     370$
5476
5477                                    ;DROP AND RAISE ON CYLINDER TO RESET ON LATCH
5478 047566  012760  041001  000024            MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
5479 047574  012760  041401  000024            MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5480
5481                                    ;;*****************************************************************
5482                                    ;       MOV     #450$,R3        ;INITIALIZE TABLE POINTER
5483
5484                                                                    ;HARDWARE ECO CHANGE TO THE PLA ON THE
5485                                                                    ;CS BOARD.
5486 047602  012703  000004                    MOV     #4,R3           ;CLOCK THE SEQUENCER THRU THE FIRST
5487                                                                    ;4 COMMAND SEQUENCES TO ALLOW THE PROGRAM
5488                                                                    ;TO RUN WITH OR WITHOUT THE ECO.
5489 047606                             372$:
5490 047606  012760  141401  000024            MOV     #DMD!DBEN!MUR!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
5491 047614  012760  041401  000024            MOV     #DMD!DBEN.MUR!MOC,RMMR1(R0)     ;LOAD RMMR1
5492 047622  005303                            DEC     R3              ;DONE 4 CLOCKS ?
5493 047624  001370                            BNE     372$            ;NO !!
5494 047626  012703  050000                    MOV     #475$,R3        ;INITIALIZE NEW TABLE POINTER
5495                                    ;;*****************************************************************
5496
5497                                    ;VERIFY TAG BUS ACCORDING TO TABLE AND TRACK ADDRESS IN R2
5498 047632                             375$:
5499 047632  016037  000040  001142            MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
5500 047640  042737  150000  001142            BIC     #RQA!RQB!TST,$BDDAT
5501 047646  011337  001140                    MOV     (R3),$GDDAT
5502 047652  010204                            MOV     R2,R4           ;GENERATE EXPECTED TAG BUS
5503 047654  000304                            SWAB    R4
5504 047656  050437  001140                    BIS     R4,$GDDAT
5505 047662  023737  001140  001142            CMP     $GDDAT,$BDDAT
5506 047670  001030                            BNE     390$
```

SEQ 0209

```
 5507 047672  012760  141401  000024          MOV     #DMD.MUR.MOC!DBEN.DBCK,RMMR1(R0)        ;LOAD RMMR1
 5508 047700  012760  041401  000024          MOV     #DMD.MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5509
 5510                                 ;ADVANCE TO NEXT TABLE ENTRY
 5511 047706  062703  000002                  ADD     #2,R3
 5512 047712  005713                          TST     (R3)
 5513 047714  100401                          BMI     380$
 5514 047716  000745                          BR      375$
 5515
 5516                                 ;SHIFT TO NEXT TRACK ADDRESS-EXIT LOOP IF DONE
 5517 047720                          380$:
 5518 047720  062702  000400                  ADD     #TA1,R2         ;ADVANCE TRACK ADDRESS
 5519 047724  032737  010000  001334          BIT     #TA16,LSTRK     ;IS IT RM05 ?
 5520 047732  001403                          BEQ     385$            ;NO !!
 5521 047734  020237  001334                  CMP     R2,LSTRK        ;DONE WITH TRACKS ON RM05 ?
 5522 047740  101615                          BLOS    330$            ;NO !!
 5523 047742                          385$:
 5524 047742  020237  001334                  CMP     R2,LSTRK        ;DONE WITH TRACKS ON RM02/3 ?
 5525 047746  101007                          BHI     400$            ;YES, EXIT
 5526 047750  000611                          BR      330$
 5527
 5528                                 ;ERROR ON TAG BUS DURING HEAD SEQUENCE
 5529 047752                          390$:
 5530 047752  010037  001136                  MOV     R0,$BDADR
 5531 047756  062737  000040  001136          ADD     #RMMR2,$BDADR
 5532 047764  104276                          EMT     276
 5533
 5534 047766  000440                  400$:   BR      500$            ;JUMP OVER TABLE
 5535
 5536                                 ;TABLE OF TAG BUS DURING HEAD SEQUENCE
 5537 047770                          450$:
 5538 047770  002000                          .WORD   CH
 5539 047772  002000                          .WORD   CH
 5540 047774  022000                          .WORD   CH!TAG
 5541 047776  022000                          .WORD   CH!TAG
 5542 050000                          475$:                           ;START TABLE HERE FOR HARDWARE ECO CHANGE
 5543 050000  022000                          .WORD   CH!TAG
 5544 050002  022000                          .WORD   CH!TAG
 5545 050004  022000                          .WORD   CH!TAG
 5546 050006  022000                          .WORD   CH!TAG
 5547 050010  177777                          .WORD   -1              ;END TABLE HERE FOR HARDWARE ECO CHANGE
 5548
 5549                                 ;       .WORD   CH
 5550 050012  002000                          .WORD   CH
 5551 050014  001777                          .WORD   1777
 5552 050016  001777                          .WORD   1777
 5553 050020  001777                          .WORD   1777
 5554 050022  001777                          .WORD   1777
 5555 050024  001777                          .WORD   1777
 5556 050026  001777                          .WORD   1777
 5557 050030  001777                          .WORD   1777
 5558 050032  001777                          .WORD   1777
 5559 050034  001777                          .WORD   1777
 5560 050036  001777                          .WORD   1777
 5561 050040  001777                          .WORD   1777
 5562 050042  001777                          .WORD   1777
 5563 050044  001777                          .WORD   1777
```

```
5564 050046  001777                    .WORD   1777
5565 050050  001777                    .WORD   1777
5566 050052  001777                    .WORD   1777
5567 050054  001777                    .WORD   1777
5568 050056  001777                    .WORD   1777
5569 050060  001777                    .WORD   1777
5570 050062  001777                    .WORD   1777
5571 050064  001777                    .WORD   1777
5572
5573 050066  177777                    .WORD   -1              ;END OF TABLE
5574
5575 050070                    500$:                           ;END OF TEST
5576
5577                           ;;*****************************************************************
                               ;*TEST 117       DATA COMMAND TESTS (2)

                               ;;*****************************************************************
     050070                    TST117:
     050070  000004                   SCOPE                   ;SCOPE CALL
     050072  000240                   NOP
     050074  012706  001100           MOV     #STACK,SP       ;LOAD THE STACK POINTER
     050100  013700  001276           MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
     050104  013701  001466           MOV     TSTQUE,R1       ;R1 = POINTER TO DEVICE
     050110  012737  000117  001226   MOV     #117,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
5578
5579                           ;VERIFY OPI SETS IF ON CYLINDER LATCH DOESNT RESET
5580
5581 050116  004737  054450           JSR     PC,SETVV        ;GO SET VOLUME VALID
     050122  000402                   BR      10$             ;BRANCH TO 10$ IF NO ERROR
     050124  104000                   EMT
5582 050126  000514                   BR      60$
5583
5584                           ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND DURING CYLINDER SEQUENCE
5585 050130                    10$:
5586 050130  012760  041401  000024   MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5587 050136  012760  000000  000014   MOV     #0,RMER1(R0)    ;LOAD RMER1
5588 050144  012760  000000  000042   MOV     #0,RMER2(R0)    ;LOAD RMER2
5589 050152  012760  000000  000006   MOV     #0,RMDA(R0)     ;LOAD RMDA
5590 050160  012760  000000  000034   MOV     #0,RMDC(R0)     ;LOAD RMDC
5591 050166  012760  000071  000000   MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
5592
5593                           ;WAIT FOR RUN AND GO TO SET
5594 050174  012737  000310  001534   MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
     050202  004777  131330           JSR     PC,@CLOCK       ;START THE CLOCK
5595 050206                    20$:
     050206  016037  000024  001142   MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5596 050214  042737  137777  001142   BIC     #^CRG,$BDDAT
5597 050222  001017                   BNE     30$
5598 050224  005737  001534           TST     WATCH
5599 050230  001366                   BNE     20$
5600 050232  004777  131302           JSR     PC,@STOPCL      ;STOP THE CLOCK
5601 050236  012737  040000  001140   MOV     #RG,$GDDAT
5602 050244  010037  001136           MOV     R0,$BDADR
5603 050250  062737  000024  001136   ADD     #RMMR1,$BDADR
5604 050256  104275                   EMT     275
5605 050260  000437                   BR      60$
5606 050262                    30$:
```

```
                050262  004777  131252                  JSR     PC,@STOPCL      ;STOP THE CLOCK
5607
5608                                            ;STEP COMMAND SEQUENCER AND VERIFY OPI SETS (19 CLOCKS)
5609    050266  012702  000023                  MOV     #19.,R2
5610    050272                                  40$:
                050272  012760  141401  000024          MOV     #DMD!MUR!MOC.DBEN.DBCK,RMMR1(R0)        ;LOAD RMMR1
5611    050300  012760  041401  000024          MOV     #DMD.MUR.MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5612    050306  005302                          DEC     R2
5613    050310  001370                          BNE     40$
5614    050312  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
5615    050320  042737  157777  001142          BIC     #^COPI,$BDDAT
5616    050326  001011                          BNE     50$             ;BRANCH IF OPI SET
5617    050330  012737  020000  001140          MOV     #OPE,$GDDAT
5618    050336  010037  001136                  MOV     R0,$BDADR
5619    050342  062737  000014  001136          ADD     #RMER1,$BDADR
5620    050350  104277                          EMT     277
5621    050352  012737  050360  001124  50$:    MOV     #60$,$LPERR     ;CHANGE LOOP ON ERROR ADDRESS
5622
5623                                            ;VERIFY DATA COMMAND ABORTS DURING SEEK WAIT LOOP
5624    050360                                  60$:
5625    050360  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
                050364  000402                  BR      70$             ;BRANCH TO 70$ IF NO ERROR
                050366  104000                  EMT
5626    050370  000567                          BR      140$
5627
5628                                            ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
5629    050372                                  70$:
5630    050372  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5631    050400  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
5632    050406  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
5633    050414  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
5634    050422  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
5635    050430  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
5636
5637                                            ;WAIT FOR RUN & GO TO SET
5638    050436  012737  000310  001534          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
                050444  004777  131066          JSR     PC,@CLOCK       ;START THE CLOCK
5639    050450                                  80$:
                050450  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5640    050456  042737  137777  001142          BIC     #^CRG,$BDDAT
5641    050464  001017                          BNE     90$
5642    050466  005737  001534                  TST     WATCH
5643    050472  001366                          BNE     80$
5644    050474  004777  131040                  JSR     PC,@STOPCL      ;STOP THE CLOCK
5645    050500  012737  040000  001140          MOV     #RG,$GDDAT
5646    050506  010037  001136                  MOV     R0,$BDADR
5647    050512  062737  000024  001136          ADD     #RMMR1,$BDADR
5648    050520  104275                          EMT     275
5649    050522  000512                          BR      140$
5650    050524                                  90$:
                050524  004777  131010                  JSR     PC,@STOPCL      ;STOP THE CLOCK
5651
5652                                            ;STEP COMMAND SEQUENCER TO ON LATCH TEST AT LOCATION 156 (17 CLOCKS)
5653    050530  012702  000021                  MOV     #17.,R2
5654    050534                                  100$:
                050534  012760  141401  000024          MOV     #DMD!MUR!MOC.DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
5655    050542  012760  041401  000024          MOV     #DMD.MUR!MOC.DBEN,RMMR1(R0)     ;LOAD RMMR1
```

```
5656 050550  005302                        DEC     R2
5657 050552  001370                        BNE     100$
5658
5659                                ;DROP ON CYLINDER TO RESET LATCH
5660 050554  012760  041001  000024        MOV     #DMD.MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
5661
5662                                ;MOVE COMMAND SEQUENCER TO SEEK WAIT LOOP (31 CLOCKS)
5663 050562  012702  000037                MOV     #31.,R2
5664 050566                         110$:
     050566  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ,LOAD RMMR1
5665 050574  012760  041001  000024        MOV     #DMD.MUR.DBEN,RMMR1(R0) ·LOAD RMMR1
5666 050602  005302                        DEC     R2
5667 050604  001370                        BNE     110$
5668
5669                                ;STEP THROUGH SEEK WAIT LOOP (6 CLOCKS) 2 TIMES
5670 050606  012702  000006                MOV     #6.,R2
5671 050612                         120$:
     050612  012760  141001  000024        MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
5672 050620  012760  041001  000024        MOV     #DMD!MUR.DBEN,RMMR1(R0) ;LOAD RMMR1
5673 050626  005302                        DEC     R2
5674 050630  001370                        BNE     120$
5675
5676                                ;SET SEEK INCOMPLETE ERROR TO CAUSE ABORT
5677 050632  012760  041201  000024        MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
5678
5679                                ;CLOCK THE SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
5680 050640  012702  000002                MOV     #2,R2
5681 050644                         130$:
     050644  012760  141201  000024        MOV     #DMD!MUR!DBEN!MSER!DBCK,RMMR1(R0)       ;LOAD RMMR1
5682 050652  012760  041201  000024        MOV     #DMD!MUR!DBEN!MSER,RMMR1(R0)    ;LOAD RMMR1
5683 050660  005302                        DEC     R2
5684 050662  001370                        BNE     130$
5685
5686                                ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
5687                                ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
5688 050664  012702  000020                MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
5689 050670                         135$:
     050670  012760  045501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
5690 050676  012760  041501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
5691 050704  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5692 050712  042737  157777  001142        BIC     #^CEBL,$BDDAT
5693 050720  001013                        BNE     140$
5694
5695 050722  005302                        DEC     R2
5696 050724  001361                        BNE     135$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
5697 050726  012737  020000  001140        MOV     #EBL,$GDDAT
5698 050734  010037  001136                MOV     R0,$BDADR
5699 050740  062737  000024  001136        ADD     #RMMR1,$BDADR
5700 050746  104300                        EMT     300
5701 050750  012737  050756  001124  140$:  MOV     #150$,$LPERR            ;CHANGE LOOP ON ERROR ADDRESS
5702
5703                                ;VERIFY DATA COMMAND ABORTS DURING OFFSET IF ON CYLINDER LATCH
5704                                ;DOESNT RESET
5705 050756                         150$:
5706 050756  004737  054450                JSR     PC,SETVV        ;GO SET VOLUME VALID
     050762  000402                        BR      160$            ;BRANCH TO 160$ IF NO ERROR
     050764  104000                        EMT
```

```
5707 050766  000536                        BR      220$
5708
5709                                ;LOAD TRACK, SECTOR, AND CYLINDER ADDRESSES
5710 050770                         160$:
5711 050770  012760  000000  000034        MOV     #0,RMDC(R0)     ;LOAD RMDC
5712 050776  012760  000000  000006        MOV     #0,RMDA(R0)     ;LOAD RMDA
5713 051004  004737  054572                JSR     PC,SETOM        ;GO SET OFFSET MODE
     051010  000402                        BR      170$            ;BRANCH TO 170$ IF NO ERROR
     051012  104000                        EMT
5714 051014  000523                        BR      220$
5715
5716                                ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
5717 051016                         170$:
5718 051016  012760  041401  000024        MOV     #DMD.MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5719 051024  012760  000000  000014        MOV     #0,RMER1(R0)    ;LOAD RMER1
5720 051032  012760  000000  000042        MOV     #0,RMER2(R0)    ;LOAD RMER2
5721 051040  012760  000071  000000        MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
5722
5723                                ;WAIT FOR RUN AND GO TO SET
5724 051046  012737  000310  001534        MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
     051054  004777  130456                JSR     PC,@CLOCK       ;START THE CLOCK
5725 051060                         180$:
     051060  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
5726 051066  042737  137777  001142        BIC     #^CRG,$BDDAT
5727 051074  001017                        BNE     190$
5728 051076  005737  001534                TST     WATCH
5729 051102  001366                        BNE     180$
5730 051104  004777  130430                JSR     PC,@STOPCL      ;STOP THE CLOCK
5731 051110  012737  040000  001140        MOV     #RG,$GDDAT
5732 051116  010037  001140                MOV     R0,$GDDAT
5733 051122  062737  000024  001136        ADD     #RMMR1,$BDADR
5734 051130  104275                        EMT     275
5735 051132  000454                        BR      220$
5736 051134                         190$:
     051134  004777  130400                JSR     PC,@STOPCL      ;STOP THE CLOCK
5737
5738                                ;STEP COMMAND SEQUENCER TO ON LATCH TEST AT LOCATION 156 (17 CLOCKS)
5739 051140  012702  000021                MOV     #17.,R2
5740 051144                         200$:
     051144  012760  141401  000024        MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
5741 051152  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
5742 051160  005302                        DEC     R2
5743 051162  001370                        BNE     200$
5744
5745                                ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
5746                                ;AT LOCATION 166
5747 051164  012760  041001  000024        MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
5748 051172  012760  041401  000024        MOV     #DMD!MUR!DBEN!MOC,RMMR1(R0)     ;LOAD RMMR1
5749
5750                                ;MOVE SEQUENCER TO SET OPI AND EBL (39 CLOCKS)
5751 051200  012702  000047                MOV     #39.,R2
5752 051204                         210$:
     051204  012760  141401  000024        MOV     #DMD.MUR!DBEN!MOC!DBCK,RMMR1(R0)        ;LOAD RMMR1
5753 051212  012760  041401  000024        MOV     #DMD.MUR.DBEN.MOC,RMMR1(R0)     ;LOAD RMMR1
5754 051220  005302                        DEC     R2
5755 051222  001370                        BNE     210$
5756
```

```
 5757                                   ;VERIFY OPI IS SET
 5758 051224  016037  000014  001142          MOV     RMER1(R0),$BDDAT        ;STORE RMER1 AT $BDDAT
 5759 051232  042737  157777  001142          BIC     #^COPI,$BDDAT
 5760 051240  001011                          BNE     220$
 5761 051242  012737  020000  001140          MOV     #OPI,$GDDAT
 5762 051250  010037  001136                  MOV     R0,$BDADR
 5763 051254  062737  000014  001136          ADD     #RMER1,$BDADR
 5764 051262  104277                          EMT     277
 5765 051264  012737  051272  001124  220$:   MOV     #230$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 5766
 5767                                   ;VERIFY DATA COMMAND ABORTS DURING OFFSET WAIT LOOP
 5768
 5769 051272                            230$:
      051272  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      051276  000403                          BR      240$            ;BRANCH TO 240$ IF NO ERROR
      051300  104000                          EMT
 5770 051302  000137  051712                  JMP     310$
 5771
 5772                                   ;LOAD SECTOR,TRACK AND CYLINDER ADDRESS
 5773 051306                            240$:
 5774 051306  012760  000000  000006          MOV     #0,RMDA(R0)     ;LOAD RMDA
 5775 051314  012760  000000  000034          MOV     #0,RMDC(R0)     ;LOAD RMDC
 5776 051322  004737  054572                  JSR     PC,SETOM        ;GO SET OFFSET MODE
      051326  000402                          BR      245$            ;BRANCH TO 245$ IF NO ERROR
      051330  104000                          EMT
 5777 051332  000567                          BR      310$
 5778
 5779                                   ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
 5780 051334                            245$:
      051334  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5781 051342  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 5782 051350  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
 5783 051356  012760  000071  000000          MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1
 5784
 5785                                   ;WAIT FOR RUN AND GO TO SET
 5786 051364  012737  000310  001534          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
      051372  004777  130140                  JSR     PC,@CLOCK       ;START THE CLOCK
 5787 051376                            250$:
      051376  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 5788 051404  042737  137777  001142          BIC     #^CRG,$BDDAT
 5789 051412  001017                          BNE     260$
 5790 051414  005737  001534                  TST     WATCH
 5791 051420  001366                          BNE     250$
 5792 051422  004777  130112                  JSR     PC,@STOPCL      ;STOP THE CLOCK
 5793 051426  012737  040000  001140          MOV     #RG,$GDDAT
 5794 051434  010037  001136                  MOV     R0,$BDADR
 5795 051440  062737  000024  001136          ADD     #RMMR1,$BDADR
 5796 051446  104275                          EMT     275
 5797 051450  000520                          BR      310$
 5798 051452                            260$:
      051452  004777  130062                  JSR     PC,@STOPCL      ;STOP THE CLOCK
 5799
 5800                                   ;STEP SEQUENCER TO LOCATION 156 (17 CLOCKS)
 5801 051456  012702  000021                  MOV     #17.,R2
 5802 051462                            270$:
      051462  012760  141401  000024          MOV     #DMD.MUR.MOC.DBEN.DBCK,RMMR1(R0)        ;LOAD RMMR1
 5803 051470  012760  041401  000024          MOV     #DMD.MUR.MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
```

```
 5804 051476  005302                          DEC     R2
 5805 051500  001370                          BNE     270$
 5806
 5807                                 ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
 5808                                 ;AT LOCATION 166
 5809 051502  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 5810 051510  012760  041401  000024          MOV     #DMD.MUR.MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5811
 5812                                 ;MOVE SEQUENCER TO LOCATION  174 (37 CLOCKS)
 5813 051516  012702  000045                  MOV     #37.,R2
 5814 051522                          280$:
      051522  012760  141401  000024          MOV     #DMD.MUR!MOC!DBEN.DBCK,RMMR1(R0)        ;LOAD RMMR1
 5815 051530  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5816 051536  005302                          DEC     R2
 5817 051540  001370                          BNE     280$
 5818
 5819                                 ;DROP ON CYLINDER TO RESET LATCH, LEAVE ON CYLINDER RESET
 5820 051542  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 5821
 5822                                 ;STEP SEQUENCER THROUGH OFFSET WAIT LOOP TWICE (7 CLOCKS)
 5823 051550  012702  000007                  MOV     #7.,R2
 5824 051554                          290$:
      051554  012760  141001  000024          MOV     #DMD!MUR!DBEN!DBCK,RMMR1(R0)    ;LOAD RMMR1
 5825 051562  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 5826 051570  005302                          DEC     R2
 5827 051572  001370                          BNE     290$
 5828
 5829                                 ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
 5830 051574  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
 5831
 5832                                 ;STEP SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
 5833 051602  012702  000002                  MOV     #2,R2
 5834 051606                          300$:
      051606  012760  141101  000024          MOV     #DMD!MUR!DBEN!MDF!DBCK,RMMR1(R0)        ;LOAD RMMR1
 5835 051614  012760  041101  000024          MOV     #DMD!MUR!DBEN!MDF,RMMR1(R0)     ;LOAD RMMR1
 5836 051622  005302                          DEC     R2
 5837 051624  001370                          BNE     300$
 5838
 5839                                 ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
 5840                                 ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
 5841 051626  012702  000020                  MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
 5842 051632                          305$:
      051632  012760  045501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
 5843 051640  012760  041501  000024          MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
 5844 051646  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 5845 051654  042737  157777  001142          BIC     #^CEBL,$BDDAT
 5846 051662  001013                          BNE     310$
 5847
 5848 051664  005302                          DEC     R2
 5849 051666  001361                          BNE     305$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
 5850 051670  012737  020000  001140          MOV     #EBL,$GDDAT
 5851 051676  010037  001136                  MOV     R0,$BDADR
 5852 051702  062737  000024  001136          ADD     #RMMR1,$BDADR
 5853 051710  104271                          EMT     271
 5854
 5855 051712  012737  051720  001124  310$:   MOV     #320$,$LPERR    ;CHANGE LOOP ON ERROR ADDRESS
 5856
```

```
                                        ;VERIFY THAT DATA COMMAND ABORTS DURING SECTOR WAIT LOOP AT LOCATION 179

 051720                                 320$:
 051720  004737  054450                        JSR     PC,SETVV        ;GO SET VOLUME VALID
 051724  000403                                BR      330$            ;BRANCH TO 330$ IF NO ERROR
 051726  104000                                EMT
 051730  000137  052356                        JMP     420$

                                        ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
 051734                                 330$:
 051734  012760  041401  000024                MOV     #DMD!MUR!MOC.DBEN,RMMR1(R0)     ;LOAD RMMR1
 051742  012760  000000  000014                MOV     #0,RMER1(R0)    ;LOAD RMER1
 051750  012760  000000  000042                MOV     #0,RMER2(R0)    ;LOAD RMER2
 051756  012760  000000  000006                MOV     #0,RMDA(R0)     ;LOAD RMDA
 051764  012760  000000  000034                MOV     #0,RMDC(R0)     ;LOAD RMDC
 051772  012760  000071  000000                MOV     #RD!GO,RMCS1(R0)        ;LOAD RMCS1

                                        ;WAIT FOR RUN AND GO TO SET
 052000  012737  000310  001534                MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
 052006  004777  127524                        JSR     PC,@CLOCK       ;START THE CLOCK
 052012                                 340$:
 052012  016037  000024  001142                MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 052020  042737  137777  001142                BIC     #^CRG,$BDDAT
 052026  001017                                BNE     350$
 052030  005737  001534                        TST     WATCH
 052034  001366                                BNE     340$
 052036  004777  127476                        JSR     PC,@STOPCL      ;STOP THE CLOCK
 052042  012737  040000  001140                MOV     #RG,$GDDAT
 052050  010037  001136                        MOV     R0,$BDADR
 052054  062737  000024  001136                ADD     #RMMR1,$BDADR
 052062  104275                                EMT     275
 052064  000534                                BR      420$
 052066                                 350$:
 052066  004777  127446                        JSR     PC,@STOPCL      ;STOP THE CLOCK

                                        ;STEP SEQUENCER TO LOCATION 156 (17 CLOCKS)
 052072  012702  000021                        MOV     #17.,R2
 052076                                 360$:
 052076  012760  141401  000024                MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 052104  012760  041401  000024                MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 052112  005302                                DEC     R2
 052114  001370                                BNE     360$

                                        ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST
                                        ;AT LOCATION 166
 052116  012760  041001  000024                MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1

                                        ;MOVE SEQUENCER TO SECTOR WAIT (34 CLOCKS)
 052124  012702  000042                        MOV     #34.,R2
 052130                                 370$:
 052130  012760  141401  000024                MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 052136  012760  041401  000024                MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 052144  005302                                DEC     R2
 052146  001370                                BNE     370$

                                        ;STEP THROUGH SECTOR WAIT LOOP TWICE AND VERIFY SEARCH IS ENABLED
                                        ;DURING THE LOOP (6 CLOCKS)
```

```
 5906 052150  012702  000006                MOV     #6.,R2
 5907 052154                          380$:
      052154  012760  141401  000024        MOV     #DMD.MUR.MOC!DBEN.DBCK,RMMR1(R0)        ;LOAD RMMR1
 5908 052162  012760  041401  000024        MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 5909 052170  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 5910 052176  042737  173777  001142        BIC     #^CESRC,$BDDAT
 5911 052204  001403                        BEQ     390$
 5912 052206  005302                        DEC     R2
 5913 052210  001361                        BNE     380$
 5914 052212  000412                        BR      400$
 5915 052214  012737  004000  001140  390$: MOV     #ESRC,$GDDAT
 5916 052222  010037  001136                MOV     R0,$BDADR
 5917 052226  062737  000024  001136        ADD     #RMMR1,$BDADR
 5918 052234  104301                        EMT     301
 5919 052236  000447                        BR      420$
 5920
 5921                                 ;SET DRIVE FAULT TO CAUSE ABORT CONDITION
 5922 052240                          400$:
      052240  012760  041501  000024        MOV     #DMD!MUR.MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
 5923
 5924                                 ;STEP SEQUENCER THROUGH ITS TEST FOR ABORT (2 CLOCKS)
 5925 052246  012702  000002                MOV     #2,R2
 5926 052252                          410$:
      052252  012760  141501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF!DBCK,RMMR1(R0)    ;LOAD RMMR1
 5927 052260  012760  041501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
 5928 052266  005302                        DEC     R2
 5929 052270  001370                        BNE     410$
 5930
 5931                                 ;ABORT EBL SHOULD NOW BE ACTIVE - USE THE MAINTENANCE REGISTER TO
 5932                                 ;FORCE BIT CLOCKS AND VERIFY THAT EBL SETS WITHIN 16 BIT CLOCKS
 5933 052272  012702  000020                MOV     #16.,R2         ;MAXIMUM NUMBER OF BIT CLOCKS
 5934 052276                          415$:
      052276  012760  045501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF!MCLK,RMMR1(R0)    ;LOAD RMMR1
 5935 052304  012760  041501  000024        MOV     #DMD!MUR!MOC!DBEN!MDF,RMMR1(R0) ;LOAD RMMR1
 5936
 5937                                 ;VERIFY EBL IS SET
 5938 052312  016037  000024  001142        MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 5939 052320  042737  157777  001142        BIC     #^CEBL,$BDDAT
 5940 052326  001013                        BNE     420$
 5941
 5942 052330  005302                        DEC     R2
 5943 052332  001361                        BNE     415$            ;CONTINUE BIT CLOCKS IF COUNT NOT 0
 5944 052334  012737  020000  001140        MOV     #EBL,$GDDAT
 5945 052342  010037  001136                MOV     R0,$BDADR
 5946 052346  062737  000024  001136        ADD     #RMMR1,$BDADR
 5947 052354  104271                        EMT     271
 5948
 5949 052356                          420$:                          ;END OF TEST
 5950
 5951                                 ;;*****************************************************************
                                      ;*TEST 120       DATA COMMAND TESTS (3)
                                      ;;*****************************************************************
      052356                          TST120:
      052356  000004                         SCOPE                   ;SCOPE CALL
      052360  000240                         NOP
      052362  012706  001100                 MOV     #STACK,SP       ;LOAD THE STACK POINTER
```

```
      052366  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS ADDRESS
      052372  013701  001466                  MOV     TSTQUE,R1       ;R1 - POINTER TO DEVICE
      052376  012737  000120  001226          MOV     #120,$TESTN     ;;SET TEST NUMBER IN APT MAIL BOX
 5952
 5953                                 ;VERIFY THE TAG BUS DURING DATA COMMAND
 5954                                 ;FIRST PART USES OFFSET FORWARD
 5955
 5956                                 ;LOAD TEST PARAMETERS IN REGISTER OUTPUT BUFFER
 5957 052404  012737  001466  001446          MOV     #822.,RMDC0     ;LAST CYLINDER
 5958 052412  013737  001334  001420          MOV     LSTRK,RMDA0     ;SETUP LAST TRACK AND
 5959 052420  112737  000035  001420          MOVB    #29.,RMDA0      ;LAST SECTOR
 5960 052426                                 5$:
 5961 052426  012737  000200  001444          MOV     #OFD,RMOF0      ;FORWARD OFFSET
 5962 052434  012737  000071  001412          MOV     #RD.GO,RMCS10   ;READ DATA
 5963 052442  012737  177400  001414          MOV     #-256.,RMWC0    ;WORD COUNT
 5964 052450  012737  104312  001416          MOV     #BUFFER,RMBA0   ;BUFFER ADDRESS
 5965
 5966                                 ;EXECUTE COMMAND AND VERIFY TAG BUS USING SUBROUTINE
 5967 052456  004737  052500                  JSR     PC,10$
 5968
 5969                                 ;SECOND PART USES OFFSET REVERSE
 5970                                 ;LOAD TEST PARAMETERS IN REGISTER OUTPUT BUFFER
 5971 052462  112737  000000  001444          MOVB    #0,RMOF0        ;REVERSE OFFSET
 5972
 5973                                 ;EXECUTE COMMAND AND VERIFY TAG BUS USING SUBROUTINE
 5974 052470  004737  052500                  JSR     PC,10$
 5975 052474  000137  053564                  JMP     300$
 5976
 5977                                 ;;*****************************************************************
 5978                                 ;SUBROUTINE USED DURING TEST
 5979                                 ;;*****************************************************************
 5980
 5981 052500                         10$:
 5982 052500  004737  054450                  JSR     PC,SETVV        ;GO SET VOLUME VALID
      052504  000403                          BR      20$             ;BRANCH TO 20$ IF NO ERROR
      052506  104000                          EMT
 5983 052510  000137  053402                  JMP     160$
 5984
 5985                                 ;LOAD TRACK, SECTOR AND CYLINDER ADDRESS, LOAD OFFSET
 5986 052514                         20$:
 5987 052514  013760  001420  000006          MOV     RMDA0,RMDA(R0)  ;LOAD RMDA
 5988 052522  013760  001446  000034          MOV     RMDC0,RMDC(R0)  ;LOAD RMDC
 5989 052530  013760  001444  000032          MOV     RMOF0,RMOF(R0)  ;LOAD RMOF
 5990 052536  004737  054572                  JSR     PC,SETOM        ;GO SET OFFSET MODE
      052542  000403                          BR      30$             ;BRANCH TO 30$ IF NO ERROR
      052544  104000                          EMT
 5991 052546  000137  053402                  JMP     160$
 5992
 5993                                 ;LOAD BUFFER ADDRESS AND WORD COUNT
 5994 052552                         30$:
 5995 052552  013760  001414  000002          MOV     RMWC0,RMWC(R0)  ;LOAD RMWC
 5996 052560  013760  001416  000004          MOV     RMBA0,RMBA(R0)  ;LOAD RMBA
 5997
 5998                                 ;ENABLE DEBUG CLOCK AND LOAD DATA COMMAND
 5999 052566  012760  041401  000024          MOV     #DMD!MUR!MOC.DBEN,RMMR1(R0)     ;LOAD RMMR1
 6000 052574  012760  000000  000014          MOV     #0,RMER1(R0)    ;LOAD RMER1
 6001 052602  012760  000000  000042          MOV     #0,RMER2(R0)    ;LOAD RMER2
```

```
 6002 052610  013760  001412  000000          MOV     RMCS10,RMCS1(R0)        ;LOAD RMCS1
 6003
 6004                                  ;WAIT FOR RUN AND GO TO SET
 6005 052616  012737  000310  001534          MOV     #200.,WATCH     ;SET WATCHDOG TIMER VALUE
      052624  004777  126706                  JSR     PC,@CLOCK       ;START THE CLOCK
 6006 052630                           40$:
      052630  016037  000024  001142          MOV     RMMR1(R0),$BDDAT        ;STORE RMMR1 AT $BDDAT
 6007 052636  042737  137777  001142          BIC     #^CRG,$BDDAT
 6008 052644  001020                          BNE     50$
 6009 052646  005737  001534                  TST     WATCH
 6010 052652  001366                          BNE     40$
 6011 052654  004777  126660                  JSR     PC,@STOPCL      ;STOP THE CLOCK
 6012 052660  012737  040000  001140          MOV     #RG,$GDDAT
 6013 052666  010037  001136                  MOV     R0,$BDADR
 6014 052672  062737  000024  001136          ADD     #RMMR1,$BDADR
 6015 052700  104275                          EMT     275
 6016 052702  000137  053402                  JMP     160$
 6017 052706                           50$:
      052706  004777  126626                  JSR     PC,@STOPCL      ;STOP THE CLOCK
 6018 052712  012704  053404                  MOV     #200$,R4        ;R4 = TABLE POINTER
 6019
 6020                                  ;STEP SEQUENCER TO HEAD SEQUENCE AT LOCATION 156 (17 CLOCKS)
 6021 052716  012705  000021                  MOV     #17.,R5         ;R5 = CLOCK COUNT
 6022 052722                           60$:
      052722  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
 6023
 6024                                  ;;*****************************************************************
 6025 052730  013737  001142  001174          MOV     $BDDAT,$TMP0    ;IF CC AND CH ARE SET AT THE SAME TIME
 6026                                                                  ;THE ECO TO THE CS BOARD IS IMPLEMENTED.
 6027
 6028 052736  042737  171777  001174          BIC     #^C<CC!CH>,$TMP0        ;SAVE CC AND CH BITS
 6029 052744  022737  006000  001174          CMP     #CC!CH,$TMP0    ;ARE CC AND CH SET ?
 6030 052752  001414                          BEQ     65$             ;YES, BRANCH TO NEW LOCATION
 6031                                  ;;*****************************************************************
 6032
 6033 052754  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
 6034 052762  012437  001140                  MOV     (R4)+,$GDDAT
 6035 052766  053737  001446  001140          BIS     RMDC0,$GDDAT    ;OR CYLINDER ADDRESS
 6036 052774  023737  001140  001142          CMP     $GDDAT,$BDDAT
 6037 053002  001011                          BNE     70$             ;BRANCH IF TAG BUS WRONG
 6038 053004                           65$:
 6039 053004  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
 6040 053012  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
 6041 053020  005305                          DEC     R5
 6042 053022  001337                          BNE     60$
 6043 053024  000407                          BR      80$
 6044 053026  010037  001136          70$:    MOV     R0,$BDADR
 6045 053032  062737  000040  001136          ADD     #RMMR2,$BDADR
 6046 053040  104276                          EMT     276
 6047 053042  000557                          BR      160$
 6048
 6049                                  ;DROP ON CYLINDER TO RESET LATCH, SET ON CYLINDER TO PASS TEST AT
 6050                                  ;LOCATION 166
 6051 053044                           80$:
 6052 053044  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
 6053 053052  012760  041401  000024          MOV     #DMD!MUR.MOC.DBEN,RMMR1(R0)     ;LOAD RMMR1
 6054
```

```
                                        ;STEP SEQUENCER TO END OF OFFSET AT LOCATION 174 (37 CLOCKS)
053060  012705  000045                  MOV     #37.,R5         ;RELOAD CLOCK COUNT
                                        90$:
053064  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT

                                        ;;*****************************************************************
053072  013737  001142  001174          MOV     $BDDAT,$TMP0    ;IF CC AND CH ARE SET AT SAME TIME
                                                                ;THE ECO TO THE CS BOARD IS IMPLEMENTED.

053100  042737  171777  001174          BIC     #^C<CC!CH>,$TMP0        ;SAVE CC AND CH BITS
053106  022737  006000  001174          CMP     #CC!CH,$TMP0    ;ARE CC AND CH SET ?
053114  001003                          BNE     92$             ;NO !!
053116  162704  000002                  SUB     #2,R4           ;ADJUST THE TABLE ADDRESS
053122  000441                          BR      115$            ;TO OTHER LOCATION
053124                                  92$:
                                        ;;*****************************************************************

053124  042737  150000  001142          BIC     #RQA!RQB!TST,$BDDAT
053132  011437  001140                  MOV     (R4),$GDDAT
053136  032714  002000                  BIT     #CH,(R4)
053142  001425                          BEQ     110$            ;BRANCH IF CONTROL/HEADER NOT ON
053144  032714  004000                  BIT     #CC,(R4)
053150  001416                          BEQ     100$            ;BRANCH IF HEADER TAG

                                        ;CONTROL TAG SHOULD BE ON-SETUP EXPECTED OFFSET
053152  052737  000010  001140          BIS     #BB03,$GDDAT    ;ASSUME OFD IS NOT SET
053160  032737  000200  001444          BIT     #OFD,RMOFO
053166  001406                          BEQ     95$
053170  042737  000010  001140          BIC     #BB03,$GDDAT    ;RESET BUS BIT 3 - DIRECTION IS REV
053176  052737  000004  001140          BIS     #BB02,$GDDAT
053204  000404                  95$:    BR      110$

                                        ;HEADER TAG SHOULD BE ON - SETUP EXPECTED TRACK ADDRESS
053206                                  100$:
053206  113703  001421                  MOVB    RMDA0+1,R3      ;GET TRACK
053212  050337  001140                  BIS     R3,$GDDAT

                                        ;COMPARE EXPECTED AND RECEIVED TAG BUS DATA
053216                                  110$:
053216  023737  001140  001142          CMP     $GDDAT,$BDDAT
053224  001013                          BNE     120$
053226                                  115$:
053226  012760  141401  000024          MOV     #DMD!MUR!MOC!DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
053234  012760  041401  000024          MOV     #DMD!MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
053242  062704  000002                  ADD     #2,R4           ;MOVE TABLE POINTER
053246  005305                          DEC     R5              ;DECREMENT CLOCK COUNT
053250  001305                          BNE     90$
053252  000407                          BR      130$
053254  010037  001136          120$:   MOV     R0,$BDADR
053260  062737  000040  001136          ADD     #RMMR2,$BDADR
053266  104276                          EMT     276
053270  000444                          BR      160$

                                        ;DROP ON CYLINDER TO RESET LATCH, RAISE ON CYLINDER TO PASS TEST AT
                                        ;SEQUENCER LOCATION 175
053272                                  130$:
053272  012760  041001  000024          MOV     #DMD!MUR!DBEN,RMMR1(R0) ;LOAD RMMR1
```

```
6110 053300  012760  041401  000024          MOV     #DMD!MUR.MOC.DBEN,RMMR1(R0)     ;LOAD RMMR1
6111
6112                                 ;STEP SEQUENCER TO SECTOR WAIT LOOP (8 CLOCKS)
6113 053306  012705  000010                  MOV     #8.,R5
6114 053312                          140$:
     053312  016037  000040  001142          MOV     RMMR2(R0),$BDDAT        ;STORE RMMR2 AT $BDDAT
6115
6116                                 ;;*****************************************************************
6117                                 ;       BIC     #RQA.RQB.TST,$BDDAT
6118                                                                 ;THE FOLLOWING CODE WAS ADDED
6119                                                                 ;TO ALLOW THE PROGRAM TO RUN WITH
6120                                                                 ;OR WITHOUT ECO TO THE CS BOARD.
6121
6122 053320  042737  171777  001142          BIC     #^C<CC!CH>,$BDDAT       ;SAVE CC AND CH BITS
6123 053326  012737  006000  001140          MOV     #CC.CH,$GDDAT   ;GET EXPECTED DATA
6124                                 ;;*****************************************************************
6125
6126 053334  023737  001140  001142          CMP     $GDDAT,$BDDAT   ;GOOD DATA SAME AS LAST CMP
6127 053342  001011                          BNE     150$            ;BRANCH IF ERROR
6128 053344  012760  141401  000024          MOV     #DMD!MUR.MOC.DBEN!DBCK,RMMR1(R0)        ;LOAD RMMR1
6129 053352  012760  041401  000024          MOV     #DMD.MUR!MOC!DBEN,RMMR1(R0)     ;LOAD RMMR1
6130 053360  005305                          DEC     R5
6131 053362  001353                          BNE     140$
6132 053364  000406                          BR      160$
6133 053366  010037  001136          150$:   MOV     R0,$BDADR
6134 053372  062737  000040  001136          ADD     #RMMR2,$BDADR
6135 053400  104276                          EMT     276
6136
6137 053402  000207                  160$:   RTS     PC
6138
6139                                 ;TABLE OF TAG BUS CONTROL AND DATA VALUES
6140 053404                          200$:
6141 053404  001777                          .WORD   1777            ;LOCATION 0
6142 053406  001777                          .WORD   1777            ;LOCATION 25
6143 053410  001777                          .WORD   1777            ;LOCATION 26
6144 053412  001777                          .WORD   1777            ;LOCATION 128
6145 053414  001777                          .WORD   1777            ;LOCATION 129
6146 053416  001777                          .WORD   1777            ;LOCATION 130
6147 053420  004000                          .WORD   CC              ;LOCATION 144
6148 053422  004000                          .WORD   CC              ;LOCATION 145
6149 053424  024000                          .WORD   CC!TAG          ;LOCATION 146
6150 053426  024000                          .WORD   CC!TAG          ;LOCATION 147
6151 053430  024000                          .WORD   CC!TAG          ;LOCATION 148
6152 053432  024000                          .WORD   CC!TAG          ;LOCATION 149
6153 053434  024000                          .WORD   CC!TAG          ;LOCATION 150
6154 053436  024000                          .WORD   CC!TAG          ;LOCATION 151
6155 053440  004000                          .WORD   CC              ;LOCATION 152
6156 053442  004000                          .WORD   CC              ;LOCATION 153
6157 053444  001777                          .WORD   1777            ;LOCATION 154
6158 053446  002000                          .WORD   CH              ;LOCATION 156
6159 053450  002000                          .WORD   CH              ;LOCATION 157
6160 053452  022000                          .WORD   CH.TAG          ;LOCATION 158
6161 053454  022000                          .WORD   CH!TAG          ;LOCATION 159
6162 053456  022000                          .WORD   CH!TAG          ;LOCATION 160
6163 053460  022000                          .WORD   CH!TAG          ;LOCATION 161
6164 053462  022000                          .WORD   CH.TAG          ;LOCATION 162
6165 053464  022000                          .WORD   CH!TAG          ;LOCATION 163
```

```
 6166 053466  002000                    .WORD   CH              ;LOCATION 164
 6167 053470  002000                    .WORD   CH              ;LOCATION 165
 6168 053472  001777                    .WORD   1777            ;LOCATION 232
 6169 053474  001777                    .WORD   1777            ;LOCATION 233
 6170 053476  001777                    .WORD   1777            ;LOCATION 234
 6171 053500  001777                    .WORD   1777            ;LOCATION 235
 6172 053502  001777                    .WORD   1777            ;LOCATION 236
 6173 053504  001777                    .WORD   1777            ;LOCATION 237
 6174 053506  001777                    .WORD   1777            ;LOCATION 238
 6175 053510  001777                    .WORD   1777            ;LOCATION 239
 6176 053512  001777                    .WORD   1777            ;LOCATION 240
 6177 053514  001777                    .WORD   1777            ;LOCATION 241
 6178 053516  001777                    .WORD   1777            ;LOCATION 242
 6179 053520  001777                    .WORD   1777            ;LOCATION 243
 6180 053522  001777                    .WORD   1777            ;LOCATION 244
 6181 053524  001777                    .WORD   1777            ;LOCATION 245
 6182 053526  001777                    .WORD   1777            ;LOCATION 246
 6183 053530  001777                    .WORD   1777            ;LOCATION 247
 6184 053532  001777                    .WORD   1777            ;LOCATION 248
 6185 053534  001777                    .WORD   1777            ;LOCATION 249
 6186 053536  001777                      ORD   1777            ;LOCATION 250
 6187 053540  001777                    .WORD   1777            ;LOCATION 251
 6188 053542  001777                    .WORD   1777            ;LOCATION 252
 6189 053544  001777                    .WORD   1777            ;LOCATION 166
 6190 053546  006000                    .WORD   CC!CH           ;LOCATION 169
 6191 053550  006000                    .WORD   CC!CH           ;LOCATION 170
 6192 053552  026000                    .WORD   CC!CH!TAG       ;LOCATION 171
 6193 053554  026000                    .WORD   CC!CH!TAG       ;LOCATION 172
 6194 053556  026000                    .WORD   CC!CH!TAG       ;LOCATION 173
 6195 053560  026000                    .WORD   CC!CH!TAG       ;LOCATION 174
 6196 053562  026000                    .WORD   CC!CH!TAG       ;LOCATION 175,ETC
 6197
 6198 053564                    300$:                           ;END OF TEST
```

```
                                .SBTTL  END OF SUB-PASS ROUTINE
                                ;
                                ;THIS IS THE END OF SUB-PASS ROUTINE. THIS ROUTINE IS USED TO
                                ;TERMINATE THE OPERATION OF THE CURRENT DEVICE UNDER TEST AND
                                ;SELECT THE NEXT DEVICE FOR TEST. IF THERE ARE NO MORE DEVICES
                                ;TO TEST, EXIT IS MADE TO '$EOP' ROUTINE. OTHERWISE, RETURN
                                ;IS MADE TO 'READY' ROUTINE.
                                ;
053564  000004                  $EOSP:  SCOPE
053566  000240                          NOP
053570  013700  001466                  MOV     TSTQUE,R0       ;GET POINTER TO TSTQUE
053574  062700  000002                  ADD     #2,R0           ;ADJUST POINTER TO NEXT DEVICE
053600  010037  001466                  MOV     R0,TSTQUE       ;SAVE POINTER TO TSTQUE
053604  005710                          TST     (R0)            ;ANY MORE DEVICES FOR TEST ?
053606  001402                          BEQ     1$              ;BR IF NO
053610  000137  007120                  JMP     READY           ;YES, JUMP TO 'READY' ROUTINE
053614  012737  001470  001466  1$:     MOV     #TSTQUE+2,TSTQUE        ;INITIALIZE POINTER TO FIRST DEVICE IN
                                                                ;TEST QUE TABLE
                                .SBTTL  END OF PASS ROUTINE
                                ;;**********************************************************************
                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY''
                                ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO READY
053622                          $EOP:
053622  000240                          NOP
053624  005037  001116                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
053630  005037  001206                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
053634  005237  001230                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
053640  042737  100000  001230          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
053646  005327                          DEC     (PC)+           ;;LOOP?
053650  000001                  $EOPCT: .WORD   1
053652  003066                          BGT     $DOAGN          ;;YES
053654  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
053656  000001                  $ENDCT: .WORD   1
053660  053650                          $EOPCT
053662  104401  053670                  TYPE    ,65$            ;;TYPE ASCIZ STRING
053666  000407                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <12><15>/END PASS #/
053706                          64$:
053706  013746  001230                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                ;;TYPE PASS NUMBER
053712  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
053714  005737  001126                  TST     $ERTTL          ;;SEE IF ANY ERRORS THIS PASS
053720  001431                          BEQ     $GT42P          ;;BR IF NO ERRORS TO REPORT
053722  104401  053730                  TYPE    ,67$            ;;TYPE ASCIZ STRING
053726  000421                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
053772                          66$:
053772  013746  001126                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
                                                                ;;TOTAL NUMBER OF ERRORS
053776  104405                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
054000  005037  001126                  CLR     $ERTTL          ;;CLEAR ERROR TOTAL
```

END OF PASS ROUTINE

```
054004  104401  001217          $GT42P. TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
054010  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
054014  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
054016  000005                          RESET                   ;;CLEAR THE WORLD
054020  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
054022  000240                          NOP                     ;;SAVE ROOM
054024  000240                          NOP                     ;;FOR
054026  000240                          NOP                     ;;ACT11
054030                          $DOAGN:
054030  000137                          JMP     @(SP)+          ;;RETURN
054032  007120                  $RTNAD: .WORD   READY
054034     377     377     000  $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
                                        .EVEN
```

```
                                        .SBTTL  CLOCK SUBROUTINES
                                        ;
                                        ;ROUTINE TO SIZE FOR CLOCKS (KW11-L OR KW11-P)
                                        ;
054040  000240                  SIZCLK: NOP
054042  013746  000004                  MOV     ERRVEC,-(SP)    ;;PUSH ERRVEC ON STACK
054046  013746  000006                  MOV     ERRVEC+2,-(SP)          ;;PUSH ERRVEC+2 ON STACK
054052  012737  054136  000004          MOV     #10$,ERRVEC     ;LOAD 04 TRAP VECTORS
054060  012737  000300  000006          MOV     #PR6,ERRVEC+2
                                        ;
                                        ;SEE IF A KW11-P CLOCK IS PRESENT - GO TO 10$ IF NOT PRESENT
054066  005777  125420                  TST     @$LPCSR         ;TEST FOR P CLOCK
054072  012737  054300  001536          MOV     #PCLOCK,CLOCK   ;LOAD SUBROUTINE ADDRESS
054100  012737  054422  001540          MOV     #PSTOP,STOPCL   ;LOAD STOP ADDRESS
054106  012777  054366  125402          MOV     #PCOUNT,@$LPVEC ;LOAD P CLOCK INTERRUPT VECTOR
054114  012777  000300  125376          MOV     #PR6,@$LPVEC+2
054122  013777  001526  125374          MOV     $LLVEC+2,@$LLVEC        ;CLEAR L CLOCK INTERRUPT VECTOR
054130  005077  125372                  CLR     @$LLVEC+2
054134  000454                          BR      30$
054136  012716  054144          10$:    MOV     #15$,(SP)       ;DUMMY RTI ADDRESS
054142  000002                          RTI                     ;RESTORE PRIORITY
                                        ;
                                        ;NO P CLOCK-SEE IF L CLOCK IS PRESENT-GO TO 20$ IF NOT PRESET
054144                          15$:
054144  012737  054222  000004          MOV     #20$,ERRVEC     ;CHANGE 04 TRAP VECTOR
054152  005777  125344                  TST     @$LLCSR         ;TEST FOR L CLOCK
054156  012737  054316  001536          MOV     #LCLOCK,CLOCK   ;LOAD SUBROUTINE ADDRESS
054164  012737  054430  001540          MOV     #LSTOP,STOPCL   ;LOAD STOP ADDRESS
054172  012777  054366  125324          MOV     #LCOUNT,@$LLVEC ;LOAD L CLOCK INTERRUPT VECTOR
054200  012777  000300  125320          MOV     #PR6,@$LLVEC+2
054206  013777  001520  125302          MOV     $LPVEC+2,@$LPVEC;CLEAR P CLOCK INTERRUPT VECTOR
054214  005077  125300                  CLR     @$LPVEC+2
054220  000422                          BR      30$
054222  012716  054230          20$:    MOV     #25$,(SP)       ;DUMMY RTI ADDRESS
054226  000002                          RTI                     ;RESTORE PRIORITY
                                        ;
                                        ;NO CLOCK AVAILABLE - AUGMENT RETURN ADDRESS
054230                          25$:
054230  005037  001536                  CLR     CLOCK           ;CLEAR SUBROUTINE ADDRESS
054234  012737  001520  001516          MOV     #$LPVEC+2,$LPVEC;CLEAR P CLOCK INTERRUPT VECTOR
054242  005037  001520                  CLR     $LPVEC+2
054246  012737  001526  001524          MOV     #$LLVEC+2,$LLVEC;CLEAR L CLOCK INTERRUPT VECTOR
054254  005037  001526                  CLR     $LLVEC+2
054260  062766  000002  000004          ADD     #2,4(SP)        ;CHANGE RETURN ADDRESS
054266                          30$:
054266  012637  000006                  MOV     (SP)+,ERRVEC+2          ;;POP STACK INTO ERRVEC+2
054272  012637  000004                  MOV     (SP)+,ERRVEC    ;;POP STACK INTO ERRVEC
054276  000207                          RTS     PC
                                        ;
                                        ;ROUTINES TO START THE CLOCK (KW11-L OR KW11-P)
                                        ;
054300  012777  177777  125206  PCLOCK: MOV     #-1,@$LPCSB     ;LOAD COUNT SET BUFFER
054306  012777  000135  125176          MOV     #135,@$LPCSR    ;LOAD CONTROL REGISTER
054314  000403                          BR      PLCLK           ;GO TO COMMON CODE
                                        ;
054316  012777  000100  125176  LCLOCK: MOV     #100,@$LLCSR    ;LOAD CONTROL REGISTER
                                        ;
```

```
     057 054324  005037  001532          PLCLK:  CLR     TIME            ;CLEAR TIMER COUNT
     058 054330  104400                          TRAP                    ;;PUSH OLD PSW AND PC ON STACK
         054332  012605                          MOV     (SP)+,R5        ;;SAVE THE PSW IN R5
     059 054334  010537  001530                  MOV     R5,$PSW         ;SAVE PRIORITY
     060 054340  042705  177437                  BIC     #^CPR7,R5       ;MASK X
     061 054344  022705  000300                  CMP     #PR6,R5         ;IS PRIORITY TOO HIGH??
     062 054350  101005                          BHI     40$             ;NO!!
     063 054352  012746  000240                  MOV     #PR5,-(SP)      ;;PUT NEW PS ON STACK
         054356  012746  054364                  MOV     #30$,-(SP)      ;;PUT NEW PC ON  STACK
         054362  000002                          RTI                     ;;POP NEW PC AND PS
         054364                          30$:
     064 054364  000207                  40$:    RTS     PC
     065
     066                                 ;ROUTINES TO HANDLE CLOCK INTERRUPTS (KW11-L OR KW11-P)
     067
     068 054366                          PCOUNT:
     069 054366                          LCOUNT:
     070 054366  062737  000021  001532          ADD     #17.,TIME       ;ADD 17MS TO ELAPSED TIME
     071 054374  103003                          BCC     10$             ;BRANCH IF NO OVERFLOW
     072 054376  012737  177777  001532          MOV     #-1,TIME        ;RESTORE MAXIMUM COUNT
     073 054404  162737  000021  001534  10$:    SUB     #17.,WATCH      ;DECREMENT REMAINING TIME
     074 054412  100002                          BPL     20$             ;BRANCH IF POSITIVE
     075 054414  005037  001534                  CLR     WATCH           ;CLEAR REMAINING TIME
     076 054420  000002                  20$:    RTI                     ;RETURN TO USER
     077
     078                                 ;ROUTINES TO STOP THE CLOCK (KW11-L OR KW11-P)
     079
     080 054422  005077  125064          PSTOP:  CLR     @$LPCSR         ;STOP P CLOCK
     081 054426  000402                          BR      PLSTP           ;GO TO COMMON STOP CODE
     082
     083 054430  005077  125066          LSTOP:  CLR     @$LLCSR         ;STOP L CLOCK
     084
     085 054434                          PLSTP:
     086 054434  013746  001530                  MOV     $PSW,-(SP)      ;;PUT NEW PS ON STACK
         054440  012746  054446                  MOV     #10$,-(SP)      ;;PUT NEW PC ON  STACK
         054444  000002                          RTI                     ;;POP NEW PC AND PS
         054446                          10$:
     087 054446  000207                          RTS     PC
```

```
                                .SBTTL  SET VOLUME VALID SUBROUTINE
                                ;
                                ;THIS SUBROUTINE INITIALIZES THE SUBSYSTEM AND SETS VOLUME VALID,
                                ;RETURNING WITH THE DRIVE STILL IN DIAGNOSTIC MODE.  THE SUBROUTINE
                                ;RETURNS TO THE WORD FOLLOWING THE CALL, EXCEPT WHEN AN ERROR IS
                                ;DETECTED, IN WHICH CASE IT RETURNS TO THE SECOND WORD FOLLOWING THE
                                ;CALL.
                                ;
                                ;CALL:  JSR     PC,SETVV        JUMP TO SUBROUTINE
                                ;       BR      ??              RETURN HERE IF NO ERROR
                                ;       ERROR                   RETURN HERE IF ERROR
                                ;
 054450                         SETVV:
 054450  004737  054674         10$:    JSR     PC,CNTCLR       ;GO CLEAR CONTROLLER
 054454  012760  000001  000024         MOV     #DMD,RMMR1(R0)  ;LOAD RMMR1
 054462  012760  001001  000024         MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
 054470  012760  000000  000014         MOV     #0,RMER1(R0)    ;LOAD RMER1
 054476  012760  000000  000042         MOV     #0,RMER2(R0)    ;LOAD RMER2
 054504  012760  000023  000000         MOV     #PACACK!GO,RMCS1(R0)    ;LOAD RMCS1
 054512  016037  000012  001142         MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
 054520  042737  177677  001142         BIC     #^CVV,$BDDAT
 054526  001020                         BNE     10$             ;BRANCH IF VOLUME VALID SET
 054530  010037  001136                 MOV     R0,$BDADR       ;SETUP FOR ERROR MSG
 054534  062737  000012  001136         ADD     #RMDS,$BDADR
 054542  012737  000100  001140         MOV     #VV,$GDDAT
 054550  062716  000002                 ADD     #2,(SP)         ;MOVE RETURN ADDRESS TO ERROR
 054554  112776  000170  000000         MOVB    #170,@(SP)      ;WRITE ERROR NUMBER
 054562  012737  000022  001174         MOV     #PACACK,$TMP0
 054570  000207                 10$:    RTS     PC              ;RETURN
```

SEQ 0228

```
                                        .SBTTL  SET OFFSET MODE SUBROUTINE
                                        ;
                                        ;THIS SUBROUTINE EXECUTES AN OFFSET COMMAND AND VERIFIES THAT OFFSET
                                        ;MODE SETS.  THE DRIVE SHOULD BE IN DIAGNOSTIC MODE WHEN CALLING THE
                                        ;SUBROUTINE, WHICH WILL LEAVE DMD ON.  THE SUBROUTINE RETURNS TO THE
                                        ;WORD FOLLOWING THE CALL UNLESS THERE IS AN ERROR, IN WHICH CASE IT
                                        ;RETURNS TO THE SECOND WORD FOLLOWING THE CALL
                                        ;
                                        ;CALL:  JSR     PC,SETOM        JUMP TO SUBROUTINE
                                        ;       BR      ??              RETURN HERE IF NO ERROR
                                        ;       ERROR                   RETURN HERE IF ERROR
                                        ;
054572                                  SETOM:
054572  012760  001001  000024                MOV     #DMD!MUR,RMMR1(R0)      ;LOAD RMMR1
054600  012760  000000  000014                MOV     #0,RMER1(R0)    ;LOAD RMER1
054606  012760  000000  000042                MOV     #0,RMER2(R0)    ;LOAD RMER2
054614  012760  000015  000000                MOV     #OFFSET!GO,RMCS1(R0)    ;LOAD RMCS1
054622  016037  000012  001142                MOV     RMDS(R0),$BDDAT ;STORE RMDS AT $BDDAT
054630  042737  177776  001142                BIC     #^COM,$BDDAT
054636  001015                                BNE     10$             ;BRANCH IF OFFSET ON
054640  012737  000001  001140                MOV     #OM,$GDDAT
054646  010037  001136                        MOV     R0,$BDADR
054652  062737  000012  001136                ADD     #RMDS,$BDADR
054660  062716  000002                        ADD     #2,(SP)         ;MOVE RETURN ADDRESS TO ERROR
054664  112776  000200  000000                MOVB    #200,@(SP)      ;WRITE ERROR NUMBER
054672                                  10$:
054672  000207                                RTS     PC              ;RETURN TO USER
```

SEQ 0224

```
                                .SBTTL  CLEAR CONTROLLER SUBROUTINE

                                ;THIS SUBROUTINE CLEARS THE MASSBUS CONTROLLER, MASSBUS ADAPTERS,
                                ;AND DRIVES, THEN SELECTS THE DRIVE.
                                ;CALL:
                                ;       JSR     PC,CNTCLR       ;CALL TO ROUTINE

054674                          CNTCLR:
054674  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
054676  013700  001276                  MOV     $BASE,R0        ;R0 = UNIBUS BASE ADDRESS
054702  012760  000040  000010          MOV     #CLR,RMCS2(R0)  ;CLEAR MASSBUS
054710  117760  124552  000010          MOVB    @TSTQUE,RMCS2(R0)       ;SELECT DEVICE UNDER TEST
054716  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
054720  000207                          RTS     PC              ;RETURN
```

```
      1                               .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

 054722                               $SAVREG:
 054722  010046                               MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 054724  010146                               MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 054726  010246                               MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 054730  010346                               MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 054732  010446                               MOV     R4,-(SP)        ;;PUSH R4 ON STACK
 054734  010546                               MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 054736  016646  000022                       MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
 054742  016646  000022                       MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
 054746  016646  000022                       MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
 054752  016646  000022                       MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
 054756  000002                             RTI

                                      ;*RESTORE R0-R5
                                      ;*CALL:
                                      ;*      RESREG
 054760                               $RESREG:
 054760  012666  000022                       MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
 054764  012666  000022                       MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
 054770  012666  000022                       MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
 054774  012666  000022                       MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
 055000  012605                             MOV     (SP)+,R5        ;;POP STACK INTO R5
 055002  012604                             MOV     (SP)+,R4        ;;POP STACK INTO R4
 055004  012603                             MOV     (SP)+,R3        ;;POP STACK INTO R3
 055006  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
 055010  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
 055012  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
 055014  000002                             RTI
```

```
                                        .SBTTL  BINARY TO ASCII AND TYPE ROUTINE

                                        ;;**********************************************************************
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 16-BIT
                                        ;*BINARY-ASCII NUMBER AND TYPE IT.
                                        ;*CALL:
                                        ;*      MOV     NUMBER,-(SP)    ;;NUMBER TO BE TYPED
                                        ;*      TYPBN                   ;;TYPE IT

055016  010146                         $TYPBN: MOV     R1,-(SP)        ;;SAVE R1 ON THE STACK
055020  016601  000006                          MOV     6(SP),R1        ;;GET THE INPUT NUMBER
055024  000261                                 SEC                     ;;SET "C" SO CAN KEEP TRACK OF THE NUMBER OF BITS
055026  112737  000060  055070          1$:     MOVB    #'0,$BIN        ;;SET CHARACTER TO AN ASCII "0".
055034  006101                                 ROL     R1              ;;GET THIS BIT
055036  001406                                 BEQ     2$              ;;DONE?
055040  105537  055070                         ADCB    $BIN            ;;NO--SET THE CHARACTER EQUAL TO THIS BIT
055044  104401  055070                         TYPE    ,$BIN           ;;GO TYPE THIS BIT
055050  000241                                 CLC                     ;;CLEAR "C" SO CAN KEEP TRACK OF BITS
055052  000765                                 BR      1$              ;;GO DO THE NEXT BIT
055054  012601                          2$:     MOV     (SP)+,R1        ;;POP THE STACK INTO R1
055056  016666  000002  000004                 MOV     2(SP),4(SP)     ;;ADJUST THE STACK
055064  012616                                 MOV     (SP)+,(SP)
055066  000002                                 RTI                     ;;RETURN TO USER
055070     000     000                 $BIN:   .BYTE   0,0             ;;STORAGE FOR ASCII CHAR. AND TERMINATOR
```

SEQ 023

```
                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

055072                          $TYPDS:
055072  010046                         MOV     R0,-(SP)        ;;PUSH R0 ON STACK
055074  010146                         MOV     R1,-(SP)        ;;PUSH R1 ON STACK
055076  010246                         MOV     R2,-(SP)        ;;PUSH R2 ON STACK
055100  010346                         MOV     R3,-(SP)        ;;PUSH R3 ON STACK
055102  010546                         MOV     R5,-(SP)        ;;PUSH R5 ON STACK
055104  012746  020200                 MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
055110  016605  000020                 MOV     20(SP),R5       ;;GET THE INPUT NUMBER
055114  100004                         BPL     1$              ;;BR IF INPUT IS POS.
055116  005405                         NEG     R5              ;;MAKE THE BINARY NUMBER POS.
055120  112766  000055  000001         MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
055126  005000                 1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
055130  012703  055306                 MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
055134  112723  000040                 MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
055140  005002                 2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
055142  016001  055276                 MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
055146  160105                 3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
055150  002402                         BLT     4$              ;;BR IF DONE
055152  005202                         INC     R2              ;;INCREASE THE BCD DIGIT BY 1
055154  000774                         BR      3$
055156  060105                 4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
055160  005702                         TST     R2              ;;CHECK IF BCD DIGIT=0
055162  001002                         BNE     5$              ;;FALL THROUGH IF 0
055164  105716                         TSTB    (SP)            ;;STILL DOING LEADING 0'S?
055166  100407                         BMI     7$              ;;BR IF YES
055170  106316                 5$:     ASLB    (SP)            ;;MSD?
055172  103003                         BCC     6$              ;;BR IF NO
055174  116663  000001  177777         MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
055202  052702  000060         6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
055206  052702  000040         7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
055212  110223                         MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
055214  005720                         TST     (R0)+           ;;JUST INCREMENTING
055216  020027  000010                 CMP     R0,#10          ;;CHECK THE TABLE INDEX
055222  002746                         BLT     2$              ;;GO DO THE NEXT DIGIT
055224  003002                         BGT     8$              ;;GO TO EXIT
055226  010502                         MOV     R5,R2           ;;GET THE LSD
055230  000764                         BR      6$              ;;GO CHANGE TO ASCII
055232  105726                 8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
055234  100003                         BPL     9$              ;;BR IF NO
055236  116663  177777  177776         MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
055244  105013                 9$:     CLRB    (R3)            ;;SET THE TERMINATOR
055246  012605                         MOV     (SP)+,R5        ;;POP STACK INTO R5
055250  012603                         MOV     (SP)+,R3        ;;POP STACK INTO R3
055252  012602                         MOV     (SP)+,R2        ;;POP STACK INTO R2
055254  012601                         MOV     (SP)+,R1        ;;POP STACK INTO R1
```

SEQ 0233

```
055256  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
055260  104401  055306                  TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
055264  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
055272  012616                          MOV     (SP)+,(SP)
055274  000002                          RTI                     ;;RETURN TO USER
055276  023420                  $DTBL.  10000.
055300  001750                          1000.
055302  000144                          100.
055304  000012                          10.
055306                          $DBLK:  .BLKW   4
```

```
                                .SBTTL  BINARY TO OCTA' (ASCII) AND TYPE

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

055316  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
055322  116637  000001  055541          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
055330  112637  055543                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
055334  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
055340  000406                          BR      $TYPON
055342  112737  000001  055541  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
055350  112737  000006  055543          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
055356  112737  000005  055540  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
055364  010346                          MOV     R3,-(SP)        ;;SAVE R3
055366  010446                          MOV     R4,-(SP)        ;;SAVE R4
055370  010546                          MOV     R5,-(SP)        ;;SAVE R5
055372  113704  055543                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
055376  005404                          NEG     R4
055400  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
055404  110437  055542                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
055410  113704  055541                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
055414  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
055420  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
055422  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
055424  000404                          BR      3$              ;;GO DO MSB
055426  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
055430  006105                          ROL     R5
055432  006105                          ROL     R5
055434  010503                          MOV     R5,R3
055436  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
055440  105337  055542                  DECB    $OMODE          ;;TYPE THIS DIGIT?
055444  100016                          BPL     7$              ;;BR IF NO
055446  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
055452  001002                          BNE     4$              ;;TEST FOR 0
055454  005704                          TST     R4              ;;SUPPRESS THIS 0?
055456  001403                          BEQ     5$              ;;BR IF YES
055460  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
```

```
055462  052703  000060            BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
055466  052703  000040     5$:    BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
055472  110337  055536            MOVB    R3,8$           ;;SAVE FOR TYPING
055476  104401  055536            TYPE    ,8$             ;;GO TYPE THIS DIGIT
055502  105337  055540     7$:    DECB    $OCNT           ;;COUNT BY 1
055506  003347                    BGT     2$              ;;BR IF MORE TO DO
055510  002402                    BLT     6$              ;;BR IF DONE
055512  005204                    INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
055514  000744                    BR      2$              ;;GO DO THE LAST DIGIT
055516  012605     6$:    MOV     (SP)+,R5        ;;RESTORE R5
055520  012604                    MOV     (SP)+,R4        ;;RESTORE R4
055522  012603                    MOV     (SP)+,R3        ;;RESTORE R3
055524  016666  000002  000004    MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
055532  012616                    MOV     (SP)+,(SP)
055534  000002                    RTI                     ;;RETURN
055536     000     8$:    .BYTE   0               ;;STORAGE FOR ASCII DIGIT
055537     000                    .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
055540     000     $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
055541     000     $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
055542  000000     $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                .SBTTL  TYPE ROUTINE

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

055544  105737  001173          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
055550  100002                          BPL     1$              ;;BR IF YES
055552  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
055554  000430                          BR      3$              ;;LEAVE
055556  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
055560  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
055564  122737  000001  001242          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
055572  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
055574  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
055602  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
055604  010037  055614                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
055610  004737  062346                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
055614  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
055616  132737  000040  001243  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
055624  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
055626  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
055630  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
055632  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
055634  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
055636  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
055642  000002                          RTI                     ;;RETURN
055644  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
055650  001430                          BEQ     8$
055652  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
055656  001006                          BNE     5$
055660  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
055662  104401                          TYPE                    ;;TYPE A CR AND LF
055664  001217                          $CRLF
055666  105037  056074                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
055672  000755                          BR      2$              ;;GET NEXT CHARACTER
055674  004737  055756          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
055700  123726  001172          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
055704  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
055706  013746  001170                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
055712  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
055716  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
055720  004737  055756                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
055724  105337  056074                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
055730  000770                          BR      7$              ;;LOOP
```

```
                                ;HORIZONTAL TAB PROCESSOR

055732  112716  000040          8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
055736  004737  055756          9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
055742  132737  000007  056074          BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
055750  001372                          BNE     9$              ;;TAB STOP
055752  005726                          TST     (SP)+           ;;POP SPACE OFF STACK
055754  000724                          BR      2$              ;;GET NEXT CHARACTER
055756                          $TYPEC:
055756  105777  123176                  TSTB    @$TKS           ;;CHAR IN KYBD BUFFER?
055762  100022                          BPL     10$             ;;BR IF NOT
055764  017746  123172                  MOV     @$TKB,-(SP)     ;;GET CHAR
055770  042716  177600                  BIC     #177600,(SP)    ;;STRIP EXTRANEOUS BITS
055774  122716  000023                  CMPB    #$XOFF,(SP)     ;;WAS CHAR XOFF
056000  001012                          BNE     102$            ;;BR IF NOT
056002                          101$:
056002  105777  123152                  TSTB    @$TKS           ;;WAIT FOR CHAR
056006  100375                          BPL     101$
056010  117716  123146                  MOVB    @$TKB,(SP)      ;;GET CHAR
056014  042716  177600                  BIC     #177600,(SP)    ;;STRIP IT
056020  122716  000021                  CMPB    #$XON,(SP)      ;;WAS IT XON?
056024  001366                          BNE     101$            ;;BR IF NOT
056026                          102$:
056026  005726                          TST     (SP)+           ;;FIX STACK
056030                          10$:
056030  105777  123130                  TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
056034  100375                          BPL     10$
056036  116677  000002  123122          MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
056044  122766  000015  000002          CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
056052  001003                          BNE     1$              ;;BRANCH IF NO
056054  105037  056074                  CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
056060  000406                          BR      $TYPEX          ;;EXIT
056062  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
056070  001402                          BEQ     $TYPEX          ;;BRANCH IF YES
056072  105227                          INCB    (PC)+           ;;COUNT THE CHARACTER
056074  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
056076  000207          $TYPEX: RTS     PC
```

```
                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09=1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN SWR<7:0>
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

056100                          $SCOPE:
056100  104410                          CKSWR                   ;;TEST  FOR CHANGE IN SOFT-SWR
056102  032777  040000  123044  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
056110  001402                          BEQ     9$              ;;NO IF SW14=0
056112  000137  056542                  JMP     $OVER           ;;JUMP OVER SCOPE ROUTINE
056116                          9$:
                                ;#####START OF CODE FOR THE XOR TESTER#####
056116  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
056120  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
056124  012737  056144  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
056132  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
056136  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
056142  000561                          BR      $SVLAD          ;;GO TO THE NEXT TEST
056144  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
056146  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
056152  000521                          BR      7$              ;;LOOP ON THE PRESENT TEST
056154                          6$:;#####END OF CODE FOR THE XOR TESTER#####
056154  032777  000400  122772          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
056162  001421                          BEQ     2$              ;;BR IF NO
056164  005046                          CLR     -(SP)           ;;CLEAR A TEMP. LOCATION
056166  117716  122762                  MOVB    @SWR,(SP)       ;;PICKUP THE DESIRED TEST NUMBER
056172  001414                          BEQ     8$              ;;BRANCH IF BAD TEST NUMBER IN SWR
056174  022716  000120                  CMP     #120,(SP)       ;;CHECK THE NUMBER IN THE SWR
056200  002411                          BLT     8$              ;;BRANCH IF TEST NUMBER IS OUT OF RANGE
056202  011637  001116                  MOV     (SP),$TSTNM     ;;UPDATE THE TEST NUMBER
056206  005316                          DEC     (SP)            ;;BACKUP BY ONE
056210  006316                          ASL     (SP)            ;;SCALE THE TEST NUMBER AS AN INDEX
056212  062716  056560                  ADD     #$SW08TBL,(SP)  ;;FORM THE ADDRESS OF TEST POINTER
056216  013637  001122                  MOV     @(SP)+,$LPADR   ;;SET LOOP ADDRESS TO DESIRED TEST
056222  000547                          BR      $OVER           ;;GO LOOP ON THE TEST
056224  005726                  8$:     TST     (SP)+           ;;CLEAN THE BAD TEST NUMBER OFF OF THE STACK
056226  105737  001117          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
056232  001502                          BEQ     3$              ;;BR IF NO
056234  022737  177777  057410          CMP     #-1,CPSAVE      ;;SEE IF TIMEOUT WAS PREVIOUSLY RECORDED
056242  001455                          BEQ     2003$           ;;KICK AROUND ROUTINE IF SO
056244  013746  000004                  MOV     ERRVEC,-(SP)    ;;SAVE CONTENTS OF ERROR VECTOR
056250  012737  056266  000004          MOV     #2000$,ERRVEC   ;;SETUP 'TRAP' RETURN ADDRESS
056256  013737  177766  057410          MOV     177766,CPSAVE   ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
056264  000406                          BR      2001$
056266  012737  177777  057410  2000$:  MOV     #-1,CPSAVE      ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
056274  012716  056302                  MOV     #2001$,(SP)     ;;SETUP RETURN ADDRESS
056300  000002                          RTI
```

```
056302  012637  000004          2001$:  MOV     (SP)+,ERRVEC    ;;RESTORE CONTENTS OF ERROR VECTOR

056306  022737  177777  057410  2002$:  CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
056314  001430                          BEQ     2003$           ;;BRANCH IF SO
056316  032737  000001  057410          BIT     #BIT00,CPSAVE   ;;SEE IF THE POWER MONITOR BIT IS ON
056324  001424                          BEQ     2003$           ;;BRANCH TO CONTINUE ROUTINE IF CLEAR
056326  042737  000001  177766          BIC     #BIT00,177766   ;;CLEAR THE BIT FOUND TO BE SET
056334  013746  001154                  MOV     SWR,-(SP)       ;;SAVE SWR ADDRESS
056340  017646  000000                  MOV     @(SP),-(SP)     ;;SAVE SWR VALUE
056344  012737  000176  001154          MOV     #176,SWR        ;;GET SOFTWARE SWR ADDRESS
056352  011677  122576                  MOV     (SP),@SWR       ;;GET CURRENT SWR VALUE
056356  042777  001000  122570          BIC     #BIT09,@SWR     ;;DON'T ALLOW LOOP ON ERROR ON THIS ERROR
056364  104177                          EMT     177             ;;CALL SPECIAL POWER FAIL BIT ERROR CALL
056366  012676  000000                  MOV     (SP)+,@(SP)     ;;RESTORE SWR TO ORIGINAL VALUE
056372  012637  001154                  MOV     (SP)+,SWR       ;;RESTORE SWR ADDRESS
056376                          2003$:
056376  123737  001131  001117          CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
056404  101015                          BHI     3$              ;;BR IF NO
056406  032777  001000  122540          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
056414  001404                          BEQ     4$              ;;BR IF NO
056416  013737  001124  001122  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
056424  000446                          BR      $OVER
056426  105037  001117          4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
056432  005037  001206                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
056436  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
056440  032777  004000  122506  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
056446  001011                          BNE     1$              ;;BR IF YES
056450  005737  001230                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
056454  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
056456  005237  001120                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
056462  023737  001206  001120          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
056470  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
056472  012737  000001  001120  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
056500  013737  056556  001206          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
056506  105237  001116          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
056512  113737  001116  001226          MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
056520  011637  001122                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
056524  011637  001124                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
056530  005037  001210                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
056534  112737  000001  001131          MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
056542  013777  001116  122406  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
056550  013716  001122                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
056554  000002                          RTI                     ;;FIXES PS
056556  000005                  $MXCNT: 5.                      ;;MAX. NUMBER OF ITERATIONS
056560                          $SW08TBL:
        000120                  .REPT   $TN-1
056560  007310                          .WORD   TST1+2          ;;STARTING ADDRESS OF TEST 1
056562  007620                          .WORD   TST2+2          ;;STARTING ADDRESS OF TEST 2
056564  010002                          .WORD   TST3+2          ;;STARTING ADDRESS OF TEST 3
056566  010152                          .WORD   TST4+2          ;;STARTING ADDRESS OF TEST 4
056570  010302                          .WORD   TST5+2          ;;STARTING ADDRESS OF TEST 5
056572  011366                          .WORD   TST6+2          ;;STARTING ADDRESS OF TEST 6
056574  012436                          .WORD   TST7+2          ;;STARTING ADDRESS OF TEST 7
056576  012560                          .WORD   TST10+2         ;;STARTING ADDRESS OF TEST 10
056600  012644                          .WORD   TST11+2         ;;STARTING ADDRESS OF TEST 11
056602  013104                          .WORD   TST12+2         ;;STARTING ADDRESS OF TEST 12
056604  013452                          .WORD   TST13+2         ;;STARTING ADDRESS OF TEST 13
```

```
056606  014234                  .WORD   TST14+2         ;;STARTING ADDRESS OF TEST 14
056610  014334                  .WORD   TST15+2         ;;STARTING ADDRESS OF TEST 15
056612  014660                  .WORD   TST16+2         ;;STARTING ADDRESS OF TEST 16
056614  015372                  .WORD   TST17+2         ;;STARTING ADDRESS OF TEST 17
056616  015474                  .WORD   TST20+2         ;;STARTING ADDRESS OF TEST 20
056620  015760                  .WORD   TST21+2         ;;STARTING ADDRESS OF TEST 21
056622  016076                  .WORD   TST22+2         ;;STARTING ADDRESS OF TEST 22
056624  016224                  .WORD   TST23+2         ;;STARTING ADDRESS OF TEST 23
056626  016476                  .WORD   TST24+2         ;;STARTING ADDRESS OF TEST 24
056630  016776                  .WORD   TST25+2         ;;STARTING ADDRESS OF TEST 25
056632  017420                  .WORD   TST26+2         ;;STARTING ADDRESS OF TEST 26
056634  017514                  .WORD   TST27+2         ;;STARTING ADDRESS OF TEST 27
056636  020002                  .WORD   TST30+2         ;;STARTING ADDRESS OF TEST 30
056640  020316                  .WORD   TST31+2         ;;STARTING ADDRESS OF TEST 31
056642  020606                  .WORD   TST32+2         ;;STARTING ADDRESS OF TEST 32
056644  021324                  .WORD   TST33+2         ;;STARTING ADDRESS OF TEST 33
056646  021636                  .WORD   TST34+2         ;;STARTING ADDRESS OF TEST 34
056650  022130                  .WORD   TST35+2         ;;STARTING ADDRESS OF TEST 35
056652  022452                  .WORD   TST36+2         ;;STARTING ADDRESS OF TEST 36
056654  023012                  .WORD   TST37+2         ;;STARTING ADDRESS OF TEST 37
056656  023466                  .WORD   TST40+2         ;;STARTING ADDRESS OF TEST 40
056660  023752                  .WORD   TST41+2         ;;STARTING ADDRESS OF TEST 41
056662  024234                  .WORD   TST42+2         ;;STARTING ADDRESS OF TEST 42
056664  024636                  .WORD   TST43+2         ;;STARTING ADDRESS OF TEST 43
056666  025006                  .WORD   TST44+2         ;;STARTING ADDRESS OF TEST 44
056670  025366                  .WORD   TST45+2         ;;STARTING ADDRESS OF TEST 45
056672  026076                  .WORD   TST46+2         ;;STARTING ADDRESS OF TEST 46
056674  026372                  .WORD   TST47+2         ;;STARTING ADDRESS OF TEST 47
056676  026774                  .WORD   TST50+2         ;;STARTING ADDRESS OF TEST 50
056700  027230                  .WORD   TST51+2         ;;STARTING ADDRESS OF TEST 51
056702  027436                  .WORD   TST52+2         ;;STARTING ADDRESS OF TEST 52
056704  030270                  .WORD   TST53+2         ;;STARTING ADDRESS OF TEST 53
056706  030542                  .WORD   TST54+2         ;;STARTING ADDRESS OF TEST 54
056710  030744                  .WORD   TST55+2         ;;STARTING ADDRESS OF TEST 55
056712  031156                  .WORD   TST56+2         ;;STARTING ADDRESS OF TEST 56
056714  031352                  .WORD   TST57+2         ;;STARTING ADDRESS OF TEST 57
056716  031532                  .WORD   TST60+2         ;;STARTING ADDRESS OF TEST 60
056720  031750                  .WORD   TST61+2         ;;STARTING ADDRESS OF TEST 61
056722  032116                  .WORD   TST62+2         ;;STARTING ADDRESS OF TEST 62
056724  032342                  .WORD   TST63+2         ;;STARTING ADDRESS OF TEST 63
056726  032456                  .WORD   TST64+2         ;;STARTING ADDRESS OF TEST 64
056730  032644                  .WORD   TST65+2         ;;STARTING ADDRESS OF TEST 65
056732  033074                  .WORD   TST66+2         ;;STARTING ADDRESS OF TEST 66
056734  033312                  .WORD   TST67+2         ;;STARTING ADDRESS OF TEST 67
056736  033540                  .WORD   TST70+2         ;;STARTING ADDRESS OF TEST 70
056740  034010                  .WORD   TST71+2         ;;STARTING ADDRESS OF TEST 71
056742  034216                  .WORD   TST72+2         ;;STARTING ADDRESS OF TEST 72
056744  034506                  .WORD   TST73+2         ;;STARTING ADDRESS OF TEST 73
056746  034734                  .WORD   TST74+2         ;;STARTING ADDRESS OF TEST 74
056750  035042                  .WORD   TST75+2         ;;STARTING ADDRESS OF TEST 75
056752  035166                  .WORD   TST76+2         ;;STARTING ADDRESS OF TEST 76
056754  035316                  .WORD   TST77+2         ;;STARTING ADDRESS OF TEST 77
056756  035450                  .WORD   TST100+2                ;;STARTING ADDRESS OF TEST 100
056760  035602                  .WORD   TST101+2                ;;STARTING ADDRESS OF TEST 101
056762  035772                  .WORD   TST102+2                ;;STARTING ADDRESS OF TEST 102
056764  036174                  .WORD   TST103+2                ;;STARTING ADDRESS OF TEST 103
056766  036322                  .WORD   TST104+2                ;;STARTING ADDRESS OF TEST 104
```

```
056770  036526                  .WORD   TST105+2        ;;STARTING ADDRESS OF TEST 105
056772  036656                  .WORD   TST106+2        ;;STARTING ADDRESS OF TEST 106
056774  037052                  .WORD   TST107+2        ;;STARTING ADDRESS OF TEST 107
056776  037250                  .WORD   TST110+2        ;;STARTING ADDRESS OF TEST 110
057000  037526                  .WORD   TST111+2        ;;STARTING ADDRESS OF TEST 111
057002  040032                  .WORD   TST112+2        ;;STARTING ADDRESS OF TEST 112
057004  041424                  .WORD   TST113+2        ;;STARTING ADDRESS OF TEST 113
057006  043152                  .WORD   TST114+2        ;;STARTING ADDRESS OF TEST 114
057010  045332                  .WORD   TST115+2        ;;STARTING ADDRESS OF TEST 115
057012  045712                  .WORD   TST116+2        ;;STARTING ADDRESS OF TEST 116
057014  050072                  .WORD   TST117+2        ;;STARTING ADDRESS OF TEST 117
057016  052360                  .WORD   TST120+2        ;;STARTING ADDRESS OF TEST 120
```

```
                                .SBTTL  ERROR HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

057020  105037  057412         $ERROR: CLRB    IBSAVE          ;;CLEAR THE ITEM BYTE SAVE LOCATION
057024  104410                         CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
057026  105237  001117         7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
057032  001775                         BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
057034  013777  001116 122114         MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
057042  032777  002000 122104         BIT     #BIT10,@SWR     ;;BELL ON ERROR?
057050  001402                         BEQ     1$              ;;NO - SKIP
057052  104401  001212                 TYPE    ,$BELL          ;;RING BELL
057056  005237  001126         1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
057062  011637  001132                 MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
057066  162737  000002 001132         SUB     #2,$ERRPC
057074  117737  122032 001130         MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
057102  032777  001000 122044         BIT     #BIT09,@SWR     ;;SEE IF LOOP ON ERROR IS SET
057110  001060                         BNE     1004$           ;;BRANCH AROUND ROUTINE IF SO
057112  122737  000177 001130         CMPB    #177,$ITEMB     ;;SEE IF THIS IS THE POWER FAIL CALL
057120  001454                         BEQ     1004$           ;;BRANCH AROUND ROUTINE IF IT IS
057122  105737  057412                 TSTB    IBSAVE          ;;SEE IF THIS IS THE 2ND ERROR CALL IN THIS ROUTINE
057126  001047                         BNE     1003$           ;;BRANCH IF SO
057130  022737  177777 057410         CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
057136  001445                         BEQ     1004$           ;;BRANCH IF SO
057140  013746  000004                 MOV     ERRVEC,-(SP)    ;;SAVE CONTENTS OF ERROR VECTOR
057144  012737  057162 000004         MOV     #1000$,ERRVEC   ;;SETUP 'TRAP' RETURN ADDRESS
057152  013737  177766 057410         MOV     177766,CPSAVE   ;;MOVE CPU ERROR REGISTER TO CPSAVE FOR TEST
057160  000406                         BR      1001$
057162  012737  177777 057410 1000$:  MOV     #-1,CPSAVE      ;;SET CPU ERROR REGISTER TIMEOUT INDICATOR
057170  012716  057176                 MOV     #1001$,(SP)     ;;SETUP RETURN ADDRESS
057174  000002                         RTI
057176  012637  000004         1001$:  MOV     (SP)+,ERRVEC    ;;RESTORE CONTENTS OF ERROR VECTOR

057202  022737  177777 057410 1002$:  CMP     #-1,CPSAVE      ;;SEE IF CPSAVE HAS CPU ERR REG TIMEOUT INDICATION
057210  001420                         BEQ     1004$           ;;BRANCH IF SO
057212  032737  000001 057410         BIT     #BIT00,CPSAVE   ;;SEE IF POWER MONITOR BIT IS SET IN CPU ERR REG
057220  001414                         BEQ     1004$           ;;BRANCH IF OK
057222  042737  000001 177766         BIC     #BIT00,177766   ;;CLEAR THE BIT FOUND SET
057230  113737  001130 057412         MOVB    $ITEMB,IBSAVE   ;;MAKE IBSAVE NON-ZERO FOR DUAL ERROR CALL
057236  112737  000177 001130         MOVB    #177,$ITEMB     ;;SET $ITEMB TO SPECIAL POWER FAIL POINTER
057244  000402                         BR      1004$           ;;BRANCH OVER IBSAVE CLEARING

057246  105037  057412         1003$:  CLRB    IBSAVE          ;;CLEAR IBSAVE SO 2ND TIME THROUGH EXITS
057252                           1004$:
057252  032777  020000 121674         BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
057260  001004                         BNE     20$             ;;SKIP TYPEOUTS
057262  004737  057414                 JSR     PC,ERRTYP       ;;GO TO USER ERROR ROUTINE
```

```
057266  104401  001217              TYPE    ,$CRLF
057272                      20$:
057272  122737  000001  001242      CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
057300  001007                      BNE     2$              ;;NO,SKIP APT ERROR REPORT
057302  113737  001130  057314      MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
057310  004737  062356              JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
057314     000              21$:    .BYTE   0
057315     000                      .BYTE   0
057316  000777              22$:    BR      22$             ;;APT ERROR LOOP
057320  105737  057412      2$:     TSTB    IBSAVF          ;;SEE IF IBSAVE IS LOADED
057324  001005                      BNE     3$              ;;BRANCH IF NOT - NO HALT ON PWR MON BIT ERROR
057326  005777  121622              TST     @SWR            ;;HALT ON ERROR
057332  100002                      BPL     3$              ;;SKIP IF CONTINUE
057334  000000                      HALT                    ;;HALT ON ERROR!
057336  104410                      CKSWR                   ;;TEST FOR CHANGE IN SOFT-SWR
057340                      3$:
057340  032777  001000  121606      BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
057346  001402                      BEQ     4$              ;;BR IF NO
057350  013716  001124              MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
057354  005737  001210      4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
057360  001402                      BEQ     5$              ;;BR IF NONE
057362  013716  001210              MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
057366                      5$:
057366  022737  054020  000042      CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
057374  001001                      BNE     6$              ;;BRANCH IF NO
057376  000000                      HALT                    ;;YES
057400                      6$:
057400  105737  057412              TSTB    IBSAVE          ;;SEE IF ITEM BYTE SAVE LOCATION HAS AN ERROR CALL
057404  001210                      BNE     7$              ;;BRANCH BACK TO CALL ORIGINAL ERROR
057406  000002                      RTI                     ;;RETURN
057410  000000              CPSAVE: .WORD   0               ;;LOCATION TO SAVE CPU ERROR REG CONTENTS
057412  000000              IBSAVE: .WORD   0               ;;LOCATION TO SAVE ITEM BYTE
```

```
                                        .SBTTL  ERROR TYPEOUT ROUTINE

                                        ;*THE ERROR TYPEOUT ROUTINE ASSEMBLES AND PRINTS INFORMATION
                                        ;*REGARDING THE DETECTION OF AN ERROR AS FOLLOWS:
                                        ;*
                                        ;*      .UNIT NUMBER, DRIVE TYPE, TEST NUMBER, ERROR NUMBER AND
                                        ;*PROGRAM COUNTER ARE PRINTED ON THE FIRST LINE;
                                        ;*      .ERROR MESSAGE IS ASSEMBLED, FORMATTED AND PRINTED ON
                                        ;*ONE OR MORE SUCCEEDING LINES;
                                        ;*      .PAIRED LINES OF ERROR HEADERS AND ERROR DATA ARE PRINTED
                                        ;*AFTER THE ERROR MESSAGE.

057414  104414                  ERRTYP: SAVREG
057416  032777  020000  121530          BIT     #SW13,@SWR      ;INHIBIT TYPEOUTS??
057424  001402                          BEQ     1$              ;NO!!
057426  000137  060254                  JMP     27$             ;YES!!

                                        ;TYPE UNIT NUMBER, DRIVE TYPE, TEST NUMBER, ERROR NUMBER, AND
                                        ;PROGRAM COUNTER
057432  104401  001217          1$:     TYPE    ,$CRLF
057436  104401  060270                  TYPE    ,ERTY00         ;TYPE 'DRV#'
057442  013746  001234                  MOV     $UNIT,-(SP)     ;;SAVE $UNIT FOR TYPEOUT
                                                                ;;TYPE DRIVE NUMBER
057446  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
057450     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
057451     000                          .BYTE   0               ;;SUPPRESS LEADING ZEROS

                                        ;TYPE 'DRIVE TYPE' RM05, RM03 OR RM02 FOR UNIT UNDER TEST
057452  013700  001276                  MOV     $BASE,R0        ;GET RM BASE ADDRESS
057456  016000  000026                  MOV     RMDT(R0),R0     ;GET DRIVE TYPE REGISTER
057462  042700  177740                  BIC     #177740,R0      ;SAVE DRIVE TYPE BITS AND
057466  012737  063257  057540          MOV     #$RM03,3$       ;GET ASCII DRIVE TYPE
057474  022700  000024                  CMP     #24,R0          ;IS DEVICE AN RM03 ?
057500  001414                          BEQ     2$              ;YES !!

057502  012737  063252  057540          MOV     #$RM02,3$       ;SAVE ASCII DRIVE TYPE
057510  022700  000025                  CMP     #25,R0          ;IS DEVICE AN RM02 ?
057514  001406                          BEQ     2$              ;YES !!

057516  012737  063264  057540          MOV     #$RM05,3$       ;SAVE ASCII DRIVE TYPE
057524  022700  000027                  CMP     #27,R0          ;IS DEVICE AN RM05 ?
057530  001004                          BNE     4$              ;NO !!
057532  104401  060325          2$:     TYPE    ,ERTY05         ;TYPE " - "
057536  104401                          TYPE                    ;TYPE DRIVE TYPE
057540  000000                  3$:     .WORD   0               ;DRIVE TYPE MESSAGE IS STORED HERE

                                        ;TYPE TEST NUMBER, ERROR NUMBER AND PROGRAM COUNTER
057542  005037  060260          4$:     CLR     TSTNMB          ;LOAD TEST NUMBER FOR
057546  013737  001226  060260          MOV     $TESTN,TSTNMB
057554  104401  060275                  TYPE    ,ERTY01         ;TYPE "TST#"
057560  013746  060260                  MOV     TSTNMB,-(SP)    ;;SAVE TSTNMB FOR TYPEOUT
                                                                ;;TYPE TEST NUMBER
057564  104403                          TYPOS                   ;;GO TYPE--OCTAL ASCII
057566     003                          .BYTE   3               ;;TYPE 3 DIGIT(S)
057567     000                          .BYTE   0               ;;SUPPRESS LEADIN(  EROS
057570  005037  060262                  CLR     ERRNMB          ;LOAD ERROR NUMBE   OR
057574  113737  001130  060262          MOVB    $ITEMB,ERRNMB   ;TYPEOUT
```

ERROR TYPEOUT ROUTINE

```
 50 057602  001406                        BEQ     5$              ;SKIP IF NO ERROR CALLED
 51 057604  104401  060305                TYPE    ,ERTY02         ;TYPE 'ERR#'
 52 057610  013746  060262                MOV     ERRNMB,-(SP)    ;;SAVE ERRNMB FOR TYPEOUT
                                                                   ;;TYPE ERROR NUMBER
    057614  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
    057616     003                        .BYTE   3               ;;TYPE 3 DIGIT(S)
    057617     000                        .BYTE   0               ;;SUPPRESS LEADING ZEROS
 53 057620  104401  060314          5$:   TYPE    ,ERTY03         ;TYPE 'PC='
 54 057624  013746  001132                MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                   ;;TYPE PROGRAM COUNTER
    057630  104403                        TYPOS                   ;;GO TYPE--OCTAL ASCII
    057632     006                        .BYTE   6               ;;TYPE 6 DIGIT(S)
    057633     001                        .BYTE   1               ;;TYPE LEADING ZEROS
 55
 56                                ;GENERATE POINTER TO ERROR TABLE UNLESS ERROR NUMBER IS 0
 57 057634  005737  060262          6$:   TST     ERRNMB          ;WAS AN ERROR CALLED?
 58 057640  001002                        BNE     7$              ;BR IF YES
 59 057642  000137  060254                JMP     27$             ;NO--EXIT
 60
 61 057646  104401  001217          7$:   TYPE    ,$CRLF          ;YES-TYPE CRLF
 62 057652  105037  060266                CLRB    BOTFLG          ;CLEAR BOT FLAG
 63 057656  105037  060267                CLRB    CHRCNT          ;CLEAR CHARACTER COUNTER
 64 057662  013700  060262                MOV     ERRNMB,R0       ;R0 POINTS TO FIRST OF
 65 057666  122700  000177                CMPB    #177,R0         ;SEE IF THIS ERROR CALL IS SPECIAL POWER FAIL CALL
 66 057672  001003                        BNE     8$              ;BRANCH IF NOT
 67 057674  012700  060332                MOV     #PFECH,R0       ;MOVE POWER FAIL ERROR CALL TABLE TO R0
 68 057700  000405                        BR      9$
 69 057702  006300          8$:   ASL     R0              ;FOUR ENTRIES IN ERROR
 70 057704  006300                ASL     R0              ;TABLE
 71 057706  006300                ASL     R0
 72 057710  062700  001532                ADD     #$ERRTB-8.,R0
 73 057714  011001          9$:   MOV     (R0),R1         ;R1 POINTS TO ERROR MESSAGE
 74                                                       ;TABLE
 75 057716  001507                        BEQ     19$             ;BRANCH IF NO ERROR MESSAGE
 76
 77                                ;TYPE THE ERROR MESSAGE
 78 057720  012102          10$:  MOV     (R1)+,R2        ;R2=ADDRESS OF MESSAGE STRING
 79 057722  001505                BEQ     19$             ;BRANCH IF END OF MESSAGE
 80 057724  010237  060072                MOV     R2,18$          ;LOAD ADDRESS OF STRING
 81 057730  005037  060264                CLR     BOTADR          ;CLEAR BOT ADDRESS
 82 057734  112203          11$:  MOVB    (R2)+,R3        ;END OF STRING??
 83 057736  001454                BEQ     17$             ;YES!!
 84 057740  122703  000015                CMPB    #CR,R3          ;CARRIAGE RETURN??
 85 057744  001003                BNE     12$             ;NO!!
 86 057746  105037  060267                CLRB    CHRCNT          ;YES-CLEAR CHAR COUNT
 87 057752  000770                BR      11$             ;GET NEXT CHARACTER
 88 057754  122703  000012          12$:  CMPB    #LF,R3          ;LINE FEED??
 89 057760  001765                BEQ     11$             ;YES-GET NEXT CHARACTER
 90 057762  122703  000011                CMPB    #HT,R3          ;HORIZONTAL TAB??
 91 057766  001007                BNE     14$             ;NO!!
 92 057770  105237  060267          13$:  INCB    CHRCNT          ;ADJUST CHARACTER COUNT
 93 057774  132737  000007  060267        BITB    #7,CHRCNT
 94 060002  001372                BNE     13$
 95 060004  000407                BR      15$
 96 060006  105237  060267          14$:  INCB    CHRCNT          ;INCREMENT CHARACTER COUNT
 97 060012  122703  000040                CMPB    #' ,R3          ;SPACE??
 98 060016  001002                BNE     15$             ;NO!!
```

```
 99 060020  010237  060264                    MOV     R2,BOTADR       ;SAVE ADDRESS OF SPACE
100 060024  122737  000100  060267  15$:      CMPB    #64.,CHRCNT     ;END OF LINE??
101 060032  103340                            BHIS    11$             ;NO!!
102 060034  013704  060264                    MOV     BOTADR,R4       ;GET ADDRESS OF LAST SPACE
103 060040  001007                            BNE     16$             ;BRANCH IF SPACE DETECTED
104 060042  104401  001217                    TYPE    ,$CRLF          ;TYPE CRLF
105 060046  105037  060267                    CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
106 060052  013702  060072                    MOV     18$,R2          ;SET UP R2 FOR TESTING
107 060056  000726                            BR      11$
108 060060  105044                    16$:    CLRB    -(R4)           ;REPLACE SPACE
109 060062  112737  177777  060266            MOVB    #-1,BOTFLG      ;SET BOT FLAG
110 060070  104401                    17$:    TYPE                    ;TYPE ERROR MESSAGE STRING
111 060072  000000                    18$:    .WORD                   ;STRING ADDRESS GOES HERE
112 060074  105737  060266                    TSTB    BOTFLG          ;WAS STRING TRUNCATED??
113 060100  001707                            BEQ     10$             ;NO!!
114 060102  104401  001217                    TYPE    ,$CRLF          ;YES-TYPE CRLF
115 060106  105037  060266                    CLRB    BOTFLG          ;CLEAR BOT FLAG
116 060112  105037  060267                    CLRB    CHRCNT          ;CLEAR CHARACTER COUNT
117 060116  013702  060264                    MOV     BOTADR,R2       ;SETUP R2 FOR TESTING
118 060122  010237  060072                    MOV     R2,18$          ;SETUP 18$ FOR TYPING
119 060126  112742  000040                    MOVB    #' ,-(R2)       ;RESTORE SPACE
120 060132  105722                            TSTB    (R2)+           ;RESTORE R2
121 060134  000677                            BR      11$             ;TYPE REST OF STRING
122
123                                   ;TYPE ERROR HEADER AND ERROR DATA
124 060136  016001  000002            19$:    MOV     2(R0),R1        ;R1 POINTS TO ERROR HEADER TABLE
125 060142  001444                            BEQ     27$             ;BRANCH IF NO HEADER
126 060144  104401  001217                    TYPE    ,$CRLF          ;(ASSUME NO DATA)
127 060150  016002  000004                    MOV     4(R0),R2        ;R2 POINTS TO DATA ADDRESS TABLE
128 060154  016003  000006                    MOV     6(R0),R3        ;R3 POINTS TO FORMAT TABLE
129 060160  012137  060170            20$:    MOV     (R1)+,21$       ;PUT HEADER ADDRESS FOR TYPE
130 060164  001433                            BEQ     27$             ;BRANCH IF END OF HEADERS
131                                                                   ;(ASSUME END OF DATA)
132 060166  104401                            TYPE
133 060170  000000                    21$:    .WORD   0               ;HEADER ADDRESS GOES HERE
134 060172  104401  001217                    TYPE    ,$CRLF
135 060176  005702                            TST     R2              ;DATA WITH HEADER??
136 060200  001767                            BEQ     20$             ;NO!!
137 060202  012204                            MOV     (R2)+,R4        ;R4 POINTS TO DATA ADDRESS
138 060204  012305                            MOV     (R3)+,R5        ;R5 POINTS TO FORMAT
139 060206  105725                    22$:    TSTB    (R5)+           ;WHAT KIND OF DATA??
140 060210  100407                            BMI     24$             ;BINARY
141 060212  001403                            BEQ     23$             ;OCTAL
142 060214  013446                            MOV     @(R4)+,-(SP)    ;DECIMAL
143 060216  104405                            TYPDS
144 060220  000405                            BR      25$
145 060222  013446                    23$:    MOV     @(R4)+,-(SP)
146 060224  104402                            TYPOC
147 060226  000402                            BR      25$
148 060230  013446                    24$:    MOV     @(R4)+,-(SP)
149 060232  104406                            TYPBN
150 060234  005714                    25$:    TST     (R4)            ;MORE DATA??
151 060236  001403                            BEQ     26$             ;NO!!
152 060240  104401  060322                    TYPE    ,ERTY04         ;YES-TYPE 2 SPACES
153 060244  000760                            BR      22$             ;AND CONTINUE
154 060246  104401  001217            26$:    TYPE    ,$CRLF          ;TYPE ONE BLANK LINE
155 060252  000742                            BR      20$             ;BEFORE NEXT HEADER
```

```
060254  104415                          27$:    RESREG
060256  000207                                  RTS     PC

060260  000000                          TSTNMB: .WORD   0               ;TEST NUMBER
060262  000000                          ERRNMB: .WORD   0               ;ERROR NUMBER
060264  000000                          BOTADR: .WORD   0               ;BEGINNING OF TEXT ADDRESS
060266     000                          BOTFLG: .BYTE   0               ;BOT FLAG
060267     000                          CHRCNT: .BYTE   0               ;CHARACTER COUNT

060270     104     122     126          ERTY00: .ASCIZ  @DRV#@
060275     054     040     124          ERTY01: .ASCIZ  @, TEST#@
060305     054     040     105          ERTY02: .ASCIZ  @, ERR#@
060314     054     040     120          ERTY03: .ASCIZ  @, PC=@
060322     040     040     000          ERTY04: .ASCIZ  @  @
060325     040     055     040          ERTY05: .ASCIZ  @ - @
                                        .EVEN
060332  060342  060430  060446          PFECH:  PFECH1,PFECH2,PFECH3,PFECH4 ;WORDS DEFINING TABLES BELOW
060342  060346  000000                  PFECH1: .+4,0
060346     120     117     127                  .ASCIZ  ?POWER MONITOR BIT IN CPU ERROR REGISTER FOUND SET?
                                                .EVEN
060430  060434  000000                  PFECH2: .+4,0
060434     103     120     125                  .ASCIZ  ?CPUERREG?
                                                .EVEN
060446  060450                          PFECH3: .+2
060450  057410  000000                          .WORD   CPSAVE,0
060454  060456                          PFECH4: .+2
060456     000     000                          .BYTE   0,0
```

```
                                .SBTTL  TTY INPUT ROUTINE

                                ;;*****************************************************************
                                .ENABL  LSB
060460  000000                  $TKCNT: .WORD   0               ;;NUMBER OF ITEMS IN QUEUE
060462  000C00                  $TKQIN: .WORD   0               ;;INPUT POINTER
060464  000000                  $TKQOUT: .WORD  0               ;;OUTPUT POINTER
060466                          $TKQSRT: .BLKB  1               ;;TTY KEYBOARD QUEUE
        060467                  $TKQEND=.
                                .EVEN

                                ;*TK INITIALIZE ROUTINE
                                ;*THIS ROUTINE WILL INITIALIZE THE TTY KEYBOARD INPUT QUEUE
                                ;*SETUP THE INTERRUPT VECTOR AND TURN ON THE KEYBOARD INTERRUPT
                                ;
                                ;*CALL:
                                ;*      JSR     PC,$TKINT
                                ;*      RETURN
                                ;
060470  005037  060460          $TKINT: CLR     $TKCNT          ;;CLEAR COUNT OF ITEMS IN QUEUE
060474  012737  060466  060462          MOV     #$TKQSRT,$TKQIN ;;MOVE THE STARTING ADDRESS OF THE
060502  013737  060462  060464          MOV     $TKQIN,$TKQOUT  ;;QUEUE INTO THE INPUT & OUTPUT POINTERS.
060510  012737  060540  000060          MOV     #$TKSRV,@#TKVEC ;;INITIALIZE THE KEYBOARD VECTOR
060516  012737  000200  000062          MOV     #200,@#TKVEC+2  ;;'BR' LEVEL 4
060524  005777  120432                  TST     @$TKB           ;;CLEAR DONE FLAG
060530  012777  000100  120422          MOV     #100,@$TKS      ;;ENABLE TTY KEYBOARD INTERRUPT
060536  000207                  RTS     PC              ;;RETURN TO CALLER

                                ;*TK SERVICE ROUTINE
                                ;*THIS ROUTINE WILL SERVICE THE TTY KEYBOARD INTERRUPT
                                ;*BY READING THE CHARACTER FROM THE INPUT BUFFER AND PUTTING
                                ;*IT IN THE QUEUE.
                                ;*IF THE CHARACTER IS A 'CONTROL-C' (^C) $TKINT IS CALLED AND
                                ;*UPON RETURN EXIT IS MADE TO THE 'CONTROL-C' RESTART ADDRESS (SHUT)
                                ;
060540  117746  120416          $TKSRV: MOVB    @$TKB,-(SP)     ;;PICKUP THE CHARACTER
060544  042716  177600                  BIC     #^C177,(SP)     ;;STRIP THE JUNK
060550  021627  000021                  CMP     (SP),#$XON      ;;IS IT A RANDOM XON?
060554  001002                  BNE     30$             ;;BRANCH IF NO
060556  005726                  TST     (SP)+           ;;CLEAN RANDOM XON OFF STACK
060560  000002                  RTI                     ;;RETURN
060562                          30$:
060562  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL C?
060566  001007                  BNE     1$              ;;BRANCH IF NO
060570  104401  061666                  TYPE    ,$CNTLC         ;;TYPE A CONTROL-C (^C)
060574  004737  060470                  JSR     PC,$TKINT       ;;INIT THE KEYBOARD
060600  005726                  TST     (SP)+           ;;CLEAN UP STACK
060602  000137  061730                  JMP     SHUT            ;;CONTROL C RESTART
060606  021627  000007          1$:     CMP     (SP),#7         ;;IS IT A CONTROL G?
060612  001004                  BNE     2$              ;;BRANCH IF NO
060614  022737  000176  001154          CMP     #SWREG,SWR      ;;IS SOFT-SWR SELECTED?
060622  001500                  BEQ     6$              ;;GO TO SWR CHANGE

060624                          2$:
060624  022737  000001  060460          CMP     #1,$TKCNT       ;;IS THE QUEUE FULL?
060632  001004                  BNE     3$              ;;BRANCH IF NO
060634  104401  001212                  TYPE    ,$BELL          ;;RING THE TTY BELL
```

```
060640  005726                          TST     (SP)+           ;;CLEAN CHARACTER OFF OF STACK
060642  000451                          BR      5$              ;;EXIT
060644  021627  000023          3$:     CMP     (SP),#23        ;;IS IT A CONTROL-S?
060650  001021                          BNE     32$             ;;BRANCH IF NO
060652  005077  120302                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
060656  005726                          TST     (SP)+           ;;CLEAN CHAR OFF STACK
060660  105777  120274          3 $:    TSTB    @$TKS           ;;WAIT FOR A CHAR
060664  100375                          BPL     31$             ;;LOOP UNTIL ITS THERE
060666  117746  120270                  MOVB    @$TKB,-(SP)     ;;GET THE CHARACTER
060672  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
060676  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
060702  001366                          BNE     31$             ;;BRANCH IF NO
060704  012777  000100  120246          MOV     #100,@$TKS      ;;REENABLE TTY KEYBOARD INTERRUPTS
060712  000002                          RTI                     ;;RETURN
060714  005237  060460          32$:    INC     $TKCNT          ;;COUNT THIS CHARACTER
060720  021627  000140                  CMP     (SP),#140       ;;IS IT UPPER CASE?
060724  002405                          BLT     4$              ;;BRANCH IF YES
060726  021627  000175                  CMP     (SP),#175       ;;IS IT A SPECIAL CHAR?
060732  003002                          BGT     4$              ;;BRANCH IF YES
060734  042716  000040                  BIC     #40,(SP)        ;;MAKE IT UPPER CASE
060740  112677  177516          4$:     MOVB    (SP)+,@$TKQIN   ;;AND PUT IT IN QUEUE
060744  005237  060462                  INC     $TKQIN          ;;UPDATE THE POINTER
060750  023727  060462  060467          CMP     $TKQIN,#$TKQEND ;;GO OFF THE END?
060756  001003                          BNE     5$              ;;BRANCH IF NO
060760  012737  060466  060462          MOV     #$TKQSRT,$TKQIN ;;RESET THE POINTER
060766  000002                  5$:     RTI                     ;;RETURN

                                ;;*****************************************************************
                                ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
                                ;*ROUTINE IS ENTERED  FROM THE TRAP HANDLER, AND WILL
                                ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP
                                ;*CALL WHEN OPERATING IN TTY INTERRUPT MODE.
060770  022737  000176  001154  $CKSWR: CMP     #SWREG,SWR      ;;IS THE SOFT-SWR SELECTED
060776  001124                          BNE     15$             ;;EXIT IF NOT
061000  105777  120154                  TSTB    @$TKS           ;;IS A CHAR WAITING?
061004  100121                          BPL     15$             ;;IF NOT, EXIT
061006  117746  120150                  MOVB    @$TKB,-(SP)     ;;YES
061012  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT  7-BIT ASCII
061016  021627  000007                  CMP     (SP),#7         ;;IS IT A CONTROL-G?
061022  001300                          BNE     2$              ;;IF NOT, PUT IT IN THE TTY QUEUE
                                                                ;;AND EXIT

                                ;;*****************************************************************
                                ;*CONTROL IS PASSED TO THIS POINT FROM EITHER THE TTY INTERRUPT SERVICE
                                ;*ROUTINE OR FROM THE SOFTWARE SWITCH REGISTER TRAP CALL, AS A RESULT OF A
                                ;*CONTROL-G BEING TYPED, AND THE SOFTWARE SWITCH REGISTER BEING SELECTED.
061024  123727  001150  000001  6$:     CMPB    $AUTOB,#1       ;;ARE WE RUNNING IN AUTO-MODE?
061032  001674                          BEQ     2$              ;;BRANCH IF YES
061034  005726                          TST     (SP)+           ;;CLEAR CONTROL-G OFF STACK
061036  004737  060470                  JSR     PC,$TKINT       ;;FLUSH THE TTY INPUT QUEUE
061042  005077  120112                  CLR     @$TKS           ;;DISABLE TTY KEYBOARD INTERRUPTS
061046  112737  000001  001151          MOVB    #1,$INTAG       ;;SET INTERRUPT MODE INDICATOR

061054  104401  061700                  TYPF    ,$CNTLG         ;;ECHO THE CONTROL-G (^G)
061060  104401  061705          $GTSWR: TYPE    ,$MSWR          ;;TYPE CURRENT CONTENTS
061064  013746  000176                  MOV     SWREG,-(SP)     ;;SAVE SWREG FOR TYPEOUT
061070  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
```

```
061072  104401  061776                  TYPE    ,$MNEW          ;;PROMPT FOR NEW SWR
061076  005046                  19$:    CLR     -(SP)           ;;CLEAR COUNTER
061100  005046                          CLR     -(SP)           ;;THE NEW SWR
061102  105777  120052          7$:     TSTB    @$TKS           ;;CHAR THERE?
061106  100375                          BPL     7$              ;;IF NOT TRY AGAIN

061110  117746  120046                  MOVB    @$TKB,-(SP)     ;;PICK UP CHAR
061114  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII

061120  021627  000003                  CMP     (SP),#3         ;;IS IT A CONTROL-C?
061124  001015                          BNE     9$              ;;BRANCH IF NOT
061126  104401  061666                  TYPE    ,$CNTLC         ;;YES, ECHO CONTROL-C (^C)
061132  062706  000006                  ADD     #6,SP           ;;CLEAN UP STACK
061136  123727  001151  000001          CMPB    $INTAG,#1       ;;REENABLE TTY KEYBOARD INTERRUPTS?
061144  001003                          BNE     8$              ;;BRANCH IF NO
061146  012777  000100  120004          MOV     #100,@$TKS      ;;ALLOW TTY KEYBOARD INTERRUPTS
061154  000137  061730          8$:     JMP     SHUT            ;;CONTROL-C RESTART


061160  021627  000025          9$:     CMP     (SP),#25        ;;IS IT A CONTROL-U?
061164  001005                          BNE     10$             ;;BRANCH IF NOT
061166  104401  061673                  TYPE    ,$CNTLU         ;;YES, ECHO CONTROL-U (^U)
061172  062706  000006          20$:    ADD     #6,SP           ;;IGNORE PREVIOUS INPUT
061176  000737                          BR      19$             ;;LET'S TRY IT AGAIN


061200  021627  000015          10$:    CMP     (SP),#15        ;;IS IT A <CR>?
061204  001022                          BNE     16$             ;;BRANCH IF NO
061206  005766  000004                  TST     4(SP)           ;;YES, IS IT THE FIRST CHAR?
061212  001403                          BEQ     11$             ;;BRANCH IF YES
061214  016677  000002  117732          MOV     2(SP),@SWR      ;;SAVE NEW SWR
061222  062706  000006          11$:    ADD     #6,SP           ;;CLEAR UP STACK
061226  104401  001217          14$:    TYPE    ,$CRLF          ;;ECHO <CR> AND <LF>
061232  123727  001151  000001          CMPB    $INTAG,#1       ;;RE-ENABLE TTY KBD INTERRUPTS?
061240  001003                          BNE     15$             ;;BRANCH IF NOT
061242  012777  000100  117710          MOV     #100,@$TKS      ;;RE-ENABLE TTY KBD INTERRUPTS
061250  000002          15$:    RTI                             ;;RETURN
061252  004737  055756          16$:    JSR     PC,$TYPEC       ;;ECHO CHAR
061256  021627  000060                  CMP     (SP),#60        ;;CHAR < 0?
061262  002420                          BLT     18$             ;;BRANCH IF YES
061264  021627  000067                  CMP     (SP),#67        ;;CHAR > 7?
061270  003015                          BGT     18$             ;;BRANCH IF YES
061272  042726  000060                  BIC     #60,(SP)+       ;;STRIP-OFF ASCII
061276  005766  000002                  TST     2(SP)           ;;IS THIS THE FIRST CHAR
061302  001403                          BEQ     17$             ;;BRANCH IF YES
061304  006316                          ASL     (SP)            ;;NO, SHIFT PRESENT
061306  006316                          ASL     (SP)            ;;    CHAR OVER TO MAKE
061310  006316                          ASL     (SP)            ;;    ROOM FOR NEW ONE.
061312  005266  000002          17$:    INC     2(SP)           ;;KEEP COUNT OF CHAR
061316  056616  177776                  BIS     -2(SP),(SP)     ;;SET IN NEW CHAR
061322  000667                          BR      7$              ;;GET THE NEXT ONE
061324  104401  001216          18$:    TYPE    ,$QUES          ;;TYPE ?<CR><LF>
061330  000720                          BR      20$             ;;SIMULATE CONTROL-U
                                .DSABL  LSB

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

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

061332  011646                 $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC AND
061334  016666  000004  000002          MOV     4(SP),2(SP)     ;;THE PS
061342  005066  000004                  CLR     4(SP)           ;;GET READY FOR A CHARACTER
061346  005046                          CLR     -(SP)           ;;PUT NEW PS ON STACK
061350  012746  061356                  MOV     #64$,-(SP)      ;;PUT NEW PC ON  STACK
061354  000002                          RTI                     ;;POP NEW PC AND PS
061356                          64$:
061356  005737  060460          1$:     TST     $TKCNT          ;;WAIT ON A CHARACTER
061362  001775                          BEQ     1$
061364  005337  060460                  DEC     $TKCNT          ;;DECREMENT THE COUNTER
061370  117766  177070  000004          MOVB    @$TKQOUT,4(SP)  ;;GET ONE CHARACTER
061376  005237  060464                  INC     $TKQOUT         ;;UPDATE THE POINTER
061402  023727  060464  060467          CMP     $TKQOUT,#$TKQEND ;;DID IT GO OFF OF THE END?
061410  001003                          BNE     2$              ;;BRANCH IF NO
061412  012737  060466  060464          MOV     #$TKQSRT,$TKQOUT ;;RESET THE POINTER
061420  000002                  2$:     RTI                     ;;RETURN
                                ;;**************************************************************
                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                ;*CALL:
                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

061422  010346                 $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
061424  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
061426  012703  061656          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
061432  022703  061666          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
061436  101456                          BLOS    4$              ;;BR IF YES
061440  104411                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
061442  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
061444  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
061450  001022                          BNE     5$              ;;BR IF NO
061452  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
061454  001007                          BNE     6$              ;;BR IF NO
061456  112737  000134  061654          MOVB    #'\,9$          ;;TYPE A BACK SLASH
061464  104401  061654                  TYPE    ,9$
061470  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
061474  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
061476  020327  061656                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
061502  103434                          BLO     4$              ;;BR IF YES
061504  111337  061654                  MOVB    (R3),9$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
061510  104401  061654                  TYPE    ,9$             ;;GO TYPE
061514  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
061516  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
061520  001406                          BEQ     7$              ;;BR IF NO
061522  112737  000134  061654          MOVB    #'\,9$          ;;TYPE A BACK SLASH
061530  104401  061654                  TYPE    ,9$
061534  005016                          CLR     (SP)            ;;CLEAR THE RUBOUT KEY
061536  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
061542  001003                          BNE     8$              ;;BR IF NO
```

```
TTY INPUT ROUTINE

061544  104401  061673                    TYPE    ,$CNTLU         ;;TYPE A CONTROL 'U'
061550  000726                            BR      1$              ;;GO START OVER
061552  122713  000022          8$:       CMPB    #22,(R3)        ;;IS CHARACTER A '^R'?
061556  001011                            BNE     3$              ;;BRANCH IF NO
061560  105013                            CLRB    (R3)            ;;CLEAR THE CHARACTER
061562  104401  001217                    TYPE    ,$CRLF          ;;TYPE A 'CR' & 'LF'
061566  104401  061656                    TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
061572  000717                            BR      2$              ;;GO PICKUP ANOTHER CHACTER
061574  104401  001216          4$:       TYPE    ,$QUES          ;;TYPE A '?'
061600  000712                            BR      1$              ;;CLEAR THE BUFFER AND LOOP
061602  111337  061654          3$:       MOVB    (R3),9$         ;;ECHO THE CHARACTER
061606  104401  061654                    TYPE    ,9$
061612  122723  000015                    CMPB    #15,(R3)+       ;;CHECK FOR RETURN
061616  001305                            BNE     2$              ;;LOOP IF NOT RETURN
061620  105063  177777                    CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
061624  104401  001220                    TYPE    ,$LF            ;;TYPE A LINE FEED
061630  005726                            TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
061632  012603                            MOV     (SP)+,R3        ;;RESTORE R3
061634  011646                            MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
061636  016666  000004  000002            MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
061644  012766  061656  000004            MOV     #$TTYIN,4(SP)
061652  000002                            RTI                     ;;RETURN
061654     000          9$:       .BYTE   0                       ;;STORAGE FOR ASCII CHAR. TO TYPE
061655     000                    .BYTE   0                       ;;TERMINATOR
061656                  $TTYIN:   .BLKB   8.                      ;;RESERVE 8 BYTES FOR TTY INPUT
061666     136  103  015  $CNTLC:  .ASCIZ  /^C/<15><12>          ;;CONTROL 'C'
061673     136  125  015  $CNTLU:  .ASCIZ  /^U/<15><12>          ;;CONTROL 'U'
061700     136  107  015  $CNTLG:  .ASCIZ  /^G/<15><12>          ;;CONTROL 'G'
061705     015  012  123  $MSWR:   .ASCIZ  <15><12>/SWR = /
061716     040  040  116  $MNEW:   .ASCIZ  /  NEW = /
                          .EVEN

2
3 061730  005737  000042          SHUT:   TST     @#42            ;ANY MONITOR PRESENT ?
4 061734  001002                          BNE     1$              ;BR IF YES
5 061736  000137  004652                  JMP     START           ;GO TO START
6 061742  005037  001300          1$:     CLR     $DEVM           ;FUDGE NO DRIVES IN MAP
7 061746  000137  053622                  JMP     $EOP            ;RETURN TO $EOP
```

```
                        .SBTTL  READ AN OCTAL NUMBER FROM THE TTY

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

061752  011646         $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
061754  016666 000004 000002    MOV     4(SP),2(SP)     ;;INPUT NUMBER
061762  010046                 MOV     R0,-(SP)        ;;PUSH R0 ON STACK
061764  010146                 MOV     R1,-(SP)        ;;PUSH R1 ON STACK
061766  010246                 MOV     R2,-(SP)        ;;PUSH R2 ON STACK
061770  104412                 1$:     RDLIN           ;;READ AN ASCIZ LINE
061772  012600                 MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
061774  005001                 CLR     R1              ;;CLEAR DATA WORD
061776  005002                 CLR     R2
062000  112046                 2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
062002  001412                 BEQ     3$              ;;IF ZERO GET OUT
062004  006301                 ASL     R1              ;;*2
062006  006102                 ROL     R2
062010  006301                 ASL     R1              ;;*4
062012  006102                 ROL     R2
062014  006301                 ASL     R1              ;;*8
062016  006102                 ROL     R2
062020  042716 177770         BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
062024  062601                 ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
062026  000764                 BR      2$              ;;LOOP
062030  005726                 3$:     TST     (SP)+   ;;CLEAN TERMINATOR FROM STACK
062032  010166 000012          MOV     R1,12(SP)       ;;SAVE THE RESULT
062036  010237 062052          MOV     R2,$HIOCT
062042  012602                 MOV     (SP)+,R2        ;;POP STACK INTO R2
062044  012601                 MOV     (SP)+,R1        ;;POP STACK INTO R1
062046  012600                 MOV     (SP)+,R0        ;;POP STACK INTO R0
062050  000002                 RTI                     ;;RETURN
062052  000000                 $HIOCT: .WORD   0       ;;HIGH ORDER BITS GO HERE
```

```
                                .SBTTL  TRAP DECODER

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

062054  016646  000002          $TRAP:  MOV     2(SP),-(SP)     ;;ASSUME THE STATUS OF
062060  042716  000020                  BIC     #20,(SP)        ;; THE CALLER--DO NOT ALLOW
062064  012746  062072                  MOV     #1$,-(SP)       ;; T-BIT TRAPS
062070  000002                          RTI                     ;;SET THE NEW STATUS
062072  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
062074  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
062100  005740                          TST     -(R0)           ;;BACKUP BY 2
062102  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
062104  006300                          ASL     R0              ;;POSITION FOR INDEXING
062106  016000  062126                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
062112  000200                          RTS     R0              ;;GO TO ROUTINE


                                ;;THIS IS USE TO HANDLE THE ''GETPRI'' MACRO

062114  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
062116  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
062124  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                                ;*BY THE ''TRAP'' INSTRUCTION.

                                ;       ROUTINE
                                ;       -------
062126  062114                  $TRPAD: .WORD   $TRAP2
062130  055544                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
062132  055342                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
062134  055316                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
062136  055356                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
062140  055072                          $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)
062142  055016                          $TYPBN  ;;CALL=TYPBN    TRAP+6(104406)  TYPE BINARY (ASCII) NUMBER

062144  061060                          $GTSWR  ;;CALL=GTSWR    TRAP+7(104407)  GET SOFT-SWR SETTING

062146  060770                          $CKSWR  ;;CALL=CKSWR    TRAP+10(104410) TEST FOR CHANGE IN SOFT-SWR
062150  061332                          $RDCHR  ;;CALL=RDCHR    TRAP+11(104411) TTY TYPEIN CHARACTER ROUTINE
062152  061422                          $RDLIN  ;;CALL=RDLIN    TRAP+12(104412) TTY TYPEIN STRING ROUTINE
062154  061752                          $RDOCT  ;;CALL=RDOCT    TRAP+13(104413) READ AN OCTAL NUMBER FROM TTY
062156  054722                          $SAVREG ;;CALL=SAVREG   TRAP+14(104414) SAVE R0-R5 ROUTINE
062160  054760                          $RESREG ;;CALL=RESREG   TRAP+15(104415) RESTORE R0-R5 ROUTINE
```

```
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;*****************************************************************
                                ;POWER DOWN ROUTINE
062162  012737  062322  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
062170  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
062176  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
062200  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
062202  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
062204  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
062206  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
062210  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
062212  017746  116736                  MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
062216  010637  062326                  MOV     SP,$SAVR6       ;;SAVE SP
062222  012737  062234  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
062230  000000                          HALT
062232  000776                          BR      .-2             ;;HANG UP

                                ;;*****************************************************************
                                ;POWER UP ROUTINE
062234  012737  062322  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
062242  013706  062326                  MOV     $SAVR6,SP       ;;GET SP
062246  005037  062326                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
062252  005237  062326          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
062256  001375                          BNE     1$              ;;OF  WORD
062260  012677  116670                  MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
062264  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
062266  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
062270  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
062272  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
062274  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
062276  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
062300  012737  062162  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
062306  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
062314  104401                          TYPE                    ;REPORT THE POWER FAILURE
062316  062330          $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
062320  000002                  RTI
062322  000000          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
062324  000776                  BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
062326  000000          $SAVR6: 0                       ;;PUT THE SP HERE
062330     015     012     120  $POWER: .ASCIZ  <15><12>'POWER'
                                .EVEN
```

```
                                .SBTTL  APT COMMUNICATIONS ROUTINE

                                ;;*****************************************************************
062340  112737  000001  062604  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
062346  112737  000001  062602  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
062354  000403                          BR      $ATYC
062356  112737  000001  062604  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
062364                          $ATYC:
062364  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
062366  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
062370  105737  062602                  TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
062374  001450                          BEQ     5$              ;;IF NOT:  BR
062376  122737  000001  001242          CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
062404  001031                          BNE     3$              ;;IF NOT:  BR
062406  132737  000100  001243          BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
062414  001425                          BEQ     3$              ;;IF NOT:  BR
062416  017600  000004                  MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
062422  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
062430  005737  001222          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
062434  001375                          BNE     1$              ;;IF NOT:  WAIT
062436  010037  001236                  MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
062442  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
062444  001376                          BNE     2$
062446  163700  001236                  SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
062452  006200                          ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
062454  010037  001240                  MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
062460  012737  000004  001222          MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
062466  000413                          BR      5$
062470  017637  000004  062514  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
062476  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
062504  013746  177776                  MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
062510  004737  055544                  JSR     PC,$TYPE        ;;CALL TYPE MACRO
062514  000000                  4$:     .WORD   0
062516                          5$:
062516  105737  062604          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
062522  001416                          BEQ     12$             ;;IF NOT:  BR
062524  005737  001242                  TST     $ENV            ;;RUNNING UNDER APT?
062530  001413                          BEQ     12$             ;;IF NOT:  BR
062532  005737  001222          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
062536  001375                          BNE     11$             ;;IF NOT:  WAIT
062540  017637  000004  001224          MOV     @4(SP),$FATAL   ;;GET ERROR #
062546  062766  000002  000004          ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
062554  005237  001222                  INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
062560  105037  062604          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
062564  105037  062603                  CLRB    $LFLG           ;;CLEAR LOG FLAG
062570  105037  062602                  CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
062574  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
062576  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
062600  000207                          RTS     PC              ;;RETURN
062602     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
062603     000          $LFLG:  .BYTE   0               ;;LOG FLAG
062604     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
                                .EVEN
        000200          APTSIZE = 200
        000001          APTENV  = 001
        000100          APTSPOOL= 100
        000040          APTCSUP = 040
```

```
                                        .SBTTL  CONSOLE MESSAGES
     2
     3 062606      075      000          EQUALS: .ASCIZ  @=@
     4 062610      101      114      114 ALL:    .ASCIZ  @ALL@<CRLF>
     5 062615      040      077      040 QUES:   .ASCIZ  @ ? @
     6 062621      054      040      000 COMMA:  .ASCIZ  @, @
     7 062624      200      124      131 MSHELP: .ASCIZ  <CRLF>@TYPE HELP TEXT (L) N ? @
     8 062655      200      122      115 CNSL01: .ASCIZ  <CRLF>@RMCS1=@
     9 062665      040      114      111 CNSL02: .ASCIZ  @ LIMITS - LO= 160000, HI= 17XXXX@<CRLF>
    10 062727      122      115      126 CNSL03: .ASCIZ  @RMVEC=@
    11 062736      040      114      111 CNSL04: .ASCIZ  @ LIMITS - LO= 0, HI= 1000@<CRLF><LF>
    12 062772      200      124      131 CNSL07: .ASCII  <CRLF>@TYPE ''A'' TO TEST ALL DRIVES, OR TYPE DRIVE NUMBER(S)@
    13 063057      200      101      116         .ASCIZ  <CRLF>@AND TERMINATE INPUT WITH A CARRIAGE RETURN.@
    14 063134      200              CNSL08: .ASCII  <CRLF>
    15 063135      040      077      111 CNSL09: .ASCIZ  @ ?ILLEGAL INPUT@<CRLF>
    16 063156      200      104      122 DRIVES: .ASCIZ  <CRLF>/DRIVE(S) TO BE TESTED/
    17 063205      116      117      116 NONE:   .ASCIZ  /NONE/
    18 063212      200      104      122 MSDRVS: .ASCIZ  <CRLF>/DRIVE(S): /
    19 063226      104      122      111 MSGDRV: .ASCIZ  /DRIVE/
    20 063234      200      125      116 SYSTAT: .ASCIZ  <CRLF>/UNIT STATUS:/
    21 063252      122      115      060 $RM02:  .ASCIZ  /RM02/
    22 063257      122      115      060 $RM03:  .ASCIZ  /RM03/
    23 063264      122      115      060 $RM05:  .ASCIZ  /RM05/
    24 063271      040      116      117 NOTRM:  .ASCIZ  @ NOT AN RM05/3/2@
    25 063312      040      114      117 LODEV:  .ASCIZ  / LOAD DEVICE/
    26 063327      040      116      117 NOTPRS: .ASCIZ  / NOT PRESENT/
    27 063344      040      116      117 NOTAVL: .ASCIZ  / NOT AVAILABLE/
    28 063363      040      117      106 UNTOFF: .ASCIZ  / OFFL''NE/
    29 063374      040      117      116 UNTON:  .ASCIZ  / ONLI...E/
    30 063404      116      000          N:     .ASCIZ  /N/
    31 063406      131      000          Y:     .ASCIZ  /Y/
    32 063410      040                  BLNKS4: .ASCII  / /
    33 063411      040                  BLNKS3: .ASCII  / /
    34 063412      040                  BLNKS2: .ASCII  / /
    35 063413      040      000         BLNKS1: .ASCIZ  / /
    36                                  .EVEN
```

```
                        .SBTTL  FUNCTION CODE TABLE

                        ;THE FUNCTION CODE TABLE IS USED TO DEFINE STATUS CONDITIONS FOR
                        ;EACH FUNCTION CODE.  BIT USAGE IS AS FOLLOWS:

                        ;       ATA - BIT 15 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF ATA SHOULD BE SET WHEN THE FUNCTION CODE IS EXECUTED, OTHERWISE,
                        ;BIT 15 IS ZERO, INDICATING THAT ATA SHOULD NOT NORMALLY BE SET.
                        ;NOTE THAT ATA MAY BE SET WHEN A COMMAND IS EXECUTED EVEN THOUGH
                        ;IT IS NOT EXPECTED AS A RESULT OF THE COMMAND.

                        ;       WCE - BIT 14 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF WRITE CHECK ERRORS ARE ENABLED AS A FUNCTION OF THE COMMAND.

                        ;       OPI - BIT 13 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF OPI ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                        ;       IVC - BIT 12 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF IVC ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.

                        ;       WLE - BIT 11 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                        ;THE WRITE ERRORS WHICH ARE ENABLED ARE 'WLE', 'WCF', 'DPE', 'UPE'.

                        ;       IAE - BIT 10 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF INVALID ADDRESS ERROR IS ENABLED FOR THAT COMMAND.

                        ;       AOE - BIT 09 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF READ AND WRITE ERRORS ARE ENABLED DURING THE EXECUTION OF THE
                        ;COMMAND.  THE ERRORS ENABLED BY THIS BIT ARE 'TRE', 'DLT', 'NEM',
                        ;'MXF', 'LBT', AND 'AOE'.

                        ;               BIT 08 IS NOT USED.

                        ;       HCE - BIT 07 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF HEADER ERRORS ARE ENABLED DURING THE EXECUTION OF THAT COMMAND.
                        ;HEADER ERRORS INCLUDE 'HCRC', 'HCE', 'FER', AND 'BSE'.

                        ;       ECH - BIT 06 IS SET IN THE ENTRY FOR A GIVEN FUNCTION CODE
                        ;IF DATA FIELD ERRORS ARE ENABLED DURING THE EXECUTION OF THAT
                        ;COMMAND.  THESE ERRORS INCLUDE 'MDPE', 'DCK', AND 'ECH'.

                        ;               BIT 05 IS NOT USED.

                        ;               BIT 04 IS NOT USED.

                        ;               BIT 03 IS NOT USED.

                        ;               BIT 02 IS NOT USED.

                        ;               BIT 01 IS NOT USED.

                        ;       ILF - BIT 00 IS SET IF THE FUNCTION CODE IS ILLEGAL.

063416                  FNCDTB:                                 ;FUNCTION CODE TABLE

063416  020000          .WORD   OPI                             ;NOP
```

```
    58 063420  130001                .WORD   OPI!ATA!ILF!IVC                ;ILLEGAL FUNCTION (2)
    59 063422  132000                .WORD   ATA!OPI!IVC!IAE                 ;SEEK
    60 063424  130000                .WORD   ATA!OPI!IVC                     ;RECALIBRATE
    61 063426  020000                .WORD   OPI                             ;DRIVE CLEAR
    62 063430  030000                .WORD   OPI!IVC                         ;RELEASE
    63 063432  130000                .WORD   OPI!ATA.IVC                     ;OFFSET
    64 063434  130000                .WORD   OPI!ATA.IVC                     ;RETURN TO CENTERLINE
    65 063436  020000                .WORD   OPI                             ;READ IN PRESET
    66 063440  020000                .WORD   OPI                             ;PACK ACKNOWLEDGE
    67 063442  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (24)
    68 063444  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (26)
    69 063446  132000                .WORD   ATA!OPI!IVC!IAE                 ;SEARCH
    70 063450  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (32)
    71 063452  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (34)
    72 063454  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (36)
    73 063456  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (40)
    74 063460  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (42)
    75 063462  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (44)
    76 063464  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (46)
    77 063466  073300                .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK DATA
    78 063470  073300                .WORD   WCE!OPI!IVC!IAE!AOE!HCE!ECH     ;WRITE CHECK HEADER AND DATA
    79 063472  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (54)
    80 063474  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (56)
    81 063476  037200                .WORD   OPI!IVC!WLE!IAE!AOE!HCE         ;WRITE DATA
    82 063500  037000                .WORD   OPI!IVC!WLE!IAE!AOE             ;WRITE HEADER AND DATA
    83 063502  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (64)
    84 063504  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (66)
    85 063506  033300                .WORD   OPI!IVC!IAE!AOE!HCE!ECH         ;READ DATA
    86 063510  033300                .WORD   OPI!IVC!IAE!AOE!HCE!ECH         ;READ HEADER AND DATA
    87 063512  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (74)
    88 063514  130001                .WORD   OPI!ATA!ILF!IVC                 ;ILLEGAL FUNCTION (76)
```

SEQ 0260

```
                                .SBTTL  ATTENTION (ATA) TABLE

                        ATNTBL: .BYTE   1.
                                .BYTE   2.
                                .BYTE   4.
                                .BYTE   8.
                                .BYTE   16.
                                .BYTE   32.
                                .BYTE   64.
                                .BYTE   128.
```

```
                                        .SBTTL  DATA PATTERN TABLE

                                RGDTPT:
                                MIXED:
063526  000000                          .WORD   0.
063530  000001                          .WORD   1.
063532  000003                          .WORD   3.
063534  000007                          .WORD   7.
063536  000017                          .WORD   15.
063540  000037                          .WORD   31.
063542  000077                          .WORD   63.
063544  000177                          .WORD   127.
063546  000377                          .WORD   255.
063550  000777                          .WORD   511.
063552  001777                          .WORD   1023.
063554  003777                          .WORD   2047.
063556  007777                          .WORD   4095.
063560  017777                          .WORD   8191.
063562  037777                          .WORD   16383.
063564  077777                          .WORD   32767.
063566  177777                  ONES:   .WORD   65535.
063570  177777                          .WORD   65535.
063572  077777                          .WORD   32767.
063574  037777                          .WORD   16383.
063576  017777                          .WORD   8191.
063600  007777                          .WORD   4095.
063602  003777                          .WORD   2047.
063604  001777                          .WORD   1023.
063606  000777                          .WORD   511.
063610  000377                          .WORD   255.
063612  000177                          .WORD   127.
063614  000077                          .WORD   63.
063616  000037                          .WORD   31.
063620  000017                          .WORD   15.
063622  000007                          .WORD   7.
063624  000003                          .WORD   3.
063626  000001                          .WORD   1.
063630  000000                  ZEROS:  .WORD   0.
063632  000000                          .WORD   0.
063634  000001                          .WORD   1.
063636  000002                          .WORD   2.
063640  000004                          .WORD   4.
063642  000010                          .WORD   8.
063644  000020                          .WORD   16.
063646  000040                          .WORD   32.
063650  000100                          .WORD   64.
063652  000200                          .WORD   128.
063654  000400                          .WORD   256.
063656  001000                          .WORD   512.
063660  002000                          .WORD   1024.
063662  004000                          .WORD   2048.
063664  010000                          .WORD   4096.
063666  020000                          .WORD   8192.
063670  040000                          .WORD   16384.
063672  100000                          .WORD   32768.
063674  100000                          .WORD   32768.
063676  040000                          .WORD   16384.
```

```
    58 063700  020000                .WORD   8192.
    59 063702  010000                .WORD   4096.
    60 063704  004000                .WORD   2048.
    61 063706  002000                .WORD   1024.
    62 063710  001000                .WORD   512.
    63 063712  000400                .WORD   256.
    64 063714  000200                .WORD   128.
    65 063716  000100                .WORD   64.
    66 063720  000040                .WORD   32.
    67 063722  000020                .WORD   16.
    68 063724  000010                .WORD   8.
    69 063726  000004                .WORD   4.
    70 063730  000002                .WORD   2.
    71 063732  000001                .WORD   1.
    72 063734  000000                .WORD   0.
    73 063736  177777                .WORD   65535.
    74 063740  177776                .WORD   65534.
    75 063742  177774                .WORD   65532.
    76 063744  177770                .WORD   65528.
    77 063746  177760                .WORD   65520.
    78 063750  177740                .WORD   65504.
    79 063752  177700                .WORD   65472.
    80 063754  177600                .WORD   65408.
    81 063756  177400                .WORD   65280.
    82 063760  177000                .WORD   65024.
    83 063762  176000                .WORD   64512.
    84 063764  174000                .WORD   63488.
    85 063766  170000                .WORD   61440.
    86 063770  160000                .WORD   57344.
    87 063772  140000                .WORD   49152.
    88 063774  100000                .WORD   32768.
    89 063776  000000                .WORD   0.
    90 064000  000000                .WORD   0.
    91 064002  100000                .WORD   32768.
    92 064004  140000                .WORD   49152.
    93 064006  160000                .WORD   57344.
    94 064010  170000                .WORD   61440.
    95 064012  174000                .WORD   63488.
    96 064014  176000                .WORD   64512.
    97 064016  177000                .WORD   65024.
    98 064020  177400                .WORD   65280.
    99 064022  177600                .WORD   65408.
   100 064024  177700                .WORD   65472.
   101 064026  177740                .WORD   65504.
   102 064030  177760                .WORD   65520.
   103 064032  177770                .WORD   65528.
   104 064034  177774                .WORD   65532
   105 064036  177776                .WORD   65534.
   106 064040  177777                .WORD   65535.
   107 064042  125252                .WORD   43690.
   108 064044  152525                .WORD   43690./2
   109 064046  125252                .WORD   43690.
   110 064050  177777                .WORD   65535.
   111 064052  177776                .WORD   65534.
   112 064054  177775                .WORD   65533.
   113 064056  177773                .WORD   65531.
   114 064060  177767                .WORD   65527.
```

```
115 064062  177757                         .WORD   65519.
116 064064  177737                         .WORD   65503.
117 064066  177677                         .WORD   65471.
118 064070  177577                         .WORD   65407.
119 064072  177377                         .WORD   65279.
120 064074  176777                         .WORD   65023.
121 064076  175777                         .WORD   64511.
122 064100  173777                         .WORD   63487.
123 064102  167777                         .WORD   61439.
124 064104  157777                         .WORD   57343.
125 064106  137777                         .WORD   49151.
126 064110  077777                         .WORD   32767.
127 064112  077777                         .WORD   32767.
128 064114  137777                         .WORD   49151.
129 064116  157777                         .WORD   57343.
130 064120  167777                         .WORD   61439.
131 064122  173777                         .WORD   63487.
132 064124  175777                         .WORD   64511.
133 064126  176777                         .WORD   65023.
134 064130  177377                         .WORD   65279.
135 064132  177577                         .WORD   65407.
136 064134  177677                         .WORD   65471.
137 064136  177737                         .WORD   65503.
138 064140  177757                         .WORD   65519.
139 064142  177767                         .WORD   65527.
140 064144  177773                         .WORD   65531.
141 064146  177775                         .WORD   65533.
142 064150  177776                         .WORD   65534.
143 064152  177777                         .WORD   65535.
144 064154                      ENRGDT:
145                             .EVEN
```

```
                                        .SBTTL  ERROR MESSAGE TABLE
                                
064154  077300  072206  000000  EMT1:   .WORD   EMS300,EMS1,0
064162  077316  077341  077366  EMT2:   .WORD   EMS301,EMS302,EMS303,EMS1,EMS304
064174  102322  101525  101666          .WORD   EMS511,EMS500,EMS501,EMS502,EMS503,0
064210  077316  077406  077341  EMT3:   .WORD   EMS301,EMS306,EMS302
064216  102322  102043  101666          .WORD   EMS511,EMS505,EMS501,EMS502,0
064230  077300  077341  077422  EMT4:   .WORD   EMS300,EMS302,EMS307,EMS2
064240  102322  101713  101666          .WORD   EMS511,EMS502,EMS501,EMS503,0
064252  077316  077463  077500  EMT5:   .WORD   EMS301,EMS310,EMS311
064260  102322  101713  101666          .WORD   EMS511,EMS502,EMS501,EMS503,EMS504
064272  077544  000000                  .WORD   EMS312,0
064276  077316  077406  077500  EMT6:   .WORD   EMS301,EMS306,EMS311
064304  102322  101713  101666          .WORD   EMS511,EMS502,EMS501,EMS503,EMS504,0
064320  077316  077602  077341  EMT7:   .WORD   EMS301,EMS313,EMS302
064326  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,EMS504,EMS503,0
064342  077710  077731  077633  EMT10:  .WORD   EMS316,EMS317,EMS314
064350  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
064360  077710  077731  077662  EMT11:  .WORD   EMS316,EMS317,EMS315
064366  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
064376  077710  077751  077633  EMT12:  .WORD   EMS316,EMS320,EMS314
064404  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
064414  077710  077751  077662  EMT13:  .WORD   EMS316,EMS320,EMS315
064422  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
064432  077710  077771  077633  EMT14:  .WORD   EMS316,EMS321,EMS314
064440  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
064450  077710  077771  077662  EMT15:  .WORD   EMS316,EMS321,EMS315
064456  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
064466  077710  100011  077633  EMT16:  .WORD   EMS316,EMS322,EMS314
064474  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
064504  077710  100011  077662  EMT17:  .WORD   EMS316,EMS322,EMS315
064512  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
064522  077316  100050  076566  EMT20:  .WORD   EMS301,EMS324,EMS250
064530  102322  102010                  .WORD   EMS511,EMS504
064534  077544  100101  000000          .WORD   EMS312,EMS326,0
064542  077316  100031  076566  EMT21:  .WORD   EMS301,EMS323,EMS250
064550  102322  102010                  .WORD   EMS511,EMS504
064554  077544  100070  000000          .WORD   EMS312,EMS325,0
064562  077316  077602  076566  EMT22:  .WORD   EMS301,EMS313,EMS250
064570  102322  102010  000000          .WORD   EMS511,EMS504,0
064576  077316  100050  076624  EMT23:  .WORD   EMS301,EMS324,EMS251
064604  102322  101666                  .WORD   EMS511,EMS501
064610  077544  100101  000000          .WORD   EMS312,EMS326,0
064616  077316  100031  076624  EMT24:  .WORD   EMS301,EMS323,EMS251
064624  102322  101666                  .WORD   EMS511,EMS501
064630  077544  100070  000000          .WORD   EMS312,EMS325,0
064636  077316  077602  076624  EMT25:  .WORD   EMS301,EMS313,EMS251
064644  102322  101666  000000          .WORD   EMS511,EMS501,0
064652  077316  077406  076670  EMT26:  .WORD   EMS301,EMS306,EMS252,EMS253,EMS327,EMS254
064666  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,EMS502
064676  100117  077633  000000          .WORD   EMS330,EMS314,0
064704  077300  076670          EMT27:  .WORD   EMS300,EMS252
064710  102322  101666  000000          .WORD   EMS511,EMS501,0
064716  077300  076670          EMT30:  .WORD   EMS300,EMS252
064722  102322  101666  102010          .WORD   EMS511,EMS501,EMS504,0
064732  077300  076670          EMT31:  .WORD   EMS300,EMS252
064736  102322  101666  101763          .WORD   EMS511,EMS501,EMS503,0
```

```
 58 064746  077316  100050  076670  EMT32:  .WORD   EMS301,EMS324,EMS252
 59 064754  102322  101666  000000          .WORD   EMS511,EMS501,0
 60 064762  077316  100050  076670  EMT33:  .WORD   EMS301,EMS324,EMS252
 61 064770  102322  101666  102010          .WORD   EMS511,EMS501,EMS504,0
 62 065000  077316  100050  076670  EMT34:  .WORD   EMS301,EMS324,EMS252
 63 065006  102322  101666  101763          .WORD   EMS511,EMS501,EMS503,0
 64 065016  077316  100031  076670  EMT35:  .WORD   EMS301,EMS323,EMS252
 65 065024  102322  101666  000000          .WORD   EMS511,EMS501,0
 66 065032  077316  077602  076670  EMT36:  .WORD   EMS301,EMS313,EMS252
 67 065040  102322  101666  000000          .WORD   EMS511,EMS501,0
 68 065046  077316  100050  077017  EMT37:  .WORD   EMS301,EMS324,EMS255
 69 065054  077544  100101  000000          .WORD   EMS312,EMS326,0
 70 065062  077316  100031  077017  EMT40:  .WORD   EMS301,EMS323,EMS255
 71 065070  102322  102010                  .WORD   EMS511,EMS504
 72 065074  077544  100070  000000          .WORD   EMS312,EMS325,0
 73 065102  077316  077602  077017  EMT41:  .WORD   EMS301,EMS313,EMS255
 74 065110  102322  102010  000000          .WORD   EMS511,EMS504,0
 75 065116  077316  100031  076566  EMT42:  .WORD   EMS301,EMS323,EMS250,EMS327,EMS255
 76 065130  102322  102010  101666          .WORD   EMS511,EMS504,EMS501,EMS503,0
 77 065142  077300  072326          EMT43:  .WORD   EMS300,EMS3
 78 065146  102322  101666  000000          .WORD   EMS511,EMS501,0
 79 065154  100136  072373  076723  EMT44:  .WORD   EMS331,EMS4,EMS253
 80 065162  102322  101666  000000          .WORD   EMS511,EMS501,0
 81 065170  077300  076723          EMT45:  .WORD   EMS300,EMS253
 82 065174  102322  101666  101763          .WORD   EMS511,EMS501,EMS503,0
 83 065204  077300  076723          EMT46:  .WORD   EMS300,EMS253
 84 065210  102322  101666  102010          .WORD   EMS511,EMS501,EMS504,0
 85 065220  077316  100050  076723  EMT47:  .WORD   EMS301,EMS324,EMS253
 86 065226  102322  101666  101763          .WORD   EMS511,EMS501,EMS503
 87 065234  077544  100101  000000          .WORD   EMS312,EMS326,0
 88 065242  077316  100050  076723  EMT50:  .WORD   EMS301,EMS324,EMS253
 89 065250  102322  101666  102010          .WORD   EMS511,EMS501,EMS504
 90 065256  077544  100101  000000          .WORD   EMS312,EMS326,0
 91 065264  077316  100031  076723  EMT51:  .WORD   EMS301,EMS323,EMS253
 92 065272  102322  101666                  .WORD   EMS511,EMS501
 93 065276  077544  100070  000000          .WORD   EMS312,EMS325,0
 94 065304  077316  077602  076723  EMT52:  .WORD   EMS301,EMS313,EMS253
 95 065312  102322  101666  000000          .WORD   EMS511,EMS501,0
 96 065320  100136  072373  077061  EMT53:  .WORD   EMS331,EMS4,EMS256
 97 065326  102322  101666  000000          .WORD   EMS511,EMS501,0
 98 065334  077316  100050  077061  EMT54:  .WORD   EMS301,EMS324,EMS256
 99 065342  102322  101666                  .WORD   EMS511,EMS501
100 065346  077544  100101  000000          .WORD   EMS312,EMS326,0
101 065354  077316  100031  077061  EMT55:  .WORD   EMS301,EMS323,EMS256
102 065362  102322  101666                  .WORD   EMS511,EMS501
103 065366  077544  100070  000000          .WORD   EMS312,EMS325,0
104 065374  077316  077602  077061  EMT56:  .WORD   EMS301,EMS313,EMS256
105 065402  102322  101666  000000          .WORD   EMS511,EMS501,0
106 065410  077111  100166          EMT57:  .WORD   EMS257,EMS332
107 065414  102322  102116  101666          .WORD   EMS511,EMS506,EMS501,0
108 065424  072424  100204          EMT60:  .WORD   EMS5,EMS333
109 065430  102322  102157  101666          .WORD   EMS511,EMS507,EMS501,0
110 065440  077316  100050  077145  EMT61:  .WORD   EMS301,EMS324,EMS260
111 065446  102322  101666                  .WORD   EMS511,EMS501
112 065452  077544  100101  000000          .WORD   EMS312,EMS326,0
113 065460  077316  100031  077145  EMT62:  .WORD   EMS301,EMS323,EMS260
114 065466  102322  101666                  .WORD   EMS511,EMS501
```

```
115 065472  077544  100070  000000          .WORD   EMS312,EMS325,0
116 065500  077316  077602  077145  EMT63:  .WORD   EMS301,EMS313,EMS260
117 065506  102322  101666  000000          .WORD   EMS511,EMS501,0
118 065514  077300  072745          EMT64:  .WORD   EMS300,EMS 2
119 065520  102322  101666  000000          .WORD   EMS511,EMS501,0
120 065526  072745  100230  100326  EMT65:  .WORD   EMS12,EMS335,EMS342
121 065534  102322  101666  000000          .WORD   EMS511,EMS501,0
122 065542  072745  100253  100326  EMT66:  .WORD   EMS12,EMS336,EMS342
123 065550  102322  101666  000000          .WORD   EMS511,EMS501,0
124 065556  077300  072474          EMT67:  .WORD   EMS300,EMS6
125 065562  102322  101666                  .WORD   EMS511,EMS501
126 065566  072540  100204  000000          .WORD   EMS7,EMS333,0
127 065574  077300  072474  100111  EMT70:  .WORD   EMS300,EMS6,EMS327,EMS7
128 065604  102322  101666  102010          .WORD   EMS511,EMS501,EMS504,0
129 065614  072474  100230  100306  EMT71:  .WORD   EMS6,EMS335,EMS340,EMS10,EMS333,EMS342
130 065630  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
131 065640  072474  100253  100306  EMT72:  .WORD   EMS6,EMS336,EMS340,EMS10,EMS334,EMS342
132 065654  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
133 065664  077316  077145  077366  EMT73:  .WORD   EMS301,EMS260,EMS303,EMS11
134 065674  102322  101666  101713          .WORD   EMS511,EMS501,EMS502,0
135 065704  100360  100374  100326  EMT74:  .WORD   EMS343,EMS344,EMS342,0
136 065714  077300  073023          EMT75:  .WORD   EMS300,EMS13
137 065720  102322  101763  000000          .WORD   EMS511,EMS503,0
138 065726  100451  073023  100423  EMT76:  .WORD   EMS346,EMS13,EMS345
139 065734  102322  101763  000000          .WORD   EMS511,EMS503,0
140 065742  100272  073023  100423  EMT77:  .WORD   EMS337,EMS13,EMS345
141 065750  102322  101763  000000          .WORD   EMS511,EMS503,0
142 065756  077316  077602  076756  EMT100: .WORD   EMS301,EMS313,EMS254,EMS347,EMS13
143 065770  102322  101763  000000          .WORD   EMS511,EMS503,0
144 065776  100451  073072  100317  EMT101: .WORD   EMS346,EMS14,EMS341,EMS15
145 066006  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
146 066016  100272  073072  100317  EMT102: .WORD   EMS337,EMS14,EMS341,EMS15
147 066026  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
148 066036  077316  077602  076756  EMT103: .WORD   EMS301,EMS313,EMS254,EMS347,EMS15
149 066050  102322  101763                  .WORD   EMS511,EMS503
150 066054  073072  100166  000000          .WORD   EMS14,EMS332,0
151 066062  100451  073276  100317  EMT104: .WORD   EMS346,EMS17,EMS341,EMS16
152 066072  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
153 066102  100272  073276  100317  EMT105: .WORD   EMS337,EMS17,EMS341,EMS16
154 066112  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
155 066122  077316  077602  076756  EMT106: .WORD   EMS301,EMS313,EMS254,EMS347,EMS16
156 066134  102322  101763                  .WORD   EMS511,EMS503
157 066140  073276  100166  000000          .WORD   EMS17,EMS332,0
158 066146  100451  073337  100317  EMT107: .WORD   EMS346,EMS20,EMS341,EMS21
159 066156  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
160 066166  073337  100515  073403  EMT110: .WORD   EMS20,EMS351,EMS21,EMS350,EMS22,EMS315
161 066202  102322  101666  000000          .WORD   EMS511,EMS501,0
162 066210  073337  100220  100510  EMT111: .WORD   EMS20,EMS334,EMS350,EMS22,EMS333
163 066222  102322  101666  000000          .WORD   EMS511,EMS501,0
164 066230  100272  073337  100317  EMT112: .WORD   EMS337,EMS20,EMS341,EMS21
165 066240  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
166 066250  100272  073337  100317  EMT113: .WORD   EMS337,EMS20,EMS341,EMS21,EMS350,EMS22,EMS334
167 066266  102322  101666  000000          .WORD   EMS511,EMS501,0
168 066274  073337  100533  073403  EMT114: .WORD   EMS20,EMS352,EMS21,EMS350,EMS22,EMS333
169 066310  102322  101666  000000          .WORD   EMS511,EMS501,0
170 066316  077316  077602  076756  EMT115: .WORD   EMS301,EMS313,EMS254,EMS347,EMS21
171 066330  102322  101763                  .WORD   EMS511,EMS503
```

```
172 066334  073337  100166  000000          .WORD   EMS20,EMS332,0
173 066342  100451  073527  100317  EMT116: .WORD   EMS346,EMS23,EMS341,EMS24
174 066352  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
175 066362  100272  073527  100317  EMT117: .WORD   EMS337,EMS23,EMS341,EMS24
176 066372  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
177 066402  077316  077602  076756  EMT120: .WORD   EMS301,EMS313,EMS254,EMS347,EMS24
178 066414  102322  101763                  .WORD   EMS511,EMS503
179 066420  073527  100166  000000          .WORD   EMS23,EMS332,0
180 066426  100451  073664  100317  EMT121: .WORD   EMS346,EMS25,EMS341,EMS26
181 066436  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
182 066446  100272  073664  100317  EMT122: .WORD   EMS337,EMS25,EMS341,EMS26
183 066456  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
184 066466  077316  077602  076756  EMT123: .WORD   EMS301,EMS313,EMS254,EMS347,EMS26
185 066500  102322  101763                  .WORD   EMS511,EMS503
186 066504  073664  100166  000000          .WORD   EMS25,EMS332,0
187 066512  077300  074021  077422  EMT124: .WORD   EMS300,EMS27,EMS307,EMS2
188 066522  102322  101763  000000          .WORD   EMS511,EMS503,0
189 066530  100136  074021  100547  EMT125: .WORD   EMS331,EMS27,EMS353
190 066536  102322  101763                  .WORD   EMS511,EMS503
191 066542  074065  077662  000000          .WORD   EMS30,EMS315,0
192 066550  100272  074021  100317  EMT126: .WORD   EMS337,EMS27,EMS341,EMS30
193 066560  102322  101763  000000          .WORD   EMS511,EMS503,0
194 066566  077316  077602  076756  EMT127: .WORD   EMS301,EMS313,EMS254,EMS347,EMS30
195 066600  102322  101763                  .WORD   EMS511,EMS503
196 066604  074021  100166  000000          .WORD   EMS27,EMS332,0
197 066612  074145  100573  076566  EMT130: .WORD   EMS31,EMS354,EMS250
198 066620  102322  102010  101763          .WORD   EMS511,EMS504,EMS503,0
199 066630  074216  100573  076566  EMT131: .WORD   EMS32,EMS354,EMS250
200 066636  102322  102010  101763          .WORD   EMS511,EMS504,EMS503,0
201 066646  100626  074276  076566  EMT132: .WORD   EMS355,EMS33,EMS250,EMS341,EMS30
202 066660  102322  102010  000000          .WORD   EMS511,EMS504,0
203 066666  100626  074326  076566  EMT133: .WORD   EMS355,EMS34,EMS250,EMS341,EMS30
204 066700  102322  102010  000000          .WORD   EMS511,EMS504,0
205 066706  100136  072373  077017  EMT134: .WORD   EMS331,EMS4,EMS255
206 066714  102322  102010  000000          .WORD   EMS511,EMS504,0
207 066722  074355  100670  100712  EMT135: .WORD   EMS35,EMS357,EMS360,EMS15
208 066732  102322  101666  000000          .WORD   EMS511,EMS501,0
209 066740  077176  100766          EMT136: .WORD   EMS261,EMS362
210 066744  102322  101763  000000          .WORD   EMS511,EMS503,0
211 066752  077300  074417  077422  EMT137: .WORD   EMS300,EMS36,EMS307,EMS2
212 066762  102322  101666  000000          .WORD   EMS511,EMS501,0
213 066770  100626  074447  077017  EMT140: .WORD   EMS355,EMS37,EMS255,EMS341,EMS30
214 067002  102322  102010  000000          .WORD   EMS511,EMS504,0
215 067010  100451  074501  100423  EMT141: .WORD   EMS346,EMS40,EMS345
216 067016  102322  102010  000000          .WORD   EMS511,EMS504,0
217 067024  100272  074501  100317  EMT142: .WORD   EMS337,EMS40,EMS341,EMS30
218 067034  102322  102010  000000          .WORD   EMS511,EMS504,0
219 067042  101007  077463  074557  EMT143: .WORD   EMS363,EMS310,EMS41
220 067050  102322  101666  000000          .WORD   EMS511,EMS501,0
221 067056  100272  074557  100317  EMT144: .WORD   EMS337,EMS41,EMS341,EMS252,EMS327,EMS253
222 067072  102322  101666  000000          .WORD   EMS511,EMS501,0
223 067100  074557  100573  101024  EMT145: .WORD   EMS41,EMS354,EMS364,EMS252,EMS365,EMS253
224 067114  102322  101666  000000          .WORD   EMS511,EMS501,0
225 067122  077316  077406  074417  EMT146: .WORD   EMS301,EMS306,EMS36
226 067130  102322  101666  101763          .WORD   EMS511,EMS501,EMS503,0
227 067140  101052  074625          EMT147: .WORD   EMS366,EMS42
228 067144  102322  101763  000000          .WORD   EMS511,EMS503,0
```

```
229 067152  101075  100547  101045  EMT150: .WORD   EMS367,EMS353,EMS365,EMS42,EMS354,EMS3
230 067166  102322  101763  000000          .WORD   EMS511,EMS503,0
231 067174  100272  074417          EMT151: .WORD   EMS337,EMS36
232 067200  102322  101666  000000          .WORD   EMS511,EMS501,0
233 067206  074707  100573  074417  EMT152: .WORD   EMS43,EMS354,EMS56
234 067214  102322  101666  000000          .WORD   EMS511,EMS501,0
235 067222  101075  100547  101045  EMT153: .WORD   EMS367,EMS353,EMS365,EMS36,EMS370
236 067234  102322  101763  000000          .WORD   EMS511,EMS503,0
237 067242  101075  100547  101045  EMT154: .WORD   EMS367,EMS353,EMS365,EMS36,EMS371
238 067254  102322  101763  000000          .WORD   EMS511,EMS503,0
239 067262  077300  074760  077422  EMT155: .WORD   EMS300,EMS44,EMS307,EMS2
240 067272  102322  101763  000000          .WORD   EMS511,EMS503,0
241 067300  101075  100547  101045  EMT156: .WORD   EMS367,EMS353,EMS365,EMS44,EMS354,EMS3
242 067314  102322  101763  000000          .WORD   EMS511,EMS503,0
243 067322  077300  075021  077422  EMT157: .WORD   EMS300,EMS45,EMS307,EMS2
244 067332  102322  101763  000000          .WORD   EMS511,EMS503,0
245 067340  101075  100547  101045  EMT160: .WORD   EMS367,EMS353,EMS365,EMS45,EMS354,EMS3
246 067354  102322  101763  101666          .WORD   EMS511,EMS503,EMS501
247 067362  074355  100204  000000          .WORD   EMS35,EMS333,0
248 067370  077300  075076  077422  EMT161: .WORD   EMS300,EMS46,EMS307,EMS2
249 067400  102322  101763  000000          .WORD   EMS511,EMS503,0
250 067406  100272  075076  100547  EMT162: .WORD   EMS337,EMS46,EMS353
251 067414  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
252 067424  074355  100230  100272  EMT163: .WORD   EMS35,EMS335,EMS337,EMS41,EMS334,EMS372
253 067440  102322  101666  000000          .WORD   EMS511,EMS501,0
254 067446  075160  100230  100272  EMT164: .WORD   EMS47,EMS335,EMS337,EMS41,EMS335,EMS372
255 067462  102322  101666  000000          .WORD   EMS511,EMS501,0
256 067470  100272  074355  100111  EMT165: .WORD   EMS337,EMS35,EMS327,EMS47
257 067500  102322  101666                  .WORD   EMS511,EMS501
258 067504  074557  100204  101211          .WORD   EMS41,EMS333,EMS372,0
259 067514  077300  075160  077422  EMT166: .WORD   EMS300,EMS47,EMS307,EMS2
260 067524  102322  101666  101763          .WORD   EMS511,EMS501,EMS503,0
261 067534  075220  100230  100306  EMT167: .WORD   EMS50,EMS335,EMS340,EMS36,EMS333
262 067546  102322  101666  101763          .WORD   EMS511,EMS501,EMS503,0
263 067556  100272  074355          EMT170: .WORD   EMS337,EMS35
264 067562  102322  101666  000000          .WORD   EMS511,EMS501,0
265 067570  075220  074326  072326  EMT171: .WORD   EMS50,EMS34,EMS3
266 067576  102322  101666  000000          .WORD   EMS511,EMS501,0
267 067604  077316  077406  075267  EMT172: .WORD   EMS301,EMS306,EMS51
268 067612  102322  101666  102010          .WORD   EMS511,EMS501,EMS504,0
269 067622  101075  100547  101045  EMT173: .WORD   EMS367,EMS353,EMS365,EMS47,EMS354,EMS3
270 067636  102322  101666  000000          .WORD   EMS511,EMS501,0
271 067644  077300  076566  100111  EMT174: .WORD   EMS300,EMS250,EMS327,EMS255,EMS327,EMS256
272 067660  100317  102421                  .WORD   EMS341,EMS600
273 067664  102322  101666  000000          .WORD   EMS511,EMS501,0
274 067672  077300  077061  100317  EMT175: .WORD   EMS300,EMS256,EMS341,EMS600
275 067702  102322  101666  000000          .WORD   EMS511,EMS501,0
276 067710  077300  076566  100317  EMT176: .WORD   EMS300,EMS250,EMS341,EMS600
277 067720  102322  102010  000000          .WORD   EMS511,EMS504,0
278 067726                          EMT177:
279 067726  100272  075336  100317  EMT200: .WORD   EMS337,EMS52,EMS341,EMS601
280 067736  102322  101666  000000          .WORD   EMS511,EMS501,0
281 067744  100451  075336  100317  EMT201: .WORD   EMS346,EMS52,EMS341,EMS602
282 067754  102322  101666  000000          .WORD   EMS511,EMS501,0
283 067762  100451  075267  100317  EMT202: .WORD   EMS346,EMS51,EMS341,EMS602
284 067772  102322  101666  000000          .WORD   EMS511,EMS501,0
285 070000  100451  075336  100423  EMT203: .WORD   EMS346,EMS52,EMS345,EMS373,EMS255
```

```
286 070012  102322  102010  101666          .WORD   EMS511,EMS504,EMS501,0
287 070022  100451  075336  100317  EMT204: .WORD   EMS346,EMS52,EMS341,EMS27
288 070032  102322  102010  101666          .WORD   EMS511,EMS504,EMS501,0
289 070042  075377  100573  072326  EMT205: .WORD   EMS53,EMS354,EMS3
290 070050  102322  101763  101713          .WORD   EMS511,EMS503,EMS502,EMS510,0
291 070062  100272  075440  101247  EMT206: .WORD   EMS337,EMS54,EMS374,EMS250,EMS327,EMS255
292 070076  100111  072326                  .WORD   EMS327,EMS3
293 070102  102322  102010  101666          .WORD   EMS511,EMS504,EMS501,0
294 070112  075440  100573  072326  EMT207: .WORD   EMS54,EMS354,EMS3
295 070120  102322  101666  000000          .WORD   EMS511,EMS501,0
296 070126  075440  100573  101024  EMT210: .WORD   EMS54,EMS354,EMS364,EMS250
297 070136  102322  102010  000000          .WORD   EMS511,EMS504,0
298 070144  075440  100573  101024  EMT211: .WORD   EMS54,EMS354,EMS364,EMS255
299 070154  102322  102010  000000          .WORD   EMS511,EMS504,0
300 070162  100272  075515          EMT212: .WORD   EMS337,EMS55
301 070166  102322  102010  000000          .WORD   EMS511,EMS504,0
302 070174  075573  100220  100423  EMT213: .WORD   EMS56,EMS334,EMS345,EMS373,EMS262,EMS327,EMS251
303 070212  102322  101666                  .WORD   EMS511,EMS501
304 070216  075573  100230  000000          .WORD   EMS56,EMS335,0
305 070224  100272  075573          EMT214: .WORD   EMS337,EMS56
306 070230  102322  101666  000000          .WORD   EMS511,EMS501,0
307 070236  075666  100204  100510  EMT215: .WORD   EMS57,EMS333,EMS350,EMS60,EMS334
308 070250  102322  101666  000000          .WORD   EMS511,EMS501,0
309 070256  101007  077406  072424  EMT216: .WORD   EMS363,EMS306,EMS5
310 070264  102322  101666  000000          .WORD   EMS511,EMS501,0
311 070272  101007  077406  076024  EMT217: .WORD   EMS363,EMS306,EMS61
312 070300  102322  101666  000000          .WORD   EMS511,EMS501,0
313 070306  076077  100204  101300  EMT220: .WORD   EMS62,EMS333,EMS375,EMS251
314 070316  102322  101666  000000          .WORD   EMS511,EMS501,0
315 070324  076077  100204  101314  EMT221: .WORD   EMS62,EMS333,EMS376,EMS262
316 070334  102322  101666  000000          .WORD   EMS511,EMS501,0
317 070342  076077  100204  101314  EMT222: .WORD   EMS62,EMS333,EMS376,EMS250
318 070352  102322  101666  000000          .WORD   EMS511,EMS501,0
319 070360  100451  076077  100317  EMT223: .WORD   EMS346,EMS62,EMS341,EMS603
320 070370  102322  101666  000000          .WORD   EMS511,EMS501,0
321 070376  100451  076160  101314  EMT224: .WORD   EMS346,EMS63,EMS376,EMS262
322 070406  102322  101666  101763          .WORD   EMS511,EMS501,EMS503,0
323 070416  076160  100204  100510  EMT225: .WORD   EMS63,EMS333,EMS350,EMS363,EMS310,EMS262
324 070432  102322  101666  000000          .WORD   EMS511,EMS501,0
325 070440  076160  100670  074417  EMT226: .WORD   EMS63,EMS357,EMS36,EMS372
326 070450  102322  101666  000000          .WORD   EMS511,EMS501,0
327 070456  076160  100650  100712  EMT227: .WORD   EMS63,EMS356,EMS360,EMS15
328 070466  102322  101666  000000          .WORD   EMS511,EMS501,0
329 070474  076160  100650  100740  EMT230: .WORD   EMS63,EMS356,EMS361,EMS15
330 070504  102322  101666  000000          .WORD   EMS511,EMS501,0
331 070512  076160  100650  074557  EMT231: .WORD   EMS63,EMS356,EMS41
332 070520  102322  101666  000000          .WORD   EMS511,EMS501,0
333 070526  074557  101330  100326  EMT232: .WORD   EMS41,EMS377,EMS342,EMS365,EMS63,EMS332
334 070542  102322  101666  000000          .WORD   EMS511,EMS501,0
335 070550  101075  100547  101045  EMT233: .WORD   EMS367,EMS353,EMS365,EMS63,EMS401
336 070562  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
337 070572  073215  101330  101211  EMT234: .WORD   EMS16,EMS377,EMS372,EMS365,EMS64,EMS354,EMS3
338 070610  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
339 070620  077300  076270  077422  EMT235: .WORD   EMS300,EMS65,EMS307,EMS2
340 070630  102322  101713  101763          .WORD   EMS511,EMS502,EMS503,0
341 070640  075573  101330  101374  EMT236: .WORD   EMS56,EMS377,EMS376,EMS252,EMS372,EMS350
342 070654  076270  101356                  .WORD   EMS65,EMS401
```

```
343 070660  102322  101713  101763          .WORD   EMS511,EMS502,EMS503,EMS501,0
344 070672  077300  076331  077422  EMT237: .WORD   EMS300,EMS66,EMS307,EMS2
345 070702  102322  101666  101763          .WORD   EMS511,EMS501,EMS503,0
346 070712  073137  101341  101211  EMT240: .WORD   EMS15,EMS400,EMS372,EMS350,EMS66,EMS401
347 070726  102322  101763  101666          .WORD   EMS511,EMS503,EMS501,0
348 070736  076331  100253  100306  EMT241: .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS604
349 070754  102322  101763  000000          .WORD   EMS511,EMS503,0
350 070762  101402  102553  101373  EMT242: .WORD   EMS403,EMS604,EMS402,EMS21,EMS377
351 070774  102322  101763                  .WORD   EMS511,EMS503
352 071000  074417  100230  000000          .WORD   EMS36,EMS335,0
353 071006  076331  100253  100306  EMT243: .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS604
354 071024  102322  101763  000000          .WORD   EMS511,EMS503,0
355 071032  076160  101356  101462  EMT244: .WORD   EMS63,EMS401,EMS405,EMS604
356 071042  102322  101763  000000          .WORD   EMS511,EMS503,0
357 071050  074417  101145  101462  EMT245: .WORD   EMS36,EMS370,EMS405,EMS604
358 071060  102322  101763  000000          .WORD   EMS511,EMS503,0
359 071066  101402  102553  101373  EMT246: .WORD   EMS403,EMS604,EMS402,EMS24,EMS377
360 071100  102322  101763                  .WORD   EMS511,EMS503
361 071104  074417  100230  000000          .WORD   EMS36,EMS335,0
362 071112  076406  100166  101462  EMT247: .WORD   EMS67,EMS332,EMS405,EMS604
363 071122  102322  101763  000000          .WORD   EMS511,EMS503,0
364 071130  076331  100253  100306  EMT250: .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS605
365 071146  102322  101763  000000          .WORD   EMS511,EMS503,0
366 071154  101402  102600  101373  EMT251: .WORD   EMS403,EMS605,EMS402,EMS21,EMS377
367 071166  102322  101763                  .WORD   EMS511,EMS503
368 071172  074417  100230  000000          .WORD   EMS36,EMS335,0
369 071200  076331  100253  100306  EMT252: .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS605
370 071216  102322  101763  000000          .WORD   EMS511,EMS503,0
371 071224  076160  101356  101462  EMT253: .WORD   EMS63,EMS401,EMS405,EMS605
372 071234  102322  101763  000000          .WORD   EMS511,EMS503,0
373 071242  074417  101145  101462  EMT254: .WORD   EMS36,EMS370,EMS405,EMS605
374 071252  102322  101763  000000          .WORD   EMS511,EMS503,0
375 071260  101402  102600  101373  EMT255: .WORD   EMS403,EMS605,EMS402,EMS24,EMS377
376 071272  102322  101763                  .WORD   EMS511,EMS503
377 071276  074417  100230  000000          .WORD   EMS36,EMS335,0
378 071304  076406  100166  101462  EMT256: .WORD   EMS67,EMS332,EMS405,EMS605
379 071314  102322  101763  000000          .WORD   EMS511,EMS503,0
380 071322  076331  100253  100306  EMT257: .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS606
381 071340  102322  101763  000000          .WORD   EMS511,EMS503,0
382 071346  101402  102616  101373  EMT260: .WORD   EMS403,EMS606,EMS402,EMS21,EMS377
383 071360  102322  101763                  .WORD   EMS511,EMS503
384 071364  074417  100230  000000          .WORD   EMS36,EMS335,0
385 071372  076331  100253  100306  EMT261: .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS606
386 071410  102322  101763  000000          .WORD   EMS511,EMS503,0
387 071416  076160  101356  101462  EMT262: .WORD   EMS63,EMS401,EMS405,EMS606
388 071426  102322  101763  000000          .WORD   EMS511,EMS503,0
389 071434  074417  101145  101462  EMT263: .WORD   EMS36,EMS370,EMS405,EMS606
390 071444  102322  101763  000000          .WORD   EMS511,EMS503,0
391 071452  101402  102616  101373  EMT264: .WORD   EMS403,EMS606,EMS402,EMS24,EMS377
392 071464  102322  101763                  .WORD   EMS511,EMS503
393 071470  074417  100230  000000          .WORD   EMS36,EMS335,0
394 071476  076520  101356  101462  EMT265: .WORD   EMS70,EMS401,EMS405,EMS606
395 071506  102322  101763  000000          .WORD   EMS511,EMS503,0
396 071514  076406  100166  101462  EMT266: .WORD   EMS67,EMS332,EMS405,EMS606
397 071524  102322  101763  000000          .WORD   EMS511,EMS503,0
398 071532  076331  100650  101505  EMT267: .WORD   EMS66,EMS356,EMS407
399 071540  102322  101763  000000          .WORD   EMS511,EMS503,0
```

```
400 071546  076331  100253  100306  EMT270: .WORD   EMS66,EMS336,EMS340,EMS15,EMS406,EMS405,EMS607
401 071564  102322  101763  000000          .WORD   EMS511,EMS503,0
402 071572  101402  102636  101373  EMT271: .WORD   EMS403,EMS607,EMS402,EMS21,EMS377
403 071604  102322  101763                  .WORD   EMS511,EMS503
404 071610  074021  100253  000000          .WORD   EMS27,EMS336,0
405 071616  074021  101145  101462  EMT272: .WORD   EMS27,EMS370,EMS405,EMS607
406 071626  102322  101763  000000          .WORD   EMS511,EMS503,0
407 071634  074021  101162  101462  EMT273: .WORD   EMS27,EMS371,EMS405,EMS607
408 071644  102322  101763  000000          .WORD   EMS511,EMS503,0
409 071652  074417  101443  101462  EMT274: .WORD   EMS36,EMS404,EMS405,EMS607
410 071662  102322  101763  000000          .WORD   EMS511,EMS503,0
411 071670  075377  101356  101462  EMT275: .WORD   EMS53,EMS401,EMS405,EMS607
412 071700  102322  101763  101713          .WORD   EMS511,EMS503,EMS502,0
413 071710  076406  100166  101462  EMT276: .WORD   EMS67,EMS332,EMS405,EMS607
414 071720  102322  101763  000000          .WORD   EMS511,EMS503,0
415 071726  076331  100253  100306  EMT277: .WORD   EMS66,EMS336,EMS340,EMS26,EMS404,EMS405,EMS607
416 071744  102322  101763  000000          .WORD   EMS511,EMS503,0
417 071752  101402  102636  101373  EMT300: .WORD   EMS403,EMS607,EMS402,EMS24,EMS377
418 071764  102322  101763                  .WORD   EMS511,EMS503
419 071770  074021  100253  000000          .WORD   EMS27,EMS336,0
420 071776  076520  101356  101462  EMT301: .WORD   EMS70,EMS401,EMS405,EMS607
421 072006  102322  101763  000000          .WORD   EMS511,EMS503,0
422 072014  077300  077017  100317  EMT302: .WORD   EMS300,EMS255,EMS341,EMS600
423 072024  102322  102010  000000          .WORD   EMS511,EMS504,0
```

ERROR MESSAGE TABLE

```
 1 072032  102654  000000          EHT1:    .WORD   EH1,0
 2 072036  102732  000000          EHT2:    .WORD   EH2,0
 3 072042  102747  000000          EHT5:    .WORD   EH5,0
 4 072046  103005  000000          EHT7:    .WORD   EH7,0
 5 072052  103024  000000          EHT57:   .WORD   EH57,0
 6 072056  103062  000000          EHT65:   .WORD   EH65,0
 7 072062  103136  000000          EHT71:   .WORD   EH71,0
 8 072066  102740  000000          EHT74:   .WORD   EH3,0
 9 072072  103213  000000          EHT115:  .WORD   EH115,0
10 072076  103310  000000          EHT130:  .WORD   EH130,0
11 072102  103426  000000          EHT132:  .WORD   EH132,0
12 072106  103524  000000          EHT142:  .WORD   EH142,0
13 072112  103622  000000          EHT145:  .WORD   EH145,0
14 072116  103737  000000          EHT150:  .WORD   EH150,0
15 072122  104035  000000          EHT213:  .WORD   EH213,0
16 072126  104132  000000          EHT220:  .WORD   EH220,0
```

ERROR MESSAGE TABLE

```
    1 072132  104170                EDT1:    .WORD   ED1
    2 072134  104200                EDT2:    .WORD   ED2
    3 072136  104204                EDT5:    .WORD   ED5
    4 072140  104212                EDT57:   .WORD   ED57
    5 072142  104220                EDT65:   .WORD   ED65
    6 072144  104230                EDT71:   .WORD   ED71
    7 072146  104200                EDT74:   .WORD   ED2
    8 072150  104240                EDT115:  .WORD   ED115
    9 072152  104252                EDT130:  .WORD   ED130
   10 072154  104240                EDT132:  .WORD   ED115
   11 072156  104266                EDT220:  .WORD   ED220
```

ERROR MESSAGE TABLE

```
 1 072160  104272          EFT1:    .WORD   EF1
 2 072162  104275          EFT2:    .WORD   EF2
 3 072164  104276          EFT5:    .WORD   EF5
 4 072166  104300          EFT57:   .WORD   EF57
 5 072170  104272          EFT65:   .WORD   EF1
 6 072172  104272          EFT71:   .WORD   EF1
 7 072174  104275          EFT74:   .WORD   EF2
 8 072176  104300          EFT115:  .WORD   EF57
 9 072200  104304          EFT130:  .WORD   EF130
10 072202  104300          EFT132:  .WORD   EF57
11 072204  104276          EFT220:  .WORD   EF5
```

```
                                        .SBTTL  ERROR MESSAGE STRINGS
                                
072206  116  117  116  EMS1:   .ASCIZ  @NONEXISTENT DEVICE 'NED' (RMCS2,BIT 12) @
072257  103  117  116  EMS2:   .ASCIZ  @CONTROLLER CLEAR 'CLR' (RMCS2,BIT 05) @
072326  106  125  116  EMS3:   .ASCIZ  @FUNCTION CODE (RMCS1, BITS 01 - 05) @
072373  125  116  125  EMS4:   .ASCIZ  @UNUSED BIT POSITIONS OF @
072424  104  105  126  EMS5:   .ASCIZ  @DEVICE AVAILABLE 'DVA' (RMCS1, BIT 11) @
072474  120  101  122  EMS6:   .ASCIZ  @PARTIY ERROR 'PAR' (RMER1, BIT 03) @
072540  104  101  124  EMS7:   .ASCIZ  @DATA PARITY ERROR 'DPE' (RMER2, BIT 03) @
072611  120  101  122  EMS10:  .ASCIZ  @PARITY TEST 'PAT' (RMCS2, BIT 04) @
072654  115  101  123  EMS11:  .ASCII  @MASSBUS CONTROL BUS PARITY ERROR 'MCPE' @
072724  050  122  115          .ASCIZ  @(RMCS1, BIT 13) @
072745  111  114  114  EMS12:  .ASCIZ  @ILLEGAL REGISTER ERROR 'ILR' (RMER1, BIT 01) @
073023  104  111  101  EMS13:  .ASCIZ  @DIAGNOSTIC MODE 'DMD' (RMMR1, BIT 00) @
073072  115  105  104  EMS14:  .ASCIZ  @MEDIUM ON LINE 'MOL' (RMDS, BIT 12) @
073137  115  101  111  EMS15:  .ASCIZ  @MAINTENANCE UNIT READY 'MUR' (RMMR1, BIT 09) @
073215  115  101  111  EMS16:  .ASCIZ  @MAINTENANCE WRITE PROTECT 'MWP' (RMMR1, BIT 03) @
073276  127  122  111  EMS17:  .ASCIZ  @WRITE LOCK 'WRL' (RMDS, BIT 11) @
073337  104  105  126  EMS20:  .ASCIZ  @DEVICE CHECK 'DVC' (RMER2, BIT 07) @
073403  115  101  111  EMS21:  .ASCIZ  @MAINTENANCE DRIVE FAULT 'MDF' (RMMR1, BIT 06) @
073462  125  116  123  EMS22:  .ASCIZ  @UNSAFE STATUS 'UNS' (RMER1, BIT 14) @
073527  123  105  105  EMS23:  .ASCIZ  @SEEK INCOMPLETE STATUS 'SKI' (RMER2, BIT 14) @
073605  115  101  111  EMS24:  .ASCIZ  @MAINTENANCE SEEK ERROR 'MSER' (RMMR1, BIT 07) @
073664  120  117  123  EMS25:  .ASCIZ  @POSITIONING IN PROGRESS 'PIP' (RMDS, BIT 13) @
073742  115  101  111  EMS26:  .ASCIZ  @MAINTENANCE ON CYLINDER 'MOC' (RMMR1, BIT 08) @
074021  105  116  104  EMS27:  .ASCIZ  @END OF BLOCK 'EBL' (RMMR1, BIT 13) @
074065  104  111  101  EMS30:  .ASCIZ  @DIAGNOSTIC END OF BLOCK 'DEBL' (RMMR1, BIT 13) @
074145  114  101  123  EMS31:  .ASCIZ  @LAST SECTOR STATUS 'LS' (RMMR1, BIT 02) @
074216  114  101  123  EMS32:  .ASCIZ  @LAST SECTOR/TRACK STATUS 'LST' (RMMR1, BIT 01) @
074276  123  105  103  EMS33:  .ASCIZ  @SECTOR ADDRESS BITS OF @
074326  124  122  101  EMS34:  .ASCIZ  @TRACK ADDRESS BITS OF @
074355  126  117  114  EMS35:  .ASCIZ  @VOLUME VALID 'VV' (RMDS, BIT 06) @
074417  107  117  040  EMS36:  .ASCIZ  @GO BIT (RMCS1, BIT 00) @
074447  103  131  114  EMS37:  .ASCIZ  @CYLINDER ADDRESS BITS OF @
074501  114  101  123  EMS40:  .ASCIZ  @LAST BLOCK TRANSFERRED, 'LBT' (RMDS, BIT 10) @
074557  103  117  115  EMS41:  .ASCIZ  @COMPOSITE ERROR 'ERR' (RMDS, BIT 14) @
074625  103  117  115  EMS42:  .ASCIZ  @COMMAND SEQUENCER TEST BIT 'TST' (RMMR2, BIT 12) @
074707  104  122  111  EMS43:  .ASCIZ  @DRIVE READY STATUS 'DRY' (RMDS, BIT 07) @
074760  103  117  116  EMS44:  .ASCIZ  @CONTINUE 'CONT' (RMMR1, BIT 06) @
075021  111  116  126  EMS45:  .ASCIZ  @INVALID COMMAND ERROR 'IVC' (RMER2, BIT 12) @
075076  114  117  123  EMS46:  .ASCIZ  @LOSS OF SYSTEM CLOCK ERROR 'LSC' (RMER2, BIT 11) @
075160  117  103  103  EMS47:  .ASCIZ  @OCCUPIED 'OCC' (RMMR1, BIT 15) @
075220  111  114  114  EMS50:  .ASCIZ  @ILLEGAL FUNCTION 'ILF' (RMER1, BIT 0) @
075267  117  106  106  EMS51:  .ASCIZ  @OFFSET DIRECTION 'OFD' (RMOF, BIT 07) @
075336  117  106  106  EMS52:  .ASCIZ  @OFFSET MODE 'OM' (RMDS, BIT 00) @
075377  122  125  116  EMS53:  .ASCIZ  @RUN AND GO 'RG' (RMMR1, BIT 14) @
075440  111  116  126  EMS54:  .ASCIZ  @INVALID ADDRESS ERROR 'IAE' (RMER1, BIT 10) @
075515  101  104  104  EMS55:  .ASCIZ  @ADDRESS OVERFLOW ERROR 'AOE' (RMER1, BIT 09) @
075573  122  105  107  EMS56:  .ASCII  @REGISTER MODIFICATION REFUSED ERROR @
075637  042  122  115          .ASCIZ  @'RMR' (RMER1, BIT 02) @
075666  104  122  111  EMS57:  .ASCIZ  @DRIVE REQUEST REQUIRED STATUS 'DRQ' (RMDT, BIT 11) @
075752  120  122  117  EMS60:  .ASCIZ  @PROGRAMMABLE STATUS 'PGM' (RMDS, BIT 09) @
076024  104  122  111  EMS61:  .ASCIZ  @DRIVE PRESENT STATUS 'DPR' (RMDS, BIT 08) @
076077  120  117  122  EMS62:  .ASCIZ  @PORT REQUEST FLOP 'RQA,RQB' (RMMR2, BITS 15,14) @
076160  101  124  124  EMS63:  .ASCIZ  @ATTENTION 'ATA' (RMDS, BIT 15) @
076220  127  122  111  EMS64:  .ASCIZ  @WRITE LOCK ERROR 'WLE' (RMER1, BIT 11) @
076270  105  130  103  EMS65:  .ASCIZ  @EXCEPTION 'REX' (RMMR1, BIT 12) @
```

```
076331  111  116  126  EMS66:   .ASCIZ  @INVALID COMMAND ERROR ''IVC'' (RMER2, BIT 12) @
076406  124  101  107  EMS67:   .ASCIZ  @TAG BUS (RMMR2, BITS 00-09) OR TAG CONTROL @
076462  114  111  116           .ASCIZ  @LINES (RMMR2, BITS 10,11,13) @
076520  123  105  101  EMS70:   .ASCIZ  @SEARCH ENABLE ''ESRC'' (RMMR1, BIT 11) @

076566  104  111  123  EMS250:  .ASCIZ  @DISK ADDRESS REGISTER (RMDA) @
076624  103  117  116  EMS251:  .ASCIZ  @CONTROL STATUS REGISTER #1 (RMCS1) @
076670  105  122  122  EMS252:  .ASCIZ  @ERROR REGISTER #1 (RMER1) @
076723  105  122  122  EMS253:  .ASCIZ  @ERROR REGISTER #2 (RMER2) @
076756  115  101  111  EMS254:  .ASCIZ  @MAINTENANCE REGISTER #1 (RMMR1) @
077017  104  105  123  EMS255:  .ASCIZ  @DESIRED CYLINDER REGISTER (RMDC) @
077061  117  106  106  EMS256:  .ASCIZ  @OFFSET REGISTER (RMOF) @
077111  104  122  111  EMS257:  .ASCIZ  @DRIVE TYPE REGISTER (RMDT) @
077145  110  117  114  EMS260:  .ASCIZ  @HOLDING REGISTER (RMHR) @
077176  123  105  122  EMS261:  .ASCIZ  @SERIAL NUMBER REGISTER (RMSN) @
077235  101  124  124  EMS262:  .ASCIZ  @ATTENTION SUMMARY REGISTER (RMAS) @

077300  103  101  116  EMS300:  .ASCIZ  @CANNOT CLEAR @
077316  103  101  116  EMS301:  .ASCIZ  @CANNOT WRITE/READ @
077341  101  116  131  EMS302:  .ASCIZ  @ANY DEVICE REGISTER @
077366  127  111  124  EMS303:  .ASCIZ  @WITHOUT @
077377  105  122  122  EMS304:  .ASCIZ  @ERROR @
077406  101  040  117  EMS306:  .ASCIZ  @A ONE FROM @
077422  125  123  111  EMS307:  .ASCIZ  @USING MASSBUS INITIALIZE, I.E., @
077463  101  040  132  EMS310:  .ASCIZ  @A ZERO FROM @
077500  105  126  105  EMS311:  .ASCIZ  @EVERY DEVICE REGISTER BIT POSITION @
077544  124  110  105  EMS312:  .ASCIZ  @THE FOLLOWING BITS ARE STUCK @
077602  101  040  123  EMS313:  .ASCIZ  @A SHIFTING ONE BIT FROM @
077633  101  120  120  EMS314:  .ASCIZ  @APPEARS STUCK AT ZERO @
077662  101  120  120  EMS315:  .ASCIZ  @APPEARS STUCK AT ONE @
077710  122  105  107  EMS316:  .ASCIZ  @REGISTER SELECT @
077731  061  040  050  EMS317:  .ASCIZ  @1 (1,2,4,8,16) @
077751  062  040  050  EMS320:  .ASCIZ  @2 (1,2,4,8,16) @
077771  064  040  050  EMS321:  .ASCIZ  @4 (1,2,4,8,16) @
100011  070  040  050  EMS322:  .ASCIZ  @8 (1,2,4,8,16) @
100031  101  114  114  EMS323:  .ASCIZ  @ALL ONES FROM @
100050  101  114  114  EMS324:  .ASCIZ  @ALL ZEROS FROM @
100070  101  124  040  EMS325:  .ASCIZ  @AT ZERO @
100101  101  124  040  EMS326:  .ASCIZ  @AT ONE @
100111  054  040  117  EMS327:  .ASCIZ  @, OR @
100117  015  012  103  EMS330:  .ASCIZ  <CR><LF>@CS MBA CLRL @
100136  103  101  116  EMS331:  .ASCIZ  @CANNOT READ ZEROS FROM @
100166  111  123  040  EMS332:  .ASCIZ  @IS INCORRECT @
100204  111  123  040  EMS333:  .ASCIZ  @IS NOT SET @
100220  111  123  040  EMS334:  .ASCIZ  @IS SET @
100230  123  110  117  EMS335:  .ASCIZ  @SHOULD NOT BE SET @
100253  123  110  117  EMS336:  .ASCIZ  @SHOULD BE SET @
100272  103  101  116  EMS337:  .ASCIZ  @CANNOT SET @
100306  102  105  103  EMS340:  .ASCIZ  @BECAUSE @
100317  125  123  111  EMS341:  .ASCIZ  @USING @
100326  104  125  122  EMS342:  .ASCIZ  @DURING REGISTER TRANSFER @
100360  125  116  105  EMS343:  .ASCIZ  @UNEXPECTED @
100374  102  125  123  EMS344:  .ASCIZ  @BUS TIMEOUT (04 TRAP) @
100423  102  131  040  EMS345:  .ASCIZ  @BY REGISTER TRANSFER @
100451  103  101  116  EMS346:  .ASCIZ  @CANNOT RESET @
100467  127  111  124  EMS347:  .ASCIZ  @WITHOUT SETTING @
100510  102  125  124  EMS350:  .ASCIZ  @BUT @
```

```
                115 100515      127     101     123  EMS351: .ASCIZ  @WAS RESET BY @
                116 100533      127     101     123  EMS352: .ASCIZ  @WAS SET BY @
                117 100547      111     116     040  EMS353: .ASCIZ  @IN DIAGNOSTIC MODE @
                118 100573      111     123     040  EMS354: .ASCIZ  @IS INCORRECT ACCORDING TO @
                119 100626      103     101     116  EMS355: .ASCIZ  @CANNOT INCREMENT @
                120 100650      127     101     123  EMS356: .ASCIZ  @WAS NOT SET BY @
                121 100670      127     101     123  EMS357: .ASCIZ  @WAS NOT RESET BY @
                122 100712      060     040     124  EMS360: .ASCIZ  @0 TO 1 TRANSITION OF @
                123 100740      061     040     124  EMS361: .ASCIZ  @1 TO 0 TRANSITION OF @
                124 100766      111     123     040  EMS362: .ASCIZ  @IS INCONSISTENT @
                125 101007      103     101     116  EMS363: .ASCIZ  @CANNOT READ @
                126 101024      124     105     123  EMS364: .ASCIZ  @TEST PATTERN IN @
                127 101045      101     116     104  EMS365: .ASCIZ  @AND @
                128 101052      103     101     116  EMS366: .ASCIZ  @CANNOT INITIALIZE @
                129 101075      124     110     105  EMS367: .ASCIZ  @THE COMMAND SEQUENCER HAS BEEN CLOCKED @
                130 101145      122     105     123  EMS370: .ASCIZ  @RESET EARLY @
                131 101162      104     111     104  EMS371: .ASCIZ  @DID NOT RESET ON TIME @
                132 101211      104     125     122  EMS372: .ASCIZ  @DURING COMMAND EXECUTION @
                133 101243      124     117     040  EMS373: .ASCIZ  @TO @
                134 101247      127     111     124  EMS374: .ASCIZ  @WITH ANY COMBINATION OF @
                135 101300      102     131     040  EMS375: .ASCIZ  @BY READING @
                136 101314      102     131     040  EMS376: .ASCIZ  @BY WRITING @
                137 101330      127     101     123  EMS377: .ASCIZ  @WAS SET @
                138 101341      127     101     123  EMS400: .ASCIZ  @WAS NOT SET @
                139 101356      104     111     104  EMS401: .ASCIZ  @DID NOT SET @
                140 101373      127     110     111  EMS402: .ASCIZ  @WHILE @
                141 101402      103     117     115  EMS403: .ASCIZ  @COMMAND SEQUENCER DID NOT ABORT @
                142 101443      127     101     123  EMS404: .ASCIZ  @WAS NOT RESET @
                143 101462      104     125     122  EMS405: .ASCIZ  @DURING @
                144 101472      127     101     123  EMS406: .ASCIZ  @WAS RESET @
                145 101505      123     105     101  EMS407: .ASCIZ  @SEARCH TIMEOUT @
                146
                147 101525      011     104     105  EMS500: .ASCII  @       DEVICE IS NON-EXISTENT,@<CR><LF>
                148 101557      011     104     105          .ASCII  @       DEVICE IS SWITCHED TO OTHER PORT@<CR><LF>
                149 101622      011     124     122          .ASCII  @       TRANSCEIVER ENABLE SWITCH IS OFF@<CR><LF>
                150 101666      011     111     106  EMS501: .ASCIZ  @       IF MODULE, M7686,@<CR><LF>
                151 101713      011     115     101  EMS502: .ASCIZ  @       MASSBUS TRANSCEIVER, M5922 OR M5923     @<CR><LF>
                152 101763      011     103     123  EMS503: .ASCIZ  @       CS MODULE, M7684,@<CR><LF>
                153 102010      011     104     123  EMS504: .ASCIZ  @       DS MODULE, M8685/M7685,@<CR><LF>
                154 102043      011     104     105  EMS505: .ASCIZ  @       DEVICE IS SWITCHED TO A/B PORT POSITION@<CR><LF>
                155 102116      011     104     105  EMS506: .ASCIZ  @       DEVICE IS NOT AN RM05/3/2, OR@<CR><LF>
                156 102157      011     104     105  EMS507: .ASCIZ  @       DEVICE IS SWITCHED TO PROGRAMMABLE PORT POSITION, OR@<CR><LF>
                157 102247      011     101     123  EMS510: .ASCIZ  @       ASSUMING THE RH CONTROLLER HAS NO FAULT@<CR><LF>
                158 102322      015     012     011  EMS511: .ASCII          <CR><LF>@       PROBABLE FAULT(S):@<CR><LF>
                159 102351      011     050     116          .ASCIZ  @       (NOT INCLUDING CABLES OR CONNECTORS)@<CR><LF>
                160
                161 102421      122     105     101  EMS600: .ASCIZ  @READ IN PRESET COMMAND @
                162 102451      117     106     106  EMS601: .ASCIZ  @OFFSET COMMAND @
                163 102471      122     105     124  EMS602: .ASCIZ  @RETURN TO CENTER CENTER COMMAND @
                164 102532      122     105     114  EMS603: .ASCIZ  @RELEASE COMMAND @
                165 102553      122     105     103  EMS604: .ASCIZ  @RECALIBRATE COMMAND @
                166 102600      123     105     105  EMS605: .ASCIZ  @SEEK COMMAND @
                167 102616      123     105     101  EMS606: .ASCIZ  @SEARCH COMMAND @
                168 102636      104     101     124  EMS607: .ASCIZ  @DATA COMMAND @
```

```
  102654  105  130  120  EH1:     .ASCII  @EXPCTD  RECEVD  REGSTR@<CRLF>
  102703  123  124  101           .ASCIZ  @STATUS  STATUS  ADRESS@
  102732  040  102  101  EH2:     .ASCII  @ BASE@<CRLF>
  102740  101  104  122  EH3:     .ASCIZ  @ADRESS@
  102747  105  130  120  EH5:     .ASCII  @EXPCTD   STUCK@<CRLF>
  102766  122  105  123           .ASCIZ  @RESULT  BIT(S)@
  103005  105  130  120  EH7:     .ASCIZ  @EXPCTD  RECEVD@
  103024  122  105  103  EH57:    .ASCII  @RECEVD  DRVTYP@<CRLF>
  103043  104  122  126           .ASCIZ  @DRVTYP  REGADR@
  103062  105  130  120  EH65:    .ASCII  @EXPCTD  RECEVD  TEST@<CRLF>
  103107  123  124  101           .ASCIZ  @STATUS  STATUS  REGSTR@
  103136  105  130  120  EH71:    .ASCII  @EXPCTD  RECEVD   TEST@<CRLF>
  103164  123  124  101           .ASCIZ  @STATUS  STATUS  PATTRN@
  103213  105  130  120  EH115:   .ASCII  @EXPCTD  RECEVD  REGSTR   TEST@<CRLF>
  103251  123  124  101           .ASCIZ  @STATUS  STATUS  ADRESS  PATTRN@
  103310  105  130  120  EH130:   .ASCII  @EXPCTD  RECEVD  REGSTR   TEST   OFFSET@<CRLF>
  103357  123  124  101           .ASCIZ  @STATUS  STATUS  ADRESS  PATTRN  REGSTR@
  103426  105  130  120  EH132:   .ASCII  @EXPCTD  ACTUAL  REGSTR  OFFSET@<CRLF>
  103465  103  117  125           .ASCIZ  @COUNT   COUNT   ADRESS  REGSTR@
  103524  105  130  120  EH142:   .ASCII  @EXPCTD  RECEVD  REGSTR  OFFSET@<CRLF>
  103563  123  124  101           .ASCIZ  @STATUS  STATUS  ADRESS  REGSTR@
  103622  105  130  120  EH145:   .ASCII  @EXPCTD  ACTUAL  REGSTR  RMER1   RMER2@<CRLF>
  103670  103  115  120           .ASCIZ  @CMPERR  CMPERR  ADRESS  PATTRN  PATTRN@

  103737  105  130  120  EH150:   .ASCII  @EXPCTD  ACTUAL  REGSTR FUNCTION@<CRLF>
  103777  122  105  123           .ASCIZ  @RESULT  RESULT  ADRESS   CODE@
  104035  105  130  120  EH213:   .ASCII  @EXPCTD  ACTUAL  STATUS   TEST@<CRLF>
  104073  122  105  123           .ASCIZ  @RESULT  RESULT  ADRESS  REGSTR@
  104132  101  103  124  EH220:   .ASCII  @ACTUAL  REGSTR@<CRLF>
  104151  122  105  123           .ASCIZ  @RESULT  ADRESS@
                         .EVEN
```

ERROR MESSAGE STRINGS

```
 1 104170  001140  001142  001136  ED1:    .WORD   $GDDAT,$BDDAT,$BDADR,0
 2 104200  001136  000000          ED2:    .WORD   $BDADR,0
 3 104204  001140  001142  000000  ED5:    .WORD   $GDDAT,$BDDAT,0
 4 104212  001142  001136  000000  ED57:   .WORD   $BDDAT,$BDADR,0
 5 104220  001140  001142  001174  ED65:   .WORD   $GDDAT,$BDDAT,$TMP0,0
 6 104230  001140  001142  001450  ED71:   .WORD   $GDDAT,$BDDAT,RMHR0,0
 7 104240  001140  001142  001136  ED115:  .WORD   $GDDAT,$BDDAT,$BDADR,$TMP0,0
 8 104252  001140  001142  001136  ED130:  .WORD   $GDDAT,$BDDAT,$BDADR,$TMP0,$TMP1,0
 9 104266  001142  001136          ED220:  .WORD   $BDDAT,$BDADR
10
11 104272     000     000     000  EF1:    .BYTE   0,0,0
12 104275     000                  EF2:    .BYTE   0
13 104276     000     000          EF5:    .BYTE   0,0
14 104300     000     000     000  EF57:   .BYTE   0,0,0,0
15 104304     000     000     000  EF130:  .BYTE   0,0,0,0,0
16                                 .EVEN
```

```
                                BUFFER:
                                BUFONE: .BLKW   258.
                                BUFTWO: .BLKW   258.

                104312                  .=BUFFER

                                HELP:
104312  200                     .ASCII  <CRLF>
104313  200                     .ASCII  <CRLF>
104314  114     111     123     .ASCII  @LIST OF TESTS@<CRLF>
104332  055     055     055     .ASCII  @-------------@<CRLF>
104350  124     061     011     .ASCII  @T1     TRANSFER TEST@<CRLF>
104371  124     062     011     .ASCII  @T2     CTOD TEST@<CRLF>
104406  124     063     011     .ASCII  @T3     MASSBUS INITIALIZE TEST@<CRLF>
104441  124     064     011     .ASCII  @T4     CLEAR STUCK ACTIVE TEST@<CRLF>
104474  124     065     011     .ASCII  @T5     TRISTATE TRANSFER TEST@<CRLF>
104526  124     066     011     .ASCII  @T6     REGISTER SELECT TEST@<CRLF>
104556  124     067     011     .ASCII  @T7     DRIVE TYPE TEST@<CRLF>
104601  124     061     060     .ASCII  @T10    DEVICE AVAILABLE TEST@<CRLF>
104633  124     061     061     .ASCII  @T11    HOLDING REGISTER TRANSFER TEST@<CRLF>
104676  124     061     062     .ASCII  @T12    CONTROL STATUS #1 TRANSFER TEST@<CRLF>
104742  124     061     063     .ASCII  @T13    ERROR REGISTER #1 TRANSFER TEST@<CRLF>
*05006  124     061     064     .ASCII  @T14    CLEAR OFFSET STUCK ACTIVE TEST@<CRLF>
105051  124     061     065     .ASCII  @T15    OFFSET REGISTER TRANSFER TEST@<CRLF>
105113  124     061     066     .ASCII  @T16    ERROR REGISTER #2 TRANSFER TEST@<CRLF>
105157  124     061     067     .ASCII  @T17    SERIAL NUMBER TEST@<CRLF>
105206  124     062     060     .ASCII  @T20    CONTROL BUS PARITY DETECTION TEST@<CRLF>
105254  124     062     061     .ASCII  @T21    CONTROL BUS PARITY GENERATION TEST@<CRLF>
105323  124     062     062     .ASCII  @T22    RMDA,RMDC FAULT TEST@<CRLF>
105354  124     062     063     .ASCII  @T23    DISK ADDRESS TRANSFER TEST@<CRLF>
105413  124     062     064     .ASCII  @T24    DESIRED CYLINDER TRANSFER TEST@<CRLF>
105456  124     062     065     .ASCII  @T25    ILLEGAL REGISTER TEST@<CRLF>
105510  124     062     066     .ASCII  @T26    RESET GO BY INIT TEST@<CRLF>
105542  124     062     067     .ASCII  @T27    DIAGNOSTIC MODE TEST@<CRLF>
105573  124     063     060     .ASCII  @T30    MOL TEST@<CRLF>
105610  124     063     061     .ASCII  @T31    WRITE LOCK TEST@<CRLF>
105634  124     063     062     .ASCII  @T32    DRIVE FAULT TEST@<CRLF>
105661  124     063     063     .ASCII  @T33    SEEK ERROR TEST@<CRLF>
105705  124     063     064     .ASCII  @T34    PIP TEST@<CRLF>
105722  124     063     065     .ASCII  @T35    EBL TEST@<CRLF>
105737  124     063     066     .ASCII  @T36    LAST SECTOR, LAST TRACK TEST@<CRLF>
106000  124     063     067     .ASCII  @T37    RMDA COUNT TEST@<CRLF>
106024  124     064     060     .ASCII  @T40    RMDC COUNT TEST@<CRLF>
106050  124     064     061     .ASCII  @T41    LBT TEST@<CRLF>
106065  124     064     062     .ASCII  @T42    COMPOSITE ERROR TEST@<CRLF>
106116  124     064     063     .ASCII  @T43    WRITE GO TEST@<CRLF>
106140  124     064     064     .ASCII  @T44    BRANCH MULTIPLEXOR TEST@<CRLF>
106174  124     064     065     .ASCII  @T45    SET/RESET GO TEST@<CRLF>
106222  124     064     066     .ASCII  @T46    END 1 RESET GO TEST@<CRLF>
106252  124     064     067     .ASCII  @T47    SET PULSE TEST@<CRLF>
106275  124     065     060     .ASCII  @T50    SET/RESET IVC TEST@<CRLF>
106324  124     065     061     .ASCII  @T51    SET LSC TEST@<CRLF>
106345  124     065     062     .ASCII  @T52    DECODE TEST@<CRLF>
106365  124     065     063     .ASCII  @T53    SET/RESET VOLUME VALID TEST@<CRLF>
106425  124     065     064     .ASCII  @T54    ILLEGAL FUNCTION TEST@<CRLF>
106457  124     065     065     .ASCII  @T55    OCCUPIED TEST@<CRLF>
106501  124     065     066     .ASCII  @T56    READ IN PRESET TEST@<CRLF>
```

```
 58 106531      124     065     067  .ASCII  @T57    RIP/RMOF TEST@<CRLF>
 59 106553      124     066     060  .ASCII  @T60    RMDA/RMDC/RIP TEST@<CRLF>
 60 106602      124     066     061  .ASCII  @T61    OFFSET COMMAND TEST@<CRLF>
 61 106632      124     066     062  .ASCII  @T62    RETURN TO CENTER TEST@<CRLF>
 62 106664      124     066     063  .ASCII  @T63    RMDC CLEAR OFFSET TEST@<CRLF>
 63 106717      124     066     064  .ASCII  @T64    EBL CLEAR OFFSET TEST@<CRLF>
 64 106751      124     066     065  .ASCII  @T65    RUN AND GO TEST@<CRLF>
 65 106775      124     066     066  .ASCII  @T66    SET IAE TEST@<CRLF>
 66 107016      124     066     067  .ASCII  @T67    SEARCH, SEEK, READ, WRITE TEST@<CRLF>
 67 107061      124     067     060  .ASCII  @T70    INVALID TRACK/SECTOR TEST@<CRLF>
 68 107117      124     067     061  .ASCII  @T71    INVALID CYLINDER TEST@<CRLF>
 69 107151      124     067     062  .ASCII  @T72    SET AOE TEST@<CRLF>
 70 107172      124     067     063  .ASCII  @T73    SET RMR TEST@<CRLF>
 71 107213      124     067     064  .ASCII  @T74    PGM STATUS CHECK@<CRLF>
 72 107240      124     067     065  .ASCII  @T75    DVA/DPR STATUS CHECK@<CRLF>
 73 107271      124     067     066  .ASCII  @T76    PORT REQUEST TEST, PART 1@<CRLF>
 74 107327      124     067     067  .ASCII  @T77    PORT REQUEST TEST, PART 2@<CRLF>
 75 107365      124     061     060  .ASCII  @T100   PORT REQUEST TEST, PART 3@<CRLF>
 76 107424      124     061     060  .ASCII  @T101   RELEASE TEST@<CRLF>
 77 107446      124     061     060  .ASCII  @T102   WRITE ATA TEST@<CRLF>
 78 107472      124     061     060  .ASCII  @T103   RESET ATA BY GO TEST@<CRLF>
 79 107524      124     061     060  .ASCII  @T104   UNIT READY ATA TEST@<CRLF>
 80 107555      124     061     060  .ASCII  @T105   ERROR ATA TEST@<CRLF>
 81 107601      124     061     060  .ASCII  @T106   REGISTER TRANSFER ATA TEST@<CRLF>
 82 107641      124     061     060  .ASCII  @T107   P SET ATA TEST@<CRLF>
 83 107665      124     061     061  .ASCII  @T110   SET WLE TEST@<CRLF>
 84 107707      124     061     061  .ASCII  @T111   EXCEPTION TEST@<CRLF>
 85 107733      124     061     061  .ASCII  @T112   RECALIBRATE TEST@<CRLF>
 86 107761      124     061     061  .ASCII  @T113   SEEK TEST@<CRLF>
 87 110000      124     061     061  .ASCII  @T114   SEARCH TEST@<CRLF>
 88 110021      124     061     061  .ASCII  @T115   SEARCH TIMEOUT TEST@<CRLF>
 89 110052      124     061     061  .ASCII  @T116   DATA COMMAND TESTS (1)@<CRLF>
 90 110106      124     061     061  .ASCII  @T117   DATA COMMAND TESTS (2)@<CRLF>
 91 110142      124     061     062  .ASCII  @T120   DATA COMMAND TESTS (3)@<CRLF>
 92 110176      200                  .ASCII  <CRLF>
 93 110177      117     120     105  .ASCII  @OPERATIONAL SWITCH SETTINGS@<CRLF>
 94 110233      055     055     055  .ASCII  @-------------------------@<CRLF>
 95 110267      123     127     111  .ASCII  @SWITCH                  USE@<CRLF>
 96 110304      055     055     055  .ASCII  @------          ------------------@<CRLF>
 97 110341      040     040     061  .ASCII  @  15            HALT ON ERROR@<CRLF>
 98 110365      040     040     061  .ASCII  @  14            LOOP ON TEST@<CRLF>
 99 110410      040     040     061  .ASCII  @  13            INHIBIT ERROR TYPEOUTS@<CRLF>
100 110445      040     040     061  .ASCII  @  12            @<CRLF>
101  10454      040     040     061  .ASCII  @  11            INHIBIT ITERATIONS@<CRLF>
102 110505      040     040     061  .ASCII  @  10            BELL ON ERROR@<CRLF>
103 110531      040     040     040  .ASCII  @   9            LOOP ON ERROR@<CRLF>
104  10555      040     040     040  .ASCII  @   8            LOOP ON TEST IN SWR<7:0>@<CRLF>
105 110614      040     040     040  .ASCII  @   7            TN128@<CRLF>
106 110630      040     040     040  .ASCII  @   6            TN64@<CRLF>
107 110643      040     040     040  .ASCII  @   5            TN32@<CRLF>
108 110656      040     040     040  .ASCII  @   4            TN16@<CRLF>
109 110671      040     040     040  .ASCII  @   3            TN8@<CRLF>
110 110703      040     040     040  .ASCII  @   2            TN4@<CRLF>
111 110715      040     040     040  .ASCII  @   1            TN2@<CRLF>
112 110727      040     040     040  .ASCIZ  @   0            TN1@<CRLF>
113
114          000200                  .END    200
```

SYMBOL TABLE

```
APASE = 176700
ACDW1 = 000000
ACDW2 = 000000
ACPUOP= 000000
ADDW0 = 000000
ADDW1 = 000000
ADDW10= 000000
ADDW11= 000000
ADDW12= 000000
ADDW13= 000000
ADDW14= 000000
ADDW15= 000000
ADDW2 = 000000
ADDW3 = 000000
ADDW4 = 000000
ADDW5 = 000000
ADDW6 = 000000
ADDW7 = 000000
ADDW8 = 000000
ADDW9 = 000000
ADEVCT= 000000
ADEVM = 000000
ADR   = 000001
AENV  = 000000
AENVM   000000
AFATAL= 000000
ALL     062610
AMADR1= 000000
AMADR2= 000000
AMADR3= 000000
AMADR4= 000000
AMAMS1= 000000
AMAMS2= 000000
AMAMS3= 000000
AMAMS4= 000000
AMSGAD= 000000
AMSGLG= 000000
AMSGTY= 000000
AMTYP1= 000000
AMTYP2= 000000
AMTYP3= 000000
AMTYP4= 000000
AOE   = 001000
APASS = 000000
APE   = 100000
APRIOR= 000000
APTCSU= 000040
APTENV= 000001
APTSIZ= 000200
APTSPO= 000100
ASWREG= 000000
ATA   = 100000
ATESTN= 000000
ATNMSK= 000377
ATNTBL  063516
AUNIT = 000000
AUSWR = 000000

AUTSIZ  001326
AVECT1= 120254
AVECT2= 000000
A16   = 000400
A17   = 001000
BADTMO  004562
BAI   = 000010
BB00  = 000001
BB01  = 000002
BB02  = 000004
BB03  = 000010
BB04  = 000020
BB05  = 000040
BB06  = 000100
BB07  = 000200
BB08  = 000400
BB09  = 001000
BIT0  = 000001
BIT00 = 000001
BIT01 = 000002
BIT02 = 000004
BIT03 = 000010
BIT04 = 000020
BIT05   000040
BIT06 = 000100
BIT07 = 000200
BIT08 = 000400
BIT09 = 001000
BIT1  = 000002
BIT10 = 002000
BIT11 = 004000
BIT12 = 010000
BIT13 = 020000
BIT14 = 040000
BIT15 = 100000
BIT2  = 000004
BIT3  = 000010
BIT4  = 000020
BIT5  = 000040
BIT6  = 000100
BIT7  = 000200
BIT8  = 000400
BIT9  = 001000
BLNKS1  063413
BLNKS2  063412
BLNKS3  063411
BLNKS4  063410
BOTADR  060264
BOTFLG  060266
BPTVEC= 000014
BSE   = 100000
BUFFER  104312
BUFONE  104312
BUFTWO  105316
CC      004000
CH    = 002000
CHGADR  001330

CHRCNT  060267
CKSWR = 104410
CLOCK   001536
CLR   = 000040
CMNSTA  006724
CNSL01  062655
CNSL02  062665
CNSL03  062727
CNSL04  062736
CNSL07  062772
CNSL08  063134
CNSL09  063135
CNTCLR  054674
COMMA   062621
CONT  = 000100
CPSAVE  057410
CR    = 000015
CRLF  = 000200
CYLMSK= 001777
DBCK  = 100000
DBEN  = 040000
DBL   = 002000
DCK   = 100000
DDISP = 177570
DEBL  = 020000
DISPLA  001156
DISPRE  000174
DLT   = 100000
DMD   = 000001
DPE   = 000010
DPEHI = 040000
DPELO = 020000
DPR   = 000400
DRIVES  063156
DRQ   = 004000
DRVCLR= 000010
DRY   = 000200
DSWR  = 177570
DTE   = 010000
DULPRT= 024024
DVA   = 004000
DVC   = 000200
EBL   = 020000
ECH   = 000100
ECI   = 004000
ECRC  = 001000
EDT1    072132
EDT115  072150
EDT130  072152
EDT132  072154
EDT2    072134
EDT220  072156
EDT5    072136
EDT57   072140
EDT65   072142
EDT71   072144
EDT74   072146

ED1     104170
ED115   104240
ED130   104252
ED2     104200
ED220   104266
ED5     104204
ED57    104212
ED65    104220
ED71    104230
EECC  = 000020
EFT1    072160
EFT115  072176
EFT130  072200
EFT132  072202
EFT2    072162
EFT220  072204
EFT5    072164
EFT57   072166
EFT65   072170
EFT71   072172
EFT74   072174
EF1     104272
EF130   104304
EF2     104275
EF5     104276
EF57    104300
EHT1    072032
EHT115  072072
EHT130  072076
EHT132  072102
EHT142  072106
EHT145  072112
EHT150  072116
EHT2    072036
EHT213  072122
EHT220  072126
EHT5    072042
EHT57   072052
EHT65   072056
EHT7    072046
EHT71   072062
EHT74   072066
EH1     102654
EH115   103213
EH130   103310
EH132   103426
EH142   103524
EH145   103622
EH150   103737
EH2     102732
EH213   104035
EH220   104132
EH3     102740
EH5     102747
EH57    103024
EH65    103062
EH7     103005

EH71    103136
EMS1    072206
EMS10   072611
EMS11   072654
EMS12   072745
EMS13   073023
EMS14   073072
EMS15   073137
EMS16   073215
EMS17   073276
EMS2    072257
EMS20   073337
EMS21   073403
EMS22   073462
EMS23   073527
EMS24   073605
EMS25   073664
EMS250  076566
EMS251  076624
EMS252  076670
EMS253  076723
EMS254  076756
EMS255  077017
EMS256  077061
EMS257  077111
EMS26   073742
EMS260  077145
EMS261  077176
EMS262  077235
EMS27   074021
EMS3    072326
EMS30   074065
EMS300  077300
EMS301  077316
EMS302  077341
EMS303  077366
EMS304  077377
EMS306  077406
EMS307  077422
EMS31   074145
EMS310  077463
EMS311  077500
EMS312  077544
EMS313  077602
EMS314  077633
EMS315  077662
EMS316  077710
EMS317  077731
EMS32   074216
EMS320  077751
EMS321  077771
EMS322  100011
EMS323  100031
EMS324  100050
EMS325  100070
EMS326  100101
EMS327  100111
```

EMS33	074276	EMS43	074707	EMT112	066230	EMT175	067672	EMT257	071322
EMS330	100117	EMS44	074760	EMT113	066250	EMT176	067710	EMT26	064652
EMS331	100136	EMS45	075021	EMT114	066274	EMT177	067726	EMT260	071346
EMS332	100166	EMS46	075076	EMT115	066316	EMT2	064162	EMT261	071372
EMS333	100204	EMS47	075160	EMT116	066342	EMT20	064522	EMT262	071416
EMS334	100220	EMS5	072424	EMT117	066362	EMT200	067726	EMT263	071434
EMS335	100230	EMS50	075220	EMT12	064376	EMT201	067744	EMT264	071452
EMS336	100253	EMS500	101525	EMT120	066402	EMT202	067762	EMT265	071476
EMS337	100272	EMS501	101666	EMT121	066426	EMT203	070000	EMT266	071514
EMS34	074326	EMS502	101713	EMT122	066446	EMT204	070022	EMT267	071532
EMS340	100306	EMS503	101763	EMT123	066466	EMT205	070042	EMT27	064704
EMS341	100317	EMS504	102010	EMT124	066512	EMT206	070062	EMT270	071546
EMS342	100326	EMS505	102043	EMT125	066530	EMT207	070112	EMT271	071572
EMS343	100360	EMS506	102116	EMT126	066550	EMT21	064542	EMT272	071616
EMS344	100374	EMS507	102157	EMT127	066566	EMT210	070126	EMT273	071634
EMS345	100423	EMS51	075267	EMT13	064414	EMT211	070144	EMT274	071652
EMS346	100451	EMS510	102247	EMT130	066612	EMT212	070162	EMT275	071670
EMS347	100467	EMS511	102322	EMT131	066630	EMT213	070174	EMT276	071710
EMS35	074355	EMS52	075336	EMT132	066646	EMT214	070224	EMT277	071726
EMS350	100510	EMS53	075377	EMT133	066666	EMT215	070236	EMT3	064210
EMS351	100515	EMS54	075440	EMT134	066706	EMT216	070256	EMT30	064716
EMS352	100533	EMS55	075515	EMT135	066722	EMT217	070272	EMT300	071752
EMS353	100547	EMS56	075573	EMT136	066740	EMT22	064562	EMT301	071776
EMS354	100573	EMS57	075666	EMT137	066752	EMT220	070306	EMT302	072014
EMS355	100626	EMS6	072474	EMT14	064432	EMT221	070324	EMT31	064732
EMS356	100650	EMS60	075752	EMT140	066770	EMT222	070342	EMT32	064746
EMS357	100670	EMS600	102421	EMT141	067010	EMT223	070360	EMT33	064762
EMS36	074417	EMS601	102451	EMT142	067024	EMT224	070376	EMT34	065000
EMS360	100712	EMS602	102471	EMT143	067042	EMT225	070416	EMT35	065016
EMS361	100740	EMS603	102532	EMT144	067056	EMT226	070440	EMT36	065032
EMS362	100766	EMS604	102553	EMT145	067100	EMT227	070456	EMT37	065046
EMS363	101007	EMS605	102600	EMT146	067122	EMT23	064576	EMT4	064230
EMS364	101024	EMS606	102616	EMT147	067140	EMT230	070474	EMT40	065062
EMS365	101045	EMS607	102636	EMT15	064450	EMT231	070512	EMT41	065102
EMS366	101052	EMS61	076024	EMT150	067152	EMT232	070526	EMT42	065116
EMS367	101075	EMS62	076077	EMT151	067174	EMT233	070550	EMT43	065142
EMS37	074447	EMS63	076160	EMT152	067206	EMT234	070572	EMT44	065154
EMS370	101145	EMS64	076220	EMT153	067222	EMT235	070620	EMT45	065170
EMS371	101162	EMS65	076270	EMT154	067242	EMT236	070640	EMT46	065204
EMS372	101211	EMS66	076331	EMT155	067262	EMT237	070672	EMT47	065220
EMS373	101243	EMS67	076406	EMT156	067300	EMT24	064616	EMT5	064252
EMS374	101247	EMS7	072540	EMT157	067322	EMT240	070712	EMT50	065242
EMS375	101300	EMS70	076520	EMT16	064466	EMT241	070736	EMT51	065264
EMS376	101314	EMTVEC=	000030	EMT160	067340	EMT242	070762	EMT52	065304
EMS377	101330	EMT1	064154	EMT161	067370	EMT243	071006	EMT53	065320
EMS4	072373	EMT10	064342	EMT162	067406	EMT244	071032	EMT54	065334
EMS40	074501	EMT100	065756	EMT163	067424	EMT245	071050	EMT55	065354
EMS400	101341	EMT101	065776	EMT164	067446	EMT246	071066	EMT56	065374
EMS401	101356	EMT102	066016	EMT165	067470	EMT247	071112	EMT57	065410
EMS402	101373	EMT103	066036	EMT166	067514	EMT25	064636	EMT6	064276
EMS403	101402	EMT104	066062	EMT167	067534	EMT250	071130	EMT60	065424
EMS404	101443	EMT105	066102	EMT17	064504	EMT251	071154	EMT61	065440
EMS405	101462	EMT106	066122	EMT170	067556	EMT252	071200	EMT62	065460
EMS406	101472	EMT107	066146	EMT171	067570	EMT253	071224	EMT63	065500
EMS407	101505	EMT11	064360	EMT172	067604	EMT254	071242	EMT64	065514
EMS41	074557	EMT110	066166	EMT173	067622	EMT255	071260	EMT65	065526
EMS42	074625	EMT111	066210	EMT174	067644	EMT256	071304	EMT66	065542

```
EMT67   065556
EMT7    064320
EMT70   065574
EMT71   065614
EMT72   065640
EMT73   065664
EMT74   065704
EMT75   065714
EMT76   065726
EMT77   065742
ENRGDT  064154
EQUALS  062606
ERR     = 040000
ERRNMB  060262
ERROR   = 104000
ERRTYP  057414
ERRVEC= 000004
ERTY00  060270
ERTY01  060275
ERTY02  060305
ERTY03  060314
ERTY04  060322
ERTY05  060325
ESRC    = 004000
FER     = 000020
FMT16   = 010000
FNCDTB  063416
FNCMSK= 000077
F0      = 000002
F1      = 000004
F2      = 000010
F3      = 000020
F4      = 000040
GETBUF  001336
GO      = 000001
GTSWR   = 104407
HCE     = 000200
HCI     = 002000
HCRC    = 000400
HELP    104312
HT      = 000011
IAE     = 002000
IBSAVE  057412
IDXMSK= 000077
IE      = 000100
ILF     = 000001
ILF02   = 000002
ILF24   = 000024
ILF26   = 000026
ILF30   = 000030
ILF32   = 000032
ILF34   = 000034
ILF36   = 000036
ILF40   = 000040
ILF42   = 000042
ILF44   = 000044
ILF46   - 000046
ILF54   = 000054
ILF56   = 000056
ILF64   = 000064
ILF66   = 000066
ILF74   = 000074
ILF76   = 000076
ILR     = 000002
ILRG50= 000050
ILRG52= 000052
ILRG54= 000054
ILRG56= 000056
ILRG60= 000060
ILRG62= 000062
ILRG64= 000064
ILRG66= 000066
ILRG70= 000070
ILRG72= 000072
ILRG74= 000074
ILRG76= 000076
IOTVEC= 000020
IPCK0   = 000001
IPCK1   = 000002
IPCK2   = 000004
IPCK3   = 000010
IR      = 000100
IVC     = 010000
LBC     = 002000
LBT     = 002000
LCLOCK  054316
LCOUNT  054366
LF      000012
LODEV   063312
LS      - 000004
LSC     = 004000
LST     = 000002
LSTOP   054430
LSTRK   001334
MCLK    = 004000
MCPE    - 020000
MDF     = 000100
MDPE    = 000400
MI      = 000004
MIXED   063526
MOC     = 000400
MOH     = 020000
MOL     = 010000
MRD     = 002000
MR1AAA= 051401
MS      = 000040
MSC     - 000002
MSDRVS  063212
MSE     = 100000
MSEN    = 010000
MSER    = 000200
MSGDRV  063226
MSHELP  062624
MUR     = 001000
MWD     = 000010
MWP     = 000010
MXF     = 001000
N       063404
NDTMSK= 115760
NED     = 010000
NEM     = 004000
NONE    063205
NOP     = 000000
NOTAVL  063344
NOTPRS  063327
NOTRM   063271
NSA     = 100000
OCC     = 100000
OFD     = 000200
OFFSET= 000014
OM      = 000001
ONES    063566
OPE     = 020000
OPI     = 020000
OR      = 000200
PACACK= 000022
PAKACK= 000022
PAR     = 000010
PAT     = 000020
PCLOCK  054300
PCOUNT  054366
PDA     = 000400
PFECH   060332
PFECH1  060342
PFECH2  060430
PFECH3  060446
PFECH4  060454
PGE     = 002000
PGM     = 001000
PHA     = 000200
PIP     = 020000
PIRQ    = 177772
PIRQVE= 000240
PLCLK   054324
PLFS    = 002000
PLSTP   054434
PR0     = 000000
PR1     = 000040
PR2     = 000100
PR3     = 000140
PR4     = 000200
PR5     = 000240
PR6     000300
PR7     = 000340
PS      = 177776
PSEL    = 002000
PSTOP   054422
PSW     = 177776
PUTBUF  001412
PWRVEC= 000024
QUES    062615
PD      - 000070
RDCHR   = 104411
RDLIN   - 104412
RDOCT   = 104413
RDY     = 000200
READY   007120
READY1  007146
RECAL   = 000006
RESREG= 104415
RESVEC= 000010
REX     = 010000
RG      = 040000
RGDTPT  063526
RH      = 000072
RIP     = 000020
RLEASE= 000012
RMAS    = 000016
RMASI   001354
RMASO   001430
RMBA    = 000004
RMBAE   = 000050
RMBAEI  001406
RMBAEO  001462
RMBAI   001342
RMBAO   001416
RMCS1   = 000000
RMCS1I  001336
RMCS1O  001412
RMCS2   = 000010
RMCS2I  001346
RMCS2O  001422
RMCS3   = 000052
RMCS3I  001410
RMCS3O  001464
RMDA    = 000006
RMDAI   001344
RMDAO   001420
RMDB    = 000022
RMDBI   001360
RMDBO   001434
RMDC    = 000034
RMDCI   001372
RMDCO   001446
RMDS    = 000012
RMDSI   001350
RMDSO   001424
RMDT    = 000026
RMDTI   001364
RMDTO   001440
RMEC1   = 000044
RMEC1I  001402
RMEC1O  001456
RMEC2   = 000046
RMEC2I  001404
RMEC2O  001460
RMER1   = 000014
RMER1I  001352
RMER1O  001426
RMER2   = 000042
RMER2I  001400
RMER2O  001454
RMHR    = 000036
RMHRI   001374
RMHRO   001450
RMLA    = 000020
RMLAI   001356
RMLAO   001432
RMMR1   - 000024
RMMR1I  001362
RMMR1O  001436
RMMR2   - 000040
RMMR2I  001376
RMMR2O  001452
RMOF    = 000032
RMOFI   001370
RMOFO   001444
RMR     = 000004
RMSN    = 000030
RMSNI   001366
RMSNO   001442
RMWC    = 000002
RMWCI   001340
RMWCO   001414
R0A     = 100000
R0B     = 040000
RTC     = 000016
R6      =%000006
R7      =%000007
SADMSK= 000377
SAVREG= 104414
SA1     = 000001
SA16    = 000020
SA2     = 000002
SA4     = 000004
SA8     = 000010
SC      = 100000
SCOPE   = 000004
SCTMSK= 003700
SC0     = 000100
SC1     = 000200
SC2     = 000400
SC3     = 001000
SC4     = 002000
SEARCH= 000030
SEEK    = 000004
SETOM   054572
SETVV   054450
SHUT    061730
SIZCLK  054040
SKI     = 040000
SNGPRT= 020024
STACK   = 001100
STANDA  006242
START   004652
```

SYMBOL TABLE

```
START1  004642    TST103  036172    TST6    011364    $ATY4   062356    $GT42P  054004
START2  004656    TST104  036320    TST60   031530    $AUTOB  001150    $HD   = 000000
STKLMT= 177774    TST105  036524    TST61   031746    $BASE   001276    $HIBTS  001100
STOPCL  001540    TST106  036654    TST62   032114    $BDADR  001136    $HIOCT  062052
SWR     001154    TST107  037050    TST63   032340    $BDDAT  001142    $ICNT   001120
SWREG   000176    TST11   012642    TST64   032454    $BELL   001212    $ILLUP  062322
SW0   = 000001    TST110  037246    TST65   032642    $BIN    055070    $INTAG  001151
SW00  = 000001    TST111  037524    TST66   033072    $CDW1   001302    $ITEMB  001130
SW01  = 000002    TST112  040030    TST67   033310    $CDW2   001304    $LF     001220
SW02  = 000004    TST113  041422    TST7    012434    $CHARC  056074    $LFLG   062603
SW03  = 000010    TST114  043150    TST70   033536    $CKSWR  060770    $LLCSR  001522
SW04  = 000020    TST115  045330    TST71   034006    $CMTAG  001114    $LLVEC  001524
SW05  = 000040    TST116  045710    TST72   034214    $CM3  = 000000    $LPADR  001122
SW06  = 000100    TST117  050070    TST73   034504    $CM4  = 000005    $LPCSB  001514
SW07  = 000200    TST12   013102    TST74   034732    $CNTLC  061666    $LPCSR  001512
SW08  = 000400    TST120  052356    TST75   035040    $CNTLG  061700    $LPERR  001124
SW09  = 001000    TST13   013450    TST76   035164    $CNTLU  061673    $LPVEC  001516
SW1   = 000002    TST14   014232    TST77   035314    $CPUOP  001250    $MADR1  001254
SW10  = 002000    TST15   014332    TYPBN = 104406    $CRLF   001217    $MADR2  001260
SW11  = 004000    TST16   014656    TYPDS = 104405    $DBLK   055306    $MADR3  001264
SW12  = 010000    TST17   015370    TYPE  = 104401    $DDW0   001306    $MADR4  001270
SW13  = 020000    TST2    007616    TYPOC = 104402    $DDW1   001310    $MAIL   001222
SW14  = 040000    TST20   015472    TYPON = 104404    $DDW2   001312    $MAMS1  001252
SW15  = 100000    TST21   015756    TYPOS = 104403    $DDW3   001314    $MAMS2  001256
SW2   = 000004    TST22   016074    UNS   = 040000    $DDW4   001316    $MAMS3  001262
SW3   = 000010    TST23   016222    UNTMSK= 000007    $DDW5   001320    $MAMS4  001266
SW4   = 000020    TST24   016474    UNTOFF  063363    $DDW6   001322    $MBADR  001102
SW5   = 000040    TST25   016774    UNTON   063374    $DDW7   001324    $MFLG   062602
SW6   = 000100    TST26   017416    UPE   = 020000    $DEVCT  001332    $MNEW   061716
SW7   = 000200    TST27   017512    USE   = 040000    $DEVM   001300    $MSGAD  001236
SW8   = 000400    TST3    010000    U0    = 000001    $DOAGN  054030    $MSGLG  001240
SW9   = 001000    TST30   020000    U1    = 000002    $DTBL   055276    $MSGTY  001222
SYSTAT  063234    TST31   020314    U2    = 000004    $ENDAD  054020    $MSWR   061705
TADMSK= 177400    TST32   020604    VV    = 000100    $ENDCT  053656    $MTYP1  001253
TAG   = 020000    TST33   021322    WATCH   001534    $ENULL  054034    $MTYP2  001257
TAGADR= 001114    TST34   021634    WC    = 000040    $ENV    001242    $MTYP3  001263
TAP   = 040000    TST35   022126    WCD   = 000050    $ENVM   001243    $MTYP4  001267
TA1   = 000400    TST36   022450    WCE   = 040000    $EOP    053622    $MXCNT  056556
TA16  = 010000    TST37   023010    WCEHI = 010000    $EOPCT  053650    $NULL   001170
TA2   = 001000    TST4    010150    WCELO = 004000    $EOSP   053564    $NWTST= 000001
TA4   = 002000    TST40   023464    WCF   = 000040    $ERFLG  001117    $OCNT   055540
TA8   = 004000    TST41   023750    WCH   = 000052    $ERMAX  001131    $OMODE  055542
TBITVE= 000014    TST42   024232    WD    = 000060    $ERROR  057020    $OVER   056542
TIME    001532    TST43   024634    WH    = 000062    $ERRPC  001132    $PASS   001230
TKVEC = 000060    TST44   025004    WLE   = 004000    $ERRTB  001542    $PASTM  001106
TPVEC = 000064    TST45   025364    WRL   = 004000    $ERTTL  001126    $POWER  062330
TRAPVE= 000034    TST46   026074    XNUDC = 176000    $ESCAP  001210    $PSW    001530
TRE   = 040000    TST47   026370    XNUER2= 001567    $ETABL  001242    $PWRDN  062162
TRTVEC= 000014    TST5    010300    XNUOF = 161577    $ETEND  001326    $PWRMG  062316
TST   = 010000    TST50   026772    XSIZ    005706    $FATAL  001224    $PWRUP  062234
TSTNMB  060260    TST51   027226    XXDP    001332    $FFLG   062604    $QUES   001216
TSTQUE  001466    TST52   027434    Y       063406    $FILLC  001172    $RDCHR  061332
TST1    007306    TST53   030266    ZEROS   063630    $FILLS  001171    $RDLIN  061422
TST10   012556    TST54   030540    $APTHD  001100    $GDADR  001134    $RDOCT  061752
TST100  035446    TST55   030742    $ATYC   062364    $GDDAT  001140    $RDSZ = 000010
TST101  035600    TST56   031154    $ATY1   062340    $GET42  054010    $RESRE  054760
TST102  035770    TST57   031350    $ATY3   062346    $GTSWR  061060    $RM02   063252
```

SYMBOL TABLE

```
$RM03   063257          $SW08T  056560          $TMP0   001174          $TRPAD  062126          $UNIT   001234
$RM05   063264          $TESTN  001226          $TMP1   001176          $TSTM   001104          $UNITM  001110
$RTNAD  054032          $TIMES  001206          $TMP2   001200          $TSTNM  001116          $USWR   001246
$SAVRE  054722          $TKB    001162          $TMP3   001202          $TTYIN  061656          $VECT1  001272
$SAVR6  062326          $TKCNT  060460          $TMP4   001204          $TYPBN  055016          $VECT2  001274
$SCOPE  056100          $TKINT  060470          $TN     = 000121        $TYPDS  055072          $XOFF   = 000023
$SETUP= 000137          $TKQEN= 060467          $TPB    001166          $TYPE   055544          $XON    = 000021
$STUP   = 177777        $TKQIN  060462          $TPFLG  001173          $TYPEC  055756          $XTSTR  056116
$SVLAD  056506          $TKQOU  060464          $TPS    001164          $TYPEX  056076          $$GET4= 000000
$SVPC   = 000210        $TKQSR  060466          $TRAP   062054          $TYPOC  055342          $$SW08= 000121
$SWR    = 167400        $TKS    001160          $TRAP2  062114          $TYPON  055356          $OFILL  055541
$SWREG  001244          $TKSRV  060540          $TRP    = 000016        $TYPOS  055316          .$X     = 001100
$SWRMK= 000000

. ABS.  110742      000
        000000      001
ERRORS DETECTED:  0

VIRTUAL MEMORY USED:  62208 WORDS  ( 243 PAGES)
DYNAMIC MEMORY AVAILABLE FOR  70 PAGES
CZRMPB.BIC,CZRMPB/C CZRMPB.DOC,CZRMPB,SYSMAC/M
```

```
$$GET4   14-20    14-20#
$$SW08   24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
         24-1     24-1     24-1     24-1     24-1     24-1     24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#
         24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#
         24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#
         24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#
         24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#
         24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#    24-1#
         24-1#    24-1#    24-1#
$OFILL   22-1     22-1#    22-1*    22-1*
$40CAT   24-1     25-1
$APTHD    5-8      5-8#
$ASTAT   31-1     31-1
$ATY1    31-1#
$ATY3    23-1     31-1#
$ATY4    25-1     31-1#
$ATYC    31-1     31-1#
$AUTOB    6-0#     9-28*   27-1     27-1     27-1
$BASE     6-0#     9-76    10-33    10-42*   11-56    13-1     13-67    13-73    13-106   13-137   13-159   13-329   13-521   13-546
         13-557   13-610   13-677   13-781   13-794   13-864   13-973   13-996   13-:49   13-:77   13-:97   13-;41   13-<03   13-=02
         13-=19   13-=77   13->42   13-?03   13-@05   13-@66   13-A26   13-B08   13-B76   13-C78   13-D31   13-D84   13-E64   13-E99
         13-G20   13-H70   13-I45   13-J84   13-K30   13-K64   13-L96   13-M56   13-M92   13-N40   13-N79   13-O09   13-O48   13-O74
         13-P18   13-P39   13-P66   13-Q10   13-Q87   13-R35   13-R85   13-S22   13-S72   13-T29   13-T47   13-T70   13-T99   13-U28
         13-U58   13-U95   13-V33   13-V53   13-V86   13-W07   13-W56   13-X01   13-X71   13-Y48   13-\10   13-_01   13-c07   13-c75
         13-g77   13-k51   18-9     26-25
$BDADR    6-0#    13-14*   13-15*   13-35*   13-36*   13-67*   13-540*  13-541*  13-553*  13-561*  13-562*  13-623*  13-624*  13-636*
         13-637*  13-650*  13-651*  13-670*  13-671*  13-690*  13-691*  13-698*  13-699*  13-706*  13-707*  13-719*  13-720*  13-727*
         13-728*  13-736*  13-737*  13-749*  13-750*  13-773*  13-774*  13-789*  13-790*  13-797*  13-798*  13-867*  13-868*  13-976*
         13-977*  13-:13*  13-:14*  13-;09*  13-;10*  13-;20*  13-;21*  13-;34*  13-;35*  13-;46*  13-;47*  13-<15*  13-<16*  13-<42*
         13-=05*  13-=21*  13-=22*  13-=80*  13-=81*  13->45*  13->46*  13-?07*  13-?08*  13-?29*  13-?30*  13-?46*  13-?47*  13-?62*
         13-?63*  13-?88*  13-?89*  13-@09*  13-@10*  13-@70*  13-@71*  13-A29*  13-A30*  13-B11*  13-B12*  13-B78*  13-B79*  13-C81*
         13-C82*  13-D33*  13-D34*  13-D87*  13-D88*  13-E66*  13-F01*  13-F02*  13-G43*  13-G57*  13-G58*  13-G70*  13-G95*  13-G96*
         13-H73*  13-I47*  13-I48*  13-J86*  13-J87*  13-K32*  13-K33*  13-K84*  13-K85*  13-L05*  13-L06*  13-L26*  13-L27*  13-L60*
         13-L61*  13-L75*  13-L76*  13-L87*  13-L88*  13-L98*  13-L99*  13-M79*  13-M80*  13-N26*  13-N27*  13-N81*  13-N82*  13-O34*
         13-O35*  13-O43*  13-O44*  13-O50*  13-O51*  13-O76*  13-O77*  13-P13*  13-P14*  13-P20*  13-P21*  13-P43*  13-P44*  13-P70*
         13-P71*  13-R21*  13-R22*  13-R65*  13-R66*  13-S09*  13-S10*  13-S59*  13-S60*  13-S74*  13-S75*  13-T42*  13-T55*  13-T64*
         13-T65*  13-T94*  13-T95*  13-U23*  13-U24*  13-U52*  13-U53*  13-U90*  13-U91*  13-V15*  13-V16*  13-V28*  13-V29*  13-V47*
         13-V48*  13-V67*  13-V68*  13-V79*  13-V80*  13-W01*  13-W02*  13-W28*  13-W29*  13-W80*  13-W81*  13-X32*  13-X33*  13-X93*
         13-X94*  13-Y23*  13-Y24*  13-Y80*  13-Y81*  13-Z18*  13-Z48*  13-Z49*  13-Z91*  13-Z92*  13-[33*  13-[48*  13-[73*  13-[74*
         13-\47*  13-\48*  13-\88*  13-]21*  13-]22*  13-]67*  13-]68*  13-^12*  13-^27*  13-^57*  13-^58*  13-_38*  13-_39*  13-_79*
         13-`15*  13-`16*  13-`78*  13-`79*  13-a23*  13-a38*  13-a93*  13-a94*  13-b10*  13-b54*  13-b55*  13-c51*  13-c52*  13-c67*
         13-c68*  13-d11*  13-d12*  13-d60*  13-d61*  13-d88*  13-d89*  13-e04*  13-e35*  13-e36*  13-e62*  13-e63*  13-e92*  13-e93*
         13-f30*  13-f31*  13-f64*  13-f65*  13-g30*  13-g31*  13-h02*  13-h03*  13-h18*  13-h19*  13-h46*  13-h47*  13-h98*  13-h99*
         13-i33*  13-i62*  13-i63*  13-i94*  13-i95*  13-j51*  13-j52*  13-j80*  13-j81*  13-k16*  13-k17*  13-k45*  13-k46*  13-l13*
         13-l14*  13-l44*  13-l45*  13-m01*  13-m02*  13-m33*  13-m34*  16-23*  16-24*  17-22*  17-23*  42-1     42-2     42-4
         42-7     42-8     42-9
```

$LF	6-0#	23-1	23-1	25-1	25-1	27-1	27-1	27-1						
$LFLG	31-1#	31-1*												
$LLCSR	7-0#	15-26	15-55*	15-83*										
$LLVEC	7-0#	15-17	15-17*	15-18*	15-29*	15-30*	15-42	15-42*	15-43*					
$LPADR	6-0#	9-23*	24-1	24-1	24-1	24-1*	24-1*	24-1*						
$LPCSB	7-0#	15-51*												
$LPCSR	7-0#	15-12	15-52*	15-80*										
$LPERR	6-0#	9-23*	13-Y84*	13-Z22*	13-Z53*	13-Z95*	13-[52*	13-\51*	13-\92*	13-]26*	13-]71*	13-^31*	13-_42*	13-_86*
	13-_20*	13-_82*	13-a42*	13-b13*	13-d15*	13-e07*	13-e66*	13-f38*	13-h21*	13-i01*	13-i65*	13-j55*	24-1	24-1
	24-1	24-1*	25-1											
$LPVEC	7-0#	15-15*	15-16*	15-31	15-31*	15-32*	15-40	15-40*	15-41*					
$MADR1	6-0#													
$MADR2	6-0#													
$MADR3	6-0#													
$MADR4	6-0#													
$MAIL	5-8	5-8	6-0#	9-23	9-28	13-1	13-73	13-106	13-137	13-159	13-329	13-521	13-546	13-557
	13-610	13-677	13-781	13-794	13-864	13-973	13-996	13-:49	13-:77	13-:97	13-;41	13-<03	13-=02	13-=19
	13-=77	13->42	13-?03	13-@05	13-@66	13-A26	13-B08	13-B76	13-C78	13-D31	13-D84	13-E64	13-E99	13-G20
	13-H70	13-I45	13-J84	13-K30	13-K64	13-L96	13-M56	13-M92	13-N40	13-N79	13-009	13-048	13-074	13-P18
	13-P39	13-P66	13-Q10	13-Q87	13-R35	13-R85	13-S22	13-S72	13-T29	13-T47	13-T70	13-T99	13-U28	13-U58
	13-U95	13-V33	13-V53	13-V86	13-W07	13-W56	13-X01	13-X71	13-Y48	13-\10	13-_01	13-c07	13-c75	13-g77
	13-k51	23-1	24-1	25-1										
$MAMS1	6-0#													
$MAMS2	6-0#													
$MAMS3	6-0#													
$MAMS4	6-0#													
$MBADR	5-8#													
$MFLG	31-1	31-1#	31-1*	31-1*										
$MNEW	27-1	27-1#												
$MSGAD	6-0#	31-1	31-1*											
$MSGLG	6-0#	31-1*												
$MSGTY	6-0#	31-1	31-1	31-1*	31-1*									
$MSWR	27-1	27-1#												
$MTYP1	6-0#													
$MTYP2	6-0#													
$MTYP3	6-0#													
$MTYP4	6-0#													
$MXCNT	24-1	24-1	24-1	24-1#										
$NULL	6-0#	23-1	23-1	23-1										
$NWTST	13-1	13-1	13-1#	13-1#	13-73	13-73	13-73#	13-73#	13-106	13-106	13-106#	13-106#	13-137	13-137
	13-137#	13-137#	13-159	13-159	13-159#	13-159#	13-329	13-329	13-329#	13-329#	13-521	13-521	13-521#	13-521#
	13-546	13-546	13-546#	13-546#	13-557	13-557	13-557#	13-557#	13-610	13-610	13-610#	13-610#	13-677	13-677
	13-677#	13-677#	13-781	13-781	13-781#	13-781#	13-794	13-794	13-794#	13-794#	13-864	13-864	13-864#	13-864#
	13-973	13-973	13-973#	13-973#	13-996	13-996	13-996#	13-996#	13-:49	13-:49	13-:49#	13-:49#	13-:77	13-:77
	13-:77#	13-:77#	13-:97	13-:97	13-:97#	13-:97#	13-;41	13-;41	13-;41#	13-;41#	13-<03	13-<03	13-<03#	13-<03#
	13-=02	13-=02	13-=02#	13-=02#	13-=19	13-=19	13-=19#	13-=19#	13-=77	13-=77	13-=77#	13-=77#	13->42	13->42
	13->42#	13->42#	13-?03	13-?03	13-?03#	13-?03#	13-@05	13-@05	13-@05#	13-@05#	13-@66	13-@66	13-@66#	13-@66#
	13-A26	13-A26	13-A26#	13-A26#	13-B08	13-B08	13-B08#	13-B08#	13-B76	13-B76	13-B76#	13-B76#	13-C78	13-C78
	13-C78#	13-C78#	13-D31	13-D31	13-D31#	13-D31#	13-D84	13-D84	13-D84#	13-D84#	13-E64	13-E64	13-E64#	13-E64#
	13-E99	13-E99	13-E99#	13-E99#	13-G20	13-G20	13-G20#	13-G20#	13-H70	13-H70	13-H70#	13-H70#	13-I45	13-I45
	13-I45#	13-I45#	13-J84	13-J84	13-J84#	13-J84#	13-K30	13-K30	13-K30#	13-K30#	13-K64	13-K64	13-K64#	13-K64#
	13-L96	13-L96	13-L96#	13-L96#	13-M56	13-M56	13-M56#	13-M56#	13-M92	13-M92	13-M92#	13-M92#	13-N40	13-N40
	13-N40#	13-N40#	13-N79	13-N79	13-N79#	13-N79#	13-009	13-009	13-009#	13-009#	13-048	13-048	13-048#	13-048#
	13-074	13-074	13-074#	13-074#	13-P18	13-P18	13-P18#	13-P18#	13-P39	13-P39	13-P39#	13-P39#	13-P66	13-P66
	13-P66#	13-P66#	13-Q10	13-Q10	13-Q10#	13-Q10#	13-Q87	13-Q87	13-Q87#	13-Q87#	13-R35	13-R35	13-R35#	13-R35#
	13-R85	13-R85	13-R85#	13-R85#	13-S22	13-S22	13-S22#	13-S22#	13-S72	13-S72	13-S72#	13-S72#	13-T29	13-T29
	13-T29#	13-T29#	13-T47	13-T47	13-T47#	13-T47#	13-T70	13-T70	13-T70#	13-T70#	13-T99	13-T99	13-T99#	13-T99#

```
                13-U28   13-U28   13-U28#  13-U28#  13-U58   13-U58   13-U58#  13-U58#  13-U95   13-U95   13-U95#  13-U95#  13-V33   13-V33
                13-V33#  13-V33#  13-V53   13-V53   13-V53#  13-V53#  13-V86   13-V86   13-V86#  13-V86#  13-W07   13-W07   13-W07#  13-W07#
                13-W56   13-W56   13-W56#  13-W56#  13-X01   13-X01   13-X01#  13-X01#  13-X71   13-X71   13-X71#  13-X71#  13-Y48   13-Y48
                13-Y48#  13-Y48#  13-\10   13-\10   13-\10#  13-\10#  13-_01   13-_01   13-_01#  13-_01#  13-c07   13-c07   13-c07#  13-c07#
                13-c75   13-c75   13-c75#  13-c75#  13-g77   13-g77   13-g77#  13-g77#  13-k51   13-k51   13-k51#  13-k51#
$OCNT           22-1#    22-1*    22-1*
$OMODE          22-1     22-1#    22-1*    22-1*    22-1*    22-1*
$OVER           24-1     24-1     24-1     24-1     24-1#
$PASS            6-0#     9-23*   14-20    14-20    14-20    14-20*   14-20*   24-1     24-1     24-1
$PASTM           5-8#
$POWER          30-1     30-1#
$PSW             7-0#    15-59*   15-86
$PWRDN           9-23    30-1     30-1#
$PWRMG          30-1#
$PWRUP          30-1     30-1#
$QUES            6-0#    23-1     23-1     25-1     25-1     27-1     27-1     27-1     27-1
$R2A            29-1
$RDCHR          27-1#    29-1     29-1
$RDDEC          29-1
$RDLIN          27-1#    29-1     29-1
$RDOCT          28-1#    29-1     29-1
$RDSZ           27-1     27-1#
$RESRE          19-1#    29-1
$RM02            9-92    26-32    32-21#
$RM03            9-98    26-28    32-22#
$RM05            9-103   26-36    32-23#
$RTNAD          14-20#
$SAVR6          30-1     30-1#    30-1*    30-1*    30-1*
$SAVRE          19-1#    29-1     29-1
$SCOPE           9-23    24-1#
$SETUP           4-978    4-978    4-978    4-978    4-978    4-978    4-978#   4-978#   4-978#   4-978#   4-978#   4-978#   4-978#   9-23
                 9-23     9-23     9-23     9-23     9-23     9-23     9-23     9-23     9-23     9-23     9-23     9-28     9-28     9-28
                14-20    14-20    24-1     25-1     25-1     25-1     25-1     27-1     27-1     27-1     27-1     27-1
$STUP            4-978    4-978    4-978    4-978    4-978    4-978    4-978#   4-978#   4-978#   4-978#   4-978#   4-978#   4-978#   4-978#
                 4-978#   4-978#   4-978#   4-978#
$SVLAD          24-1     24-1#
$SVPC            5-5      5-5#
$SW08T          24-1     24-1#
$SWR             4-667#   4-678    4-679    4-679    4-679    4-679    4-679    4-679    4-679    4-679    6-0      6-0      6-0      9-23
                 9-23     9-23     9-23     9-23     13-1     13-73    13-106   13-137   13-159   13-329   13-521   13-546   13-557   13-610
                13-677   13-781   13-794   13-864   13-973   13-996   13-:49   13-:77   13-:97   13-;41   13-<03   13-=02   13- 19   13- 77
                13->42   13-?03   13-@05   13-@66   13-A26   13-B08   13-B76   13-C78   13-D31   13-D84   13-E64   13-E99   13-G20   13-H70
                13-I45   13-J84   13-K30   13-K64   13-L96   13-M56   13-M92   13-N40   13-N79   13-O09   13-O48   13-O74   13-P18   13-P39
                13-P66   13-Q10   13-Q87   13-R35   13-R85   13-S22   13-S72   13-T29   13-T47   13-T70   13-T99   13-U28   13-U58   13-U95
                13-V33   13-V53   13-V86   13-W07   13-W56   13-X01   13-X71   13-Y48   13-\10   13-_01   13-c07   13-c75   13-g77   13-k51
                14-20    14-20    14-20    14-20    14-20    24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1
                24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     24-1     25-1     25-1     25-1
                25-1     25-1     25-1     25-1     25-1     25-1     25-1     25-1     30-1
$SWREG           6-0#     9-23
$SWRMK           4-679    4-679    4-679    4-679    4-679    4-679    4-679    4-679    4-679    24-1     24-1     24-1     24-1     24-1
                24-1     24-1     24-1     24-1     24-1
$TESTN           6-0#    13-1*    13-73*   13-106*  13-137*  13-159*  13-329*  13-521*  13-546*  13-557*  13-610*  13-677*  13-781*  13-794*
                13-864*  13-973*  13-996*  13-:49*  13-:77*  13-:97*  13-;41*  13-<03*  13-=02*  13-=19*  13-=77*  13->42*  13-?03*  13-@05*
                13-@66*  13-A26*  13-B08*  13-B76*  13-C78*  13-D31*  13-D84*  13-E64*  13-E99*  13-G20*  13-H70*  13-I45*  13-J84*  13-K30*
                13-K64*  13-L96*  13-M56*  13-M92*  13-N40*  13-N79*  13-O09*  13-O48*  13-O74*  13-P18*  13-P39*  13-P66*  13-Q10*  13-Q87*
                13-R35*  13-R85*  13-S22*  13-S72*  13-T29*  13-T47*  13-T70*  13-T99*  13-U28*  13-U58*  13-U95*  13-V33*  13-V53*  13-V86*
                13-W07*  13-W56*  13-X01*  13-X71*  13-Y48*  13-\10*  13-_01*  13-c07*  13-c75*  13-g77*  13-k51*  24-1*    26-45
```

CROSS REFERENCE TABLE (CREF V01-05)

SEQ 0292

$TIMES	6-0#	9-23*	11-47*	14-20*	24-1	24-1	24-1	24-1*	24-1*					
$TKB	6-0#	23-1	23-1	27-1	27-1	27-1	27-1	27-1	27-1	27-1				
$TKCNT	27-1	27-1	27-1#	27-1*	27-1*	27-1*								
$TKINT	10-4	11-48	27-1	27-1	27-1#									
$TKQEN	27-1	27-1	27-1#											
$TKQIN	27-1	27-1	27-1#	27-1*	27-1*	27-1*	27-1*							
$TKQOU	27-1	27-1	27-1#	27-1*	27-1*	27-1*								
$TKQSR	27-1	27-1	27-1	27-1#										
$TKS	6-0#	23-1	23-1	27-1	27-1	27-1	27-1	27-1	27-1*	27-1*	27-1*	27-1*	27-1*	27-1*
$TKSRV	27-1	27-1#												
$TMP0	6-0#	13-<50*	13-=66*	13-=67*	13->26*	13->88*	13-?90*	13-@51*	13-A11*	13-A80*	13-B36*	13-B54*	13-B98*	13-C99*
	13-D51*	13-E24*	13-E54*	13-F30*	13-I73*	13-K18*	13-M30*	13-M81*	13-N25*	13-P93*	13-R23*	13-R67*	13-S11*	13-S36*
	13-S98*	13-T33*	13-T37	13-U76*	13-U80	13-W27*	13-W83*	13-X34*	13-Y25*	13-L25*	13-L28*	13-L29	13-L60*	13-L63*
	13-L64	16-28*	42-5	42-7	42-8									
$TMP1	6-0#	10-13*	10-14	10-16	10-36*	10-38	10-42	10-50*	10-52	10-56	10-66*	10-67	10-73	10-75
	10-76	10-78	10-83*	10-84	10-87	10-88	10-90	10-95	13-B19*	13-E25*	13-E55*	13-X35*	42-8	
$TMP2	6-0#													
$TMP3	6-0#													
$TMP4	6-0#													
$TN	4-668#	4-678	13-1	13-1	13-1	13-1#	13-73	13-73	13-73	13-73#	13-106	13-106	13-106	13-106#
	13-137	13-137	13-137	13-137#	13-159	13-159	13-159	13-159#	13-329	13-329	13-329	13-329#	13-521	13-521
	13-521	13-521#	13-546	13-546	13-546	13-546#	13-557	13-557	13-557	13-557#	13-610	13-610	13-610	13-610#
	13-677	13-677	13-677	13-677#	13-781	13-781	13-781	13-781#	13-794	13-794	13-794	13-794#	13-864	13-864
	13-864	13-864#	13-973	13-973	13-973#	13-996	13-996	13-996	13-996#	13-:49	13-:49	13-:49	13-:49	13-:49#
	13-:77	13-:77	13-:77	13-:77#	13-:97	13-:97	13-:97	13-:97#	13-;41	13-;41	13-;41	13-;41#	13-<03	13-<03
	13-<03	13-<03#	13-=02	13-=02	13-=02	13-=02#	13-=19	13-=19	13-=19	13-=19#	13-=77	13-=77	13-=77	13-=77#
	13->42	13->42	13->42	13->42#	13-?03	13-?03	13-?03	13-?03#	13-@05	13-@05	13-@05	13-@05#	13-@66	13-@66
	13-@66	13-@66#	13-A26	13-A26	13-A26	13-A26#	13-B08	13-B08	13-B08	13-B08#	13-B76	13-B76	13-B76	13-B76#
	13-C78	13-C78	13-C78	13-C78#	13-D31	13-D31	13-D31	13-D31#	13-D84	13-D84	13-D84	13-D84#	13-E64	13-E64
	13-E64	13-E64#	13-E99	13-E99	13-E99	13-E99#	13-G20	13-G20	13-G20	13-G20#	13-H70	13-H70	13-H70	13-H70#
	13-I45	13-I45	13-I45	13-I45#	13-J84	13-J84	13-J84	13-J84#	13-K30	13-K30	13-K30	13-K30#	13-K64	13-K64
	13-K64	13-K64#	13-L96	13-L96	13-L96	13-L96#	13-M56	13-M56	13-M56	13-M56#	13-M92	13-M92	13-M92	13-M92#
	13-N40	13-N40	13-N40	13-N40#	13-N79	13-N79	13-N79	13-N79#	13-O09	13-O09	13-O09	13-O09#	13-O48	13-O48
	13-O48	13-O48#	13-O74	13-O74	13-O74	13-O74#	13-P18	13-P18	13-P18	13-P18#	13-P39	13-P39	13-P39	13-P39#
	13-P66	13-P66	13-P66	13-P66#	13-Q10	13-Q10	13-Q10	13-Q10#	13-Q87	13-Q87	13-Q87	13-Q87#	13-R35	13-R35
	13-R35	13-R35#	13-R85	13-R85	13-R85	13-R85#	13-S22	13-S22	13-S22	13-S22#	13-S72	13-S72	13-S72	13-S72#
	13-T29	13-T29	13-T29	13-T29#	13-T47	13-T47	13-T47	13-T47#	13-T70	13-T70	13-T70	13-T70#	13-T99	13-T99
	13-T99	13-T99#	13-U28	13-U28	13-U28	13-U28#	13-U58	13-U58	13-U58	13-U58#	13-U95	13-U95	13-U95	13-U95#
	13-V33	13-V33	13-V33	13-V33#	13-V53	13-V53	13-V53	13-V53#	13-V86	13-V86	13-V86	13-V86#	13-W07	13-W07
	13-W07	13-W07#	13-W56	13-W56	13-W56	13-W56#	13-X01	13-X01	13-X01	13-X01#	13-X71	13-X71	13-X71	13-X71#
	13-Y48	13-Y48	13-Y48	13-Y48#	13-\10	13-\10	13-\10	13-\10#	13-_01	13-_01	13-_01	13-_01#	13-c07	13-c07
	13-c07	13-c07#	13-c75	13-c75	13-c75	13-c75#	13-g77	13-g77	13-g77	13-g77#	13-k51	13-k51	13-k51	13-k51#
	24-1	24-1												
$TPB	6-0#	23-1	23-1	23-1*										
$TPFLG	6-0#	23-1	23-1	23-1										
$TPS	6-0#	23-1	23-1	23-1										
$TRAP	9-23	29-1#												
$TRAP2	29-1	29-1#												
$TRP	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1
	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1
	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1
	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1	29-1#	29-1#	29-1#	29-1#
	29-1#	29-1#	29-1#	29-1#	29-1#	29-1#	29-1#	29-1#	29-1#	29-1#				
$TRPAD	29-1	29-1#												
$TSTM	5-8#													
$TSTNM	6-0#	11-46*	14-20*	24-1	24-1	24-1	24-1	24-1	24-1*	24-1*	25-1	25-1	25-1	
$TTYIN	27-1	27-1	27-1	27-1	27-1	27-1#								

CROSS REFERENCE TABLE (CREF V01-05)

$TYPBN	20-1#	29-1	29-1		
$TYPDS	21-1#	29-1	29-1		
$TYPE	23-1#	29-1	29-1	31-1	
$TYPEC	23-1	23-1	23-1	23-1#	27-1
$TYPEX	23-1	23-1	23-1#		
$TYPOC	22-1#	29-1	29-1		
$TYPON	22-1	22-1#	29-1		
$TYPOS	22-1#	29-1			
$UNIT	6-0#	11-50*	11-69	26-22	
$UNITM	5-8#				
$USWR	6-0#				
$VECT1	6-0#	10-46	10-56*		
$VECT2	6-0#				
$XOFF	23-1	23-1			
$XON	23-1	23-1	27-1		
$XTSTR	24-1#				
.$ASTA	31-1	31-1			
.$X	5-8	5-8#			
A16	4-924#	13-<85			
A17	4-923#	13-<85			
ABASE	4-975#	6-0	6-0		
ACDW1	6-0	6-0			
ACDW2	6-0	6-0			
ACPUOP	6-0	6-0			
ADDW0	6-0	6-0			
ADDW1	6-0	6-0			
ADDW10	6-0				
ADDW11	6-0				
ADDW12	6-0				
ADDW13	6-0				
ADDW14	6-0				
ADDW15	6-0				
ADDW2	6-0	6-0			
ADDW3	6-0	6-0			
ADDW4	6-0	6-0			
ADDW5	6-0	6-0			
ADDW6	6-0	6-0			
ADDW7	6-0	6-0			
ADDW8	6-0				
ADDW9	6-0				
ADEVCT	6-0	6-0			
ADEVM	6-0	6-0			

ADR	13-1	13-1#	13-1#	13-73	13-73#	13-73#	13-106	13-106#	13-106#	13-137	13-137#	13-137#	13-159	13-159#
	13-159#	13-329	13-329#	13-329#	13-521	13-521#	13-521#	13-546	13-546#	13-546#	13-557	13-557#	13-557#	13-610
	13-610#	13-610#	13-677	13-677#	13-677#	13-781	13-781#	13-781#	13-794	13-794#	13-794#	13-864	13-864#	13-864#
	13-973	13-973#	13-973#	13-996	13-996#	13-996#	13-:49	13-:49#	13-:49#	13-:77	13-:77#	13-:77#	13-:97	13-:97#
	13-:97#	13-;41	13-;41#	13-;41#	13-<03	13-<03#	13-<03#	13-=02	13-=02#	13-=02#	13-=19	13-=19#	13-=19#	13-@77
	13-@77#	13-@77#	13->42	13->42#	13->42#	13-?03	13-?03#	13-?03#	13-@05	13-@05#	13-@66	13-@66#	13-@66#	13-@66#
	13-A26	13-A26#	13-A26#	13-B08	13-B08#	13-B08#	13-B76	13-B76#	13-B76#	13-C78	13-C78#	13-C78#	13-D31	13-D31#
	13-D31#	13-D84	13-D84#	13-D84#	13-E64	13-E64#	13-E64#	13-E99	13-E99#	13-E99#	13-G20	13-G20#	13-G20#	13-H70
	13-H70#	13-H70#	13-I45	13-I45#	13-I45#	13-J84	13-J84#	13-J84#	13-K30	13-K30#	13-K30#	13-K64	13-K64#	13-K64#
	13-L96	13-L96#	13-L96#	13-M56	13-M56#	13-M56#	13-M92	13-M92#	13-M92#	13-N40	13-N40#	13-N40#	13-N79	13-N79#
	13-N79#	13-009	13-009#	13-009#	13-048	13-048#	13-048#	13-074	13-074#	13-074#	13-P18	13-P18#	13-P18#	13-P39
	13-P39#	13-P39#	13-P66	13-P66#	13-P66#	13-Q10	13-Q10#	13-Q10#	13-Q87	13-Q87#	13-Q87#	13-R35	13-R35#	13-R35#
	13-R85	13-R85#	13-R85#	13-S22	13-S22#	13-S22#	13-S72	13-S72#	13-S72#	13-T29	13-T29#	13-T29#	13-T47	13-T47#
	13-T47#	13-T70	13-T70#	13-T70#	13-T99	13-T99#	13-T99#	13-U28	13-U28#	13-U28#	13-U58	13-U58#	13-U58#	13-U95
	13-U95#	13-U95#	13-V33	13-V33#	13-V33#	13-V53	13-V53#	13-V53#	13-V86	13-V86#	13-V86#	13-W07	13-W07#	13-W07#

	13-j89	13-j95	13-j99	13-k00	13-k07	13-k08	13-k22	13-k26	13-k27	13-k34	13-k35	13-k99	13-l39	13-l40
	13-l52	13-l53	13-l95	13-l96	13-m09	13-m10	13-m28	13-m29						
DBL	4-958#													
DCK	4-764#	4-781	13-695	13-724										
DDISP	4-682#	6-0	9-23											
DEBL	4-804#	13-A44	13-A55	13-C04	13-C44	13-D06	13-D68	13-P57	13-S52					
DISPLA	6-0#	9-23*	9-23*	24-1*	25-1*									
DISPRE	5-1#	9-23												
DLT	4-930#													
DMD	4-816#	4-834#	4-835	13-144	13-154	13-=28	13-=35	13-=38	13-=47	13-=49	13-=51	13-=64	13-=85	13-=95
	13-=96	13->16	13->32	13->34	13->37	13->51	13->60	13->61	13->77	13->93	13->95	13->98	13-?13	13-?41
	13-?42	13-?77	13-?95	13-?97	13-@00	13-@13	13-@24	13-@26	13-@40	13-@56	13-@58	13-@61	13-@74	13-@75
	13-@85	13-A01	13-A16	13-A18	13-A21	13-A43	13-A44	13-A45	13-A54	13-A55	13-A71	13-A85	13-A87	13-A90
	13-B93	13-B99	13-C04	13-C05	13-C39	13-C40	13-C44	13-C45	13-D01	13-D02	13-D06	13-D07	13-D64	13-D68
	13-D69	13-D92	13-E17	13-E41	13-E72	13-E73	13-F08	13-F09	13-F35	13-F36	13-G28	13-G29	13-G66	13-G67
	13-H78	13-H79	13-I03	13-I04	13-I54	13-I55	13-I84	13-I85	13-J93	13-J94	13-K10	13-K11	13-K37	13-K38
	13-K55	13-K74	13-K75	13-K95	13-K96	13-L16	13-L17	13-L55	13-L56	13-M05	13-M06	13-M23	13-M24	13-M65
	13-M69	13-M70	13-N03	13-N11	13-N12	13-N44	13-N45	13-N58	13-N59	13-N87	13-N88	13-N96	13-N97	13-O15
	13-O16	13-O25	13-O26	13-O59	13-O62	13-O63	13-O92	13-O95	13-O96	13-P55	13-P57	13-P58	13-P76	13-P77
	13-Q24	13-Q34	13-Q35	13-R02	13-R09	13-R10	13-R51	13-R55	13-R56	13-R96	13-R99	13-S00	13-S34	13-S46
	13-S47	13-S52	13-S53	13-S81	13-S82	13-S90	13-U67	13-U70	13-U71	13-U99	13-V02	13-V36	13-V39	13-V57
	13-V63	13-V75	13-V90	13-W14	13-W65	13-W71	13-W72	13-X11	13-Y05	13-Y08	13-Y09	13-Y55	13-Y62	13-Y63
	13-Y68	13-Y70	13-Y71	13-Y93	13-Z01	13-Z02	13-Z07	13-Z11	13-Z12	13-Z31	13-Z39	13-Z40	13-Z62	13-Z69
	13-Z70	13-Z75	13-Z76	13-Z81	13-Z82	13-[04	13-[12	13-[13	13-[18	13-[23	13-[24	13-[37	13-[41	13-[42
	13-[61	13-[85	13-[86	13-\20	13-\29	13-\30	13-\35	13-\37	13-\38	13-\61	13-\71	13-\72	13-\77	13-\81
	13-\82	13-]02	13-]12	13-]13	13-]36	13-]45	13-]46	13-]51	13-]52	13-]57	13-]58	13-]81	13-]91	13-]92
	13-]97	13-^02	13-^03	13-^16	13-^20	13-^21	13-^42	13-^69	13-^70	13-_11	13-_20	13-_21	13-_26	13-_28
	13-_29	13-_52	13-_62	13-_63	13-_68	13-_72	13-_73	13-_96	13-`06	13-`07	13-`31	13-`40	13-`41	13-`47
	13-`50	13-`55	13-`56	13-`62	13-`63	13-`64	13-`68	13-`69	13-`92	13-a02	13-a03	13-a08	13-a13	13-a14
	13-a27	13-a31	13-a32	13-a52	13-a61	13-a62	13-a68	13-a71	13-a76	13-a77	13-a87	13-a88	13-a98	13-b02
	13-b03	13-b24	13-b40	13-b41	13-b66	13-b67	13-c14	13-c23	13-c24	13-c29	13-c30	13-c34	13-c35	13-c42
	13-c43	13-c58	13-c84	13-c93	13-c94	13-c99	13-d03	13-d04	13-d24	13-d33	13-d34	13-d39	13-d43	13-d44
	13-d51	13-d52	13-d69	13-d70	13-d77	13-d78	13-d96	13-d97	13-e16	13-e25	13-e26	13-e43	13-e44	13-e76
	13-f03	13-f04	13-f09	13-f13	13-f14	13-f21	13-f22	13-f48	13-f72	13-f73	13-f78	13-f79	13-f90	13-f91
	13-g07	13-g08	13-g86	13-h10	13-h11	13-h30	13-h54	13-h55	13-h60	13-h64	13-h65	13-h71	13-h72	13-h77
	13-h81	13-h82	13-h89	13-h90	13-i18	13-i40	13-i41	13-i47	13-i48	13-i52	13-i53	13-i80	13-j02	13-j03
	13-j09	13-j10	13-j14	13-j15	13-j20	13-j24	13-j25	13-j30	13-j34	13-j35	13-j42	13-j43	13-j64	13-j88
	13-j89	13-j95	13-j99	13-k00	13-k07	13-k08	13-k22	13-k26	13-k27	13-k34	13-k35	13-k99	13-l39	13-l40
	13-l52	13-l53	13-l95	13-l96	13-m09	13-m10	13-m28	13-m29	16-15	16-16	17-14			
DPE	4-882#	13-898	13-919	13-:15										
DPEHI	4-954#													
DPELO	4-955#													
DPR	4-757#	13-T61	13-T63											
DRIVES	11-3	32-16#												
DRQ	4-842#	13-T37	13-U80											
DRVCLR	4-702#	13-F67	13-J30											
DRY	4-758#	13-G52	13-G54	13-G89	13-G94									
DSWR	4-682#	6-0	9-23											
DTE	4-767#	4-781	13-703	13-733										
DULPRT	4-845#	13-526	13-531	13-536										
DVA	4-688#	9-90	13-550	13-552	13-T52	13-T54								
DVC	4-881#	13-152	13-959	13-961	13-?17	13-?40	13-?44	13-?81	13-?85					
EBL	4-821#	13-A36	13-A47	13-A57	13-A59	13-A74	13-A76	13-d54	13-d59	13-d85	13-f24	13-f29	13-h92	13-h97
	13-j45	13-j50	13-k39	13-k44										
ECH	4-773#	4-781	13-695	13-724	33-77	33-78	33-85	33-86						
ECI	4-850#													
ECRC	4-825#													

```
EH1     37-1    41-1#
EH115   37-9    41-14#
EH130   37-10   41-16#
EH132   37-11   41-18#
EH142   37-12   41-20#
EH145   37-13   41-22#
EH150   37-14   41-25#
EH2     37-2    41-3#
EH213   37-15   41-27#
EH220   37-16   41-29#
EH3     37-8    41-4#
EH5     37-3    41-5#
EH57    37-5    41-8#
EH65    37-6    41-10#
EH7     37-4    41-7#
EH71    37-7    41-12#
EHT1     8-3     8-49    8-52    8-55    8-58    8-61    8-64    8-70    8-73    8-76    8-79    8-82    8-85    8-88
         8-91    8-94    8-97    8-100   8-106   8-109   8-112   8-115   8-118   8-121   8-124   8-127   8-130   8-133
         8-136   8-139   8-145   8-148   8-151   8-154   8-157   8-166   8-169   8-184   8-187   8-190   8-196   8-199
         8-205   8-208   8-214   8-217   8-220   8-223   8-226   8-229   8-235   8-238   8-244   8-247   8-253   8-256
         8-259   8-277   8-280   8-283   8-286   8-298   8-301   8-307   8-310   8-316   8-319   8-322   8-325   8-328
         8-334   8-340   8-343   8-346   8-349   8-355   8-358   8-367   8-376   8-379   8-385   8-388   8-391   8-394
         8-397   8-424   8-427   8-430   8-445   8-448   8-451   8-454   8-457   8-460   8-473   8-479   8-485   8-488
         8-492   8-495   8-498   8-501   8-504   8-508   8-511   8-515   8-518   8-521   8-524   8-527   8-530   8-533
         8-537   8-540   8-543   8-546   8-549   8-552   8-555   8-558   8-561   8-564   8-567   8-570   8-573   8-576
         8-580   8-583   8-586   8-589   37-1#
EHT115   8-232   8-241   8-250   8-262    8-409   8-412   8-476   37-9#
EHT130   8-265   8-268   37-10#
EHT132   8-271   8-274   8-289   37-11#
EHT142   8-292   8-295   8-415   37-12#
EHT145   8-304   37-13#
EHT150   8-313    8-331   8-337   8-361   8-364   8-370   8-400   8-406   8-467   8-470   8-482   37-14#
EHT2     8-6     37-2#
EHT213   8-418   8-421   8-464   37-15#
EHT220   8-433   8-436   8-439    8-442   37-16#
EHT5     8-15    8-18    37-3#
EHT57    8-142   37-5#
EHT65    8-160    8-163   37-6#
EHT7     8-22    37-4#
EHT71    8-172    8-175   8-178   8-193   8-202   8-211   37-7#
EHT74    8-181   37-8#
EMS1    36-3    36-4    40-3#
EMS10   36-129   36-131   40-10#
EMS11   36-133   40-11#
EMS12   36-118   36-120   36-122   40-13#
EMS13   36-136   36-138   36-140   36-142   40-14#
EMS14   36-144   36-146   36-150   40-15#
EMS15   36-144   36-146   36-148   36-207   36-327   36-329   36-346   36-348   36-364   36-380   36-400   40-16#
EMS16   36-151   36-153   36-155   36-337   40-17#
EMS17   36-151   36-153   36-157   40-18#
EMS2    36-8    36-187   36-211   36-239   36-243   36-248   36-259   36-339   36-344   40-4#
EMS20   36-158   36-160   36-162   36-164   36-166   36-168   36-172   40-19#
EMS21   36-153   36-160   36-164   36-166   36-168   36-170   36-350   36-366   36-382   36-402   40-20#
EMS22   36-160   36-162   36-166   36-168   40-21#
EMS23   36-173   36-175   36-179   40-22#
EMS24   36-173   36-175   36-177   36-359   36-375   36-391   36-417   40-23#
EMS25   36-180   36-182   36-186   40-24#
```

EMS250	36-33	36-36	36-39	36-75	36-197	36-199	36-201	36-203	36-271	36-276	36-291	36-296	36-317	40-63#
EMS251	36-41	36-44	36-47	36-302	36-313	40-64#								
EMS252	36-49	36-52	36-54	36-56	36-58	36-60	36-62	36-64	36-66	36-221	36-223	36-341	40-65#	
EMS253	36-49	36-79	36-81	36-83	36-85	36-88	36-91	36-94	36-221	36-223	40-66#			
EMS254	36-49	36-142	36-148	36-155	36-170	36-177	36-184	36-194	40-67#					
EMS255	36-68	36-70	36-73	36-75	36-205	36-213	36-271	36-285	36-291	36-298	36-422	40-68#		
EMS256	36-96	36-98	36-101	36-104	36-271	36-274	40-69#							
EMS257	36-106	40-70#												
EMS26	36-180	36-182	36-184	36-353	36-369	36-385	36-415	40-25#						
EMS260	36-110	36-113	36-116	36-133	40-71#									
EMS261	36-209	40-72#												
EMS262	36-302	36-315	36-321	36-323	40-73#									
EMS27	36-187	36-189	36-192	36-196	36-287	36-404	36-405	36-407	36-419	40-26#				
EMS3	36-77	36-229	36-241	36-245	36-265	36-269	36-289	36-292	36-294	36-337	40-5#			
EMS30	36-191	36-192	36-194	36-201	36-203	36-213	36-217	40-27#						
FMS300	36-3	36-8	36-52	36-54	36-56	36-77	36-81	36-83	36-118	36-124	36-127	36-136	36-187	36-211
	36-239	36-243	36-248	36-259	36-271	36-274	36-276	36-339	36-344	36-422	40-75#			
EMS301	36-4	36-6	36-10	36-13	36-15	36-33	36-36	36-39	36-41	36-44	36-47	36-49	36-58	36-60
	36-62	36-64	36-66	36-68	36-70	36-73	36-75	36-85	36-88	36-91	36-94	36-98	36-101	36-104
	36-110	36-113	36-116	36-133	36-142	36-148	36-155	36-170	36-177	36-184	36-194	36-225	36-267	40-76#
EMS302	36-4	36-6	36-8	36-15	40-77#									
EMS303	36-4	36-133	40-78#											
EMS304	36-4	40-79#												
EMS306	36-6	36-13	36-49	36-225	36-267	36-309	36-311	40-80#						
EMS307	36-8	36-187	36-211	36-239	36-243	36-248	36-259	36-339	36-344	40-81#				
EMS31	36-197	40-28#												
EMS310	36-10	36-219	36-323	40-82#										
EMS311	36-10	36-13	40-83#											
EMS312	36-12	36-35	36-38	36-43	36-46	36-69	36-72	36-87	36-90	36-93	36-100	36-103	36-112	36-115
	40-84#													
EMS313	36-15	36-39	36-47	36-66	36-73	36-94	36-104	36-116	36-142	36-148	36-155	36-170	36-177	36-184
	36-194	40-85#												
EMS314	36-17	36-21	36-25	36-29	36-51	40-86#								
EMS315	36-19	36-23	36-27	36-31	36-160	36-191	40-87#							
EMS316	36-17	36-19	36-21	36-23	36-25	36-27	36-29	36-31	40-88#					
EMS317	36-17	36-19	40-89#											
EMS32	36-199	40-29#												
EMS320	36-21	36-23	40-90#											
EMS321	36-25	36-27	40-91#											
EMS322	36-29	36-31	40-92#											
EMS323	36-36	36-44	36-64	36-70	36-75	36-91	36-101	36-113	40-93#					
EMS324	36-33	36-41	36-58	36-60	36-62	36-68	36-85	36-88	36-98	36-110	40-94#			
EMS325	36-38	36-46	36-72	36-93	36-103	36-115	40-95#							
EMS326	36-35	36-43	36-69	36-87	36-90	36-100	36-112	40-96#						
EMS327	36-49	36-75	36-127	36-221	36-256	36-271	36-271	36-291	36-292	36-302	40-97#			
EMS33	36-201	40-30#												
EMS330	36-51	40-98#												
EMS331	36-79	36-96	36-189	36-205	40-99#									
EMS332	36-106	36-150	36-157	36-172	36-179	36-186	36-196	36-333	36-362	36-378	36-396	36-413	40-100#	
EMS333	36-108	36-126	36-129	36-162	36-168	36-247	36-258	36-261	36-307	36-313	36-315	36-317	36-323	40-101#
EMS334	36-131	36-162	36-166	36-252	36-302	36-307	40-102#							
EMS335	36-120	36-129	36-252	36-254	36-254	36-261	36-304	36-352	36-361	36-368	36-377	36-384	36-393	40-103#
EMS336	36-122	36-131	36-348	36-353	36-364	36-369	36-380	36-385	36-400	36-404	36-415	36-419	40-104#	
EMS337	36-140	36-146	36-153	36-164	36-166	36-175	36-182	36-192	36-217	36-221	36-231	36-250	36-252	36-254
	36-256	36-263	36-279	36-291	36-300	36-305	40-105#							
EMS34	36-203	36-265	40-31#											
EMS340	36-129	36-131	36-261	36-348	36-353	36-364	36-369	36-380	36-385	36-400	36-415	40-106#		

EMS501	36-5	36-7	36-9	36-11	36-14	36-16	36-18	36-20	36-22	36-24	36-26	36-28	36-30	36-32
	36-42	36-45	36-48	36-50	36-53	36-55	36-57	36-59	36-61	36-63	36-65	36-67	36-76	36-78
	36-80	36-82	36-84	36-86	36-89	36-92	36-95	36-97	36-99	36-102	36-105	36-107	36-109	36-111
	36-114	36-117	36-119	36-121	36-123	36-125	36-128	36-130	36-132	36-134	36-145	36-147	36-152	36-154
	36-159	36-161	36-163	36-165	36-167	36-169	36-174	36-176	36-181	36-183	36-208	36-212	36-220	36-222
	36-224	36-226	36-232	36-234	36-246	36-251	36-253	36-255	36-257	36-260	36-262	36-264	36-266	36-268
	36-270	36-273	36-275	36-280	36-282	36-284	36-286	36-288	36-293	36-295	36-303	36-306	36-308	36-310
	36-312	36-314	36-316	36-318	36-320	36-322	36-324	36-326	36-328	36-330	36-332	36-334	36-336	36-338
	36-343	36-345	36-347	40-150#										
EMS502	36-5	36-7	36-9	36-11	36-14	36-16	36-18	36-20	36-22	36-24	36-26	36-28	36-30	36-32
	36-50	36-130	36-132	36-134	36-290	36-340	36-343	36-412	40-151#					
EMS503	36-5	36-9	36-11	36-14	36-16	36-50	36-57	36-63	36-76	36-82	36-86	36-137	36-139	36-141
	36-143	36-145	36-147	36-149	36-152	36-154	36-156	36-159	36-165	36-171	36-174	36-176	36-178	36-181
	36-183	36-185	36-188	36-190	36-193	36-195	36-198	36-200	36-210	36-226	36-228	36-230	36-236	36-238
	36-240	36-242	36-244	36-246	36-249	36-251	36-260	36-262	36-290	36-322	36-336	36-338	36-340	36-343
	36-345	36-347	36-349	36-351	36-354	36-356	36-358	36-360	36-363	36-365	36-367	36-370	36-372	36-374
	36-376	36-379	36-381	36-383	36-386	36-388	36-390	36-392	36-395	36-397	36-399	36-401	36-403	36-406
	36-408	36-410	36-412	36-414	36-416	36-418	36-421	40-152#						
EMS504	36-11	36-14	36-16	36-34	36-37	36-40	36-55	36-61	36-71	36-74	36-76	36-84	36-89	36-128
	36-198	36-200	36-202	36-204	36-206	36-214	36-216	36-218	36-268	36-277	36-286	36-288	36-293	36-297
	36-299	36-301	36-423	40-153#										
EMS505	36-7	40-154#												
EMS506	36-107	40-155#												
EMS507	36-109	40-156#												
EMS51	36-267	36-283	40-44#											
EMS510	36-290	40-157#												
EMS511	36-5	36-7	36-9	36-11	36-14	36-16	36-18	36-20	36-22	36-24	36-26	36-28	36-30	36-32
	36-34	36-37	36-40	36-42	36-45	36-48	36-50	36-53	36-55	36-57	36-59	36-61	36-63	36-65
	36-67	36-71	36-74	36-76	36-78	36-80	36-82	36-84	36-86	36-89	36-92	36-95	36-97	36-99
	36-102	36-105	36-107	36-109	36-111	36-114	36-117	36-119	36-121	36-123	36-125	36-128	36-130	36-132
	36-134	36-137	36-139	36-141	36-143	36-145	36-147	36-149	36-152	36-154	36-156	36-159	36-161	36-163
	36-165	36-167	36-169	36-171	36-174	36-176	36-178	36-181	36-183	36-185	36-188	36-190	36-193	36-195
	36-198	36-200	36-202	36-204	36-206	36-208	36-210	36-212	36-214	36-216	36-218	36-220	36-222	36-224
	36-226	36-228	36-230	36-232	36-234	36-236	36-238	36-240	36-242	36-244	36-246	36-249	36-251	36-253
	36-255	36-257	36-260	36-262	36-264	36-266	36-268	36-270	36-273	36-275	36-277	36-280	36-282	36-284
	36-286	36-288	36-290	36-293	36-295	36-297	36-299	36-301	36-303	36-306	36-308	36-310	36-312	36-314
	36-316	36-318	36-320	36-322	36-324	36-326	36-328	36-330	36-332	36-334	36-336	36-338	36-340	36-343
	36-345	36-347	36-349	36-351	36-354	36-356	36-358	36-360	36-363	36-365	36-367	36-370	36-372	36-374
	36-376	36-379	36-381	36-383	36-386	36-388	36-390	36-392	36-395	36-397	36-399	36-401	36-403	36-406
	36-408	36-410	36-412	36-414	36-416	36-418	36-421	36-423	40-158#					
EMS52	36-279	36-281	36-285	36-287	40-45#									
EMS53	36-289	36-411	40-46#											
EMS54	36-291	36-294	36-296	36-298	40-47#									
EMS55	36-300	40-48#												
EMS56	36-302	36-304	36-305	36-341	40-49#									
EMS57	36-307	40-51#												
EMS6	36-124	36-127	36-129	36-131	40-8#									
EMS60	36-307	40-52#												
EMS600	36-272	36-274	36-276	36-422	40-161#									
EMS601	36-279	40-162#												
EMS602	36-281	36-283	40-163#											
EMS603	36-319	40-164#												
EMS604	36-348	36-350	36-353	36-355	36-357	36-359	36-362	40-165#						
EMS605	36-364	36-366	36-369	36-371	36-373	36-375	36-378	40-166#						
EMS606	36-380	36-382	36-385	36-387	36-389	36-391	36-394	36-396	40-167#					
EMS607	36-400	36-402	36-405	36-407	36-409	36-411	36-413	36-415	36-417	36-420	40-168#			
EMS61	36-311	40-53#												

EMS62	36-313	36-315	36-317	36-319	40-54#								
EMS63	36-321	36-323	36-325	36-327	36-329	36-331	36-333	36-335	36-355	36-371	36-387	40-55#	
EMS64	36-337	40-56#											
EMS65	36-339	36-342	40-57#										
EMS66	36-344	36-346	36-348	36-353	36-364	36-369	36-380	36-385	36-398	36-400	36-415	40-58#	
EMS67	36-362	36-378	36-396	36-413	40-59#								
EMS7	36-126	36-127	40-9#										
EMS70	36-394	36-420	40-61#										
EMT1	8-3	36-3#											
EMT10	8-25	36-17#											
EMT100	8-193	36-142#											
EMT101	8-196	36-144#											
EMT102	8-199	36-146#											
EMT103	8-202	36-148#											
EMT104	8-205	36-151#											
EMT105	8-208	36-153#											
EMT106	8-211	36-155#											
EMT107	8-214	36-158#											
EMT11	8-28	36-19#											
EMT110	8-217	36-160#											
EMT111	8-220	36-162#											
EMT112	8-223	36-164#											
EMT113	8-226	36-166#											
EMT114	8-229	36-168#											
EMT115	8-232	36-170#											
EMT116	8-235	36-173#											
EMT117	8-238	36-175#											
EMT12	8-31	36-21#											
EMT120	8-241	36-177#											
EMT121	8-244	36-180#											
EMT122	8-247	36-182#											
EMT123	8-250	36-184#											
EMT124	8-253	36-187#											
EMT125	8-256	36-189#											
EMT126	8-259	36-192#											
EMT127	8-262	36-194#											
EMT13	8-34	36-23#											
EMT130	8-265	36-197#											
EMT131	8-268	36-199#											
EMT132	8-271	36-201#											
EMT133	8-274	36-203#											
EMT134	8-277	36-205#											
EMT135	8-280	36-207#											
EMT136	8-283	36-209#											
EMT137	8-286	36-211#											
EMT14	8-37	36-25#											
EMT140	8-289	36-213#											
EMT141	8-292	36-215#											
EMT142	8-295	36-217#											
EMT143	8-298	36-219#											
EMT144	8-301	36-221#											
EMT145	8-304	36-223#											
EMT146	8-307	36-225#											
EMT147	8-310	36-227#											
EMT15	8-40	36-27#											
EMT150	8-313	36-229#											
EMT151	8-316	36-231#											

EMT152	8-319	36-233#
EMT153	8-322	36-235#
EMT154	8-325	36-237#
EMT155	8-328	36-239#
EMT156	8-331	36-241#
EMT157	8-334	36-243#
EMT16	8-43	36-29#
EMT160	8-337	36-245#
EMT161	8-340	36-248#
EMT162	8-343	36-250#
EMT163	8-346	36-252#
EMT164	8-349	36-254#
EMT165	8-352	36-256#
EMT166	8-355	36-259#
EMT167	8-358	36-261#
EMT17	8-46	36-31#
EMT170	8-361	36-263#
EMT171	8-364	36-265#
EMT172	8-367	36-267#
EMT173	8-370	36-269#
EMT174	8-373	36-271#
EMT175	8-376	36-274#
EMT176	8-379	36-276#
EMT177	36-278#	
EMT2	8-6	36-4#
EMT20	8-49	36-33#
EMT200	8-385	36-279#
EMT201	8-388	36-281#
EMT202	8-391	36-283#
EMT203	8-394	36-285#
EMT204	8-397	36-287#
EMT205	8-400	36-289#
EMT206	8-403	36-291#
EMT207	8-406	36-294#
EMT21	8-52	36-36#
EMT210	8-409	36-296#
EMT211	8-412	36-298#
EMT212	8-415	36-300#
EMT213	8-418	36-302#
EMT214	8-421	36-305#
EMT215	8-424	36-307#
EMT216	8-427	36-309#
EMT217	8-430	36-311#
EMT22	8-55	36-39#
EMT220	8-433	36-313#
EMT221	8-436	36-315#
EMT222	8-439	36-317#
EMT223	8-442	36-319#
EMT224	8-445	36-321#
EMT225	8-448	36-323#
EMT226	8-451	36-325#
EMT227	8-454	36-327#
EMT23	8-58	36-41#
EMT230	8-457	36-329#
EMT231	8-460	36-331#
EMT232	8-464	36-333#
EMT233	8-467	36-335#

SEQ 0305

EMT234	8-470	36-337#	
EMT235	8-473	36-339#	
EMT236	8-476	36-341#	
EMT237	8-479	8-482	36-344#
EMT24	8-61	36-44#	
EMT240	36-346#		
EMT241	8-485	36-348#	
EMT242	8-488	36-350#	
EMT243	8-492	36-353#	
EMT244	8-495	36-355#	
EMT245	8-498	36-357#	
EMT246	8-501	36-359#	
EMT247	8-504	36-362#	
EMT25	8-64	36-47#	
EMT250	8-508	36-364#	
EMT251	8-511	36-366#	
EMT252	8-515	36-369#	
EMT253	8-518	36-371#	
EMT254	8-521	36-373#	
EMT255	8-524	36-375#	
EMT256	8-527	36-378#	
EMT257	8-530	36-380#	
EMT26	8-67	36-49#	
EMT260	8-533	36-382#	
EMT261	8-537	36-385#	
EMT262	8-540	36-387#	
EMT263	8-543	36-389#	
EMT264	8-546	36-391#	
EMT265	8-549	36-394#	
EMT266	8-552	36-396#	
EMT267	8-555	36-398#	
EMT27	8-70	36-52#	
EMT270	8-558	36-400#	
EMT271	8-561	36-402#	
EMT272	8-564	36-405#	
EMT273	8-567	36-407#	
EMT274	8-570	36-409#	
EMT275	8-573	36-411#	
EMT276	8-576	36-413#	
EMT277	8-580	36-415#	
EMT3	8-9	36-6#	
EMT30	8-73	36-54#	
EMT300	8-583	36-417#	
EMT301	8-586	36-420#	
EMT302	8-589	36-422#	
EMT31	8-76	36-56#	
EMT32	8-79	36-58#	
EMT33	8-82	36-60#	
EMT34	8-85	36-62#	
EMT35	8-88	36-64#	
EMT36	8-91	36-66#	
EMT37	8-94	36-68#	
EMT4	8-12	36-8#	
EMT40	8-97	36-70#	
EMT41	8-100	36-73#	
EMT42	8-103	36-75#	
EMT43	8-106	36-77#	

SEQ 0306

```
EMT44    8-109   36-79#
EMT45    8-112   36-81#
EMT46    8-115   36-83#
EMT47    3-118   36-85#
EMT5     8-15    36-10#
EMT50    8-121   36-88#
EMT51    8-124   36-91#
EMT52    8-127   36-94#
EMT53    8-130   36-96#
EMT54    8-133   36-98#
EMT55    8-136   36-101#
EMT56    8-139   36-104#
EMT57    8-142   36-106#
EMT6     8-18    36-13#
EMT60    8-145   36-108#
EMT61    8-148   36-110#
EMT62    8-151   36-113#
EMT63    8-154   36-116#
EMT64    8-157   36-118#
EMT65    8-160   36-120#
EMT66    8-163   36-122#
EMT67    8-166   36-124#
EMT7     8-22    36-15#
EMT70    8-169   36-127#
EMT71    8-172   36-129#
EMT72    8-175   36-131#
EMT73    8-178   36-133#
EMT74    8-181   36-135#
EMT75    8-184   36-136#
EMT76    8-187   36-138#
EMT77    8-190   36-140#
EMTVEC   4-682#   9-23*   9-23*
ENRGDT   35-144#
EQUALS   32-3#
ERR      4-751#  13-D96  13-E00  13-E06  13-E22  13-E47  13-E48  13-L63
ERRNMB   26-48*  26-49*  26-52   26-57   26-64   26-160#
ERROR    4-682#
ERRTYP   25-1    26-13#
ERRVEC   4-682#   9-23    9-23*   9-23*   9-25*   9-26*  13-<24  13-<25  13-<26*  13-<27*  13-<33*  13-<34*  13-<38*  13-<39*
         13-<67  13-<68  13-<69*  13-<70*  13-<90*  13-<91*  13-<98*  13-<99*  15-6     15-7     15-8*    15-9*    15-25*   15-45*
         15-46*  24-1     24-1    24-1*   24-1*   24-1*   24-1*   24-1*   25-1     25-1*    25-1*
ERTY00   26-21   26-165#
ERTY01   26-46   26-166#
ERTY02   26-51   26-167#
ERTY03   26-53   26-168#
ERTY04   26-152  26-169#
ERTY05   26-39   26-170#
ESRC     4-823#  13-a85  13-a92  13-c45  13-c50  13-k10  13-k15
F0       4-693#
F1       4-692#
F2       4-691#
F3       4-690#
F4       4-689#
FER      4-775#   4-781  13-695  13-724
FMT16    4-849#  13-870  13-B25  13-B43  13-B63  13-B65  13-C65  13-C67  13-D24  13-D26  13-D77  13-D79  13-E36  13-P51
        13-S65  13-S67
FNCDTB   13-K14  13-M75  13-N20  13-R17  33-55#
```

	13-G68	13-H88*	13-H89	13-I05	13-I72*	13-K09*	13-K48*	13-K72*	13-K93*	13-L14*	13-L83*	13-M20*	13-M68*	13-N06*
	13-N55*	13-N94*	13-O23*	13-O60*	13-O93*	13-P56*	13-P85*	13-Q28*	13-R05*	13-R52*	13-R97*	13-S41*	13-S87*	13-T51
	13-T77*	13-T85	13-U06*	13-U35*	13-U68*	13-V42*	13-W48	13-W68*	13-X17*	13-Y06*	13-Y58*	13-Y96*	13-Z15	13-Z34*
	13-Z65*	13-[07*	13-[29	13-[45	13-[64*	13-\25*	13-\66*	13-\85	13-]07*	13-]41*	13-]86*	13-^08	13-^24	13-^47*
	13-_16*	13-_57*	13-_76	13-`01*	13-`36*	13-`97*	13-a19	13-a35	13-a57*	13-b06	13-b29*	13-c19*	13-c89*	13-d29*
	13-e00	13-e21*	13-e81*	13-f53*	13-g91*	13-h35*	13-i21*	13-i83*	13-j69*	13-l02*	16-19*	17-17*		
RMCS1I	7-0#	13-85*	13-93*	13-119*	13-124*	13-126	13-180*	13-189*	13-193*	13-199	13-229*	13-237*	13-242	
RMCS1O	7-0#	13-k62*	13-l02											
RMCS2	4-970#	9-85*	9-86*	9-88	11-57*	11-58*	13-8	13-15	13-25	13-29	13-36	13-44	13-56	13-:24*
	13-<29	18-10*	18-11*											
RMCS2I	7-0#													
RMCS2O	7-0#	13-:00*	13-:01*	13-:24	13-:30	13-:39	13-:42*							
RMCS3	4-973#													
RMCS3I	7-0#													
RMCS3O	7-0#													
RMDA	4-893#	13-79*	13-86	13-166*	13-173*	13-181	13-216*	13-222*	13-230	13-263*	13-270	13-384*	13-390	13-406*
	13-412	13-467*	13-471	13-484*	13-488	13-503*	13-568*	13-569*	13-579*	13-580*	13-595*	13-596*	13-:82*	13-:84*
	13-:86	13-;03*	13-;04*	13-;05	13-;10	13-;14*	13-;15*	13-;16	13-;21	13-;28*	13-;29*	13-;30	13-;35	13-B20*
	13-B79	13-B96*	13-C06	13-C38*	13-C48	13-D00*	13-D49*	13-N50*	13-N64	13-O19*	13-O31	13-O35	13-P52*	13-Q19*
	13-Q97*	13-R47*	13-R93*	13-S38*	13-T21	13-U43*	13-W45	13-Y03*	13-\21*	13-\62*	13-]03*	13-]37*	13-]82*	13-^43*
	13-_12*	13-_53*	13-_97*	13-`32*	13-`93*	13-a53*	13-b25*	13-c18*	13-c87*	13-d27*	13-e19*	13-e79*	13-f51*	13-g89*
	13-h34*	13-i12*	13-i74*	13-j67*	13-k87*									
RMDAI	7-0#	13-86*	13-95	13-181*	13-194*	13-200	13-230*	13-243	13-270*	13-279	13-390*	13-393	13-412*	13-415
	13-471*	13-473	13-488*	13-491	13-:86*	13-:88								
RMDAO	7-0#	13-C27*	13-C38	13-C60*	13-C62*	13-C84*	13-C85*	13-D00	13-D27*	13-D37*	13-D38*	13-D49	13-D80*	13-P41*
	13-P42*	13-P52	13-R37*	13-R38*	13-R47	13-R67	13-R73	13-R75*	13-R76	13-R78*	13-R80*	13-R81	13-S25*	13-S26*
	13-S38	13-S68*	13-k58*	13-k59*	13-k87	13-l88								
RMDB	4-971#	13-60												
RMDBI	7-0#													
RMDBO	7-0#													
RMDC	4-902#	13-80*	13-87	13-176*	13-184	13-218*	13-224*	13-232	13-265*	13-272	13-369*	13-373	13-429*	13-433
	13-447*	13-453	13-485*	13-499*	13-505	13-:83*	13-:85*	13-:87	13-;47	13-;50*	13-;51	13-;59*	13-;60*	13-;61
	13-;69*	13-;70*	13-;71	13-;87*	13-;88*	13-;91	13-C82	13-C89*	13-D08	13-D48*	13-N51*	13-N67	13-O20*	13-O40
	13-O44	13-O56*	13-P30*	13-Q20*	13-Q98*	13-R46*	13-R92*	13-S37*	13-Y04*	13-\22*	13-\63*	13-]04*	13-]38*	13-]83*
	13-^44*	13-_13*	13-_54*	13-_98*	13-`33*	13-`94*	13-a54*	13-b26*	13-c17*	13-c88*	13-d28*	13-e20*	13-e80*	13-i52*
	13-g90*	13-h33*	13-i11*	13-i75*	13-j68*	13-k88*								
RMDCI	7-0#	13-87*	13-97*	13-184*	13-191*	13-197*	13-203	13-232*	13-239*	13-245	13-272*	13-289	13-373*	13-376*
	13-377	13-433*	13-437*	13-438	13-453*	13-459*	13-460	13-505*	13-509	13-510	13-:87*	13-:90*	13-:91	
RMDCO	7-0#	13-R87*	13-R92	13-S11	13-S17*	13-S18	13-k57*	13-k88	13-l35					
RMDS	4-894#	9-87	9-110	13-469*	13-501*	13-=81	13-=86	13-=97	13->09	13->18	13->46	13->52	13->62	13->79
	13-a71	13-a76	13-a86	13-A03	13-D34	13-D52	13-D70	13-D88	13-D95	13-E05	13-E21	13-E46	13-G53	13-G58
	13-G93	13-G96	13-K78	13-K85	13-K99	13-L06	13-L61	13-L62	13-L67	13-L99	13-M09	13-M27	13-O51	13-O68
	13-O77	13-P01	13-P21	13-P31	13-P44	13-P59	13-T34	13-T43	13-T60	13-T65	13-T88	13-U17	13-U46	13-V12
	13-V16	13-V43	13-V48	13-V64	13-V68	13-V76	13-V80	13-V97	13-W02	13-W22	13-W29	13-W77	13-W81	13-Z87
	13-Z92	13-]63	13-]68	13-`74	13-`79	16-20	16-24	17-18	17-23					
RMDSI	7-0#													
RMDSO	7-0#													
RMDT	4-899#	9-93	11-59	13-448*	13-523	13-541	13-T33	13-U76	26-26					
RMDTI	7-0#													
RMDTO	7-0#													
RMEC1	4-906#	13-450*												
RMEC1I	7-0#													
RMEC1O	7-0#													
RMEC2	4-907#	13-62	13-411*	13-<40										
RMEC2I	7-0#													
RMEC2O	7-0#													
RMER1	4-895#	13-112*	13-120	13-142*	13-147	13-167*	13-174*	13-182	13-225*	13-233	13-266*	13-273	13-368*	13-372

	13-428*	13-432	13-445*	13-451	13-483*	13-487	13-502*	13-682*	13-685	13-691	13-699	13-707	13-712*	13-713*
	13-714	13-720	13-728	13-737	13-743*	13-744*	13-745	13-750	13-762*	13-763	13-774	13-:09	13-:14	13-:26
	13-<11	13-<16	13-<46	13-?15*	13-?25	13-?30	13-?58	13-?63	13-B94*	13-D65*	13-D93*	13-E03*	13-E18*	13-E20*
	13-E42*	13-E74*	13-F10*	13-G30*	13-H80*	13-I56*	13-J95*	13-K39*	13-K71*	13-K92*	13-L13*	13-L57*	13-L84	13-L88
	13-M07*	13-M73	13-M80	13-N46*	13-N89*	13-O17*	13-O18*	13-P78*	13-Q40	13-R15	13-R22	13-R61	13-R66	13-S05
	13-S10	13-S56	13-S60	13-S75	13-S83*	13-S93	13-V00*	13-V37*	13-V58*	13-V91*	13-V96*	13-W15*	13-X27	13-X33
	13-Y14*	13-Y56*	13-Y76	13-Y81	13-Y94*	13-Z32*	13-Z45	13-Z49	13-Z63*	13-[05*	13-[62*	13-\23*	13-\43	13-\48
	13-\64*	13-]05*	13-]18	13-]22	13-]39*	13-]84*	13-^45*	13-_14*	13-_34	13-_39	13-_55*	13-_99*	13-`12	13-`16
	13-`34*	13-`95*	13-a55*	13-b27*	13-c15*	13-c60	13-c68	13-c85*	13-d07	13-d12	13-d25*	13-e17*	13-e77*	13-f49*
	13-g87*	13-h14	13-h19	13-h31*	13-i19*	13-i58	13-i63	13-i81*	13-j65*	13-l00*	16-17*	17-15*		
RMER1I	7-0#	13-120*	13-128	13-147*	13-150*	13-182*	13-195*	13-201	13-233*	13-246	13-273*	13-295	13-372*	13-374
	13-432*	13-434*	13-435	13-451*	13-454	13-487*	13-489	13-685*	13-686	13-694	13-702	13-714*	13-715	13-723
	13-732													
RMER1O	7-0#	13-K66*	13-K71	13-K81	13-K92	13-L02	13-L13	13-L23	13-L33	13-L42*				
RMER2	4-905#	13-113*	13-121	13-143*	13-148	13-169*	13-177*	13-185	13-226*	13-234	13-267*	13-274	13-386*	13-392
	13-408*	13-414	13-449*	13-468*	13-472	13-486*	13-500*	13-506	13-868	13-874*	13-878	13-905*	13-906*	13-907
	13-926*	13-927*	13-928	13-954*	13-955	13-:10	13-?08	13-?14*	13-?16	13-?43	13-?47	13-?78*	13-?80	13-?89
	13-a10	13-a14*	13-a15	13-a25*	13-a27	13-a41*	13-a43	13-B95*	13-D66*	13-D94*	13-E04*	13-E19*	13-E43*	13-E44*
	13-E75*	13-F11*	13-G31*	13-H81*	13-I57*	13-J87	13-J96*	13-J97	13-K12	13-K33	13-K40*	13-K41	13-K56	13-L58*
	13-M08*	13-N47*	13-N90*	13-P79*	13-S84*	13-V01*	13-V38*	13-V59*	13-V92*	13-Y57*	13-Y95*	13-Z33*	13-Z64*	13-[06*
	13-[63*	13-\24*	13-\65*	13-]06*	13-]40*	13-]85*	13-^46*	13-_15*	13-_56*	13-`00*	13-`35*	13-`96*	13-a56*	13-b28*
	13-c16*	13-c86*	13-d26*	13-e18*	13-e78*	13-f50*	13-g88*	13-h32*	13-i20*	13-i82*	13-j66*	13-l01*	16-18*	17-16*
RMER2I	7-0#	13-121*	13-125*	13-130	13-148*	13-152*	13-185*	13-192*	13-198*	13-204	13-234*	13-240*	13-247	13-274*
	13-299	13-392*	13-398*	13-399	13-414*	13-420*	13-421	13-472*	13-475*	13-476	13-506*	13-512*	13-513	13-878*
	13-881	13-889	13-897	13-907*	13-910	13-918	13-928*	13-930	13-938	13-:10*	13-:15			
RMER2O	7-0#													
RMHR	4-903#	13-371*	13-410*	13-562	13-570	13-581	13-597	13-:25*	13-:61*	13-:64				
RMHRI	7-0#	13-:64*												
RMHRO	7-0#	13-:02*	13-:04*	13-:25	13-:37	13-:44*	13-:52*	13-:58*	13-:61	13-:73	42-6			
RMLA	4-897#	13-409*												
RMLAI	7-0#													
RMLAO	7-0#													
RMMR1	4-898#	13-144*	13-149	13-370*	13-=22	13-=27	13-=35*	13-=36*	13-=37	13-=46*	13-=47*	13-=48	13-=61*	13-=62*
	13-=63	13-=85*	13-=95*	13-=96*	13->16*	13->17*	13->51*	13->60*	13->61*	13->77*	13->78*	13-?13*	13-?41*	13-?42*
	13-?77*	13-?79*	13-a13*	13-a24*	13-a26*	13-a40*	13-a42*	13-a74*	13-a75*	13-a85*	13-A01*	13-A02*	13-A30	13-A35
	13-A43*	13-A44*	13-A45*	13-A46	13-A54*	13-A55*	13-A56	13-A71*	13-A72*	13-A73	13-B12	13-B21	13-B93*	13-B99*
	13-C04*	13-C05*	13-C39*	13-C40*	13-C44*	13-C45*	13-D01*	13-D02*	13-D06*	13-D07*	13-D64*	13-D68*	13-D69*	13-D92*
	13-E17*	13-E41*	13-E72*	13-E73*	13-F08*	13-F09*	13-F35*	13-F36*	13-G28*	13-G29*	13-G66*	13-G67*	13-H78*	13-H79*
	13-I03*	13-I04*	13-I48	13-I54*	13-I55*	13-I60	13-I84*	13-I85*	13-I86	13-J93*	13-J94*	13-K10*	13-K11*	13-K37*
	13-K38*	13-K55*	13-K74*	13-K75*	13-K95*	13-K96*	13-L16*	13-L17*	13-L20	13-L27	13-L55*	13-L56*	13-L72	13-L76
	13-M05*	13-M06*	13-M23*	13-M24*	13-M65*	13-M69*	13-M70*	13-N03*	13-N11*	13-N12*	13-N17	13-N27	13-N44*	13-N45*
	13-N58*	13-N59*	13-N87*	13-N88*	13-N96*	13-N97*	13-O15*	13-O16*	13-O25*	13-O26*	13-O59*	13-O62*	13-O63*	13-O92*
	13-O95*	13-O96*	13-P55*	13-P57*	13-P58*	13-P71	13-P76*	13-P77*	13-P86	13-Q24*	13-Q34*	13-Q35*	13-R02*	13-R09*
	13-R10*	13-R51*	13-R55*	13-R56*	13-R96*	13-R99*	13-S00*	13-S34*	13-S46*	13-S47*	13-S52*	13-S53*	13-S81*	13-S82*
	13-T18	13-U67*	13-U70*	13-U71*	13-U99*	13-V02*	13-V36*	13-V39*	13-V57*	13-V63*	13-V75*	13-V90*	13-W14*	13-W65*
	13-W71*	13-W72*	13-X12*	13-X20*	13-X22*	13-X90	13-X94	13-Y05*	13-Y08*	13-Y09*	13-Y16	13-Y24	13-Y55*	13-Y62*
	13-Y63*	13-Y68*	13-Y70*	13-Y71*	13-Y93*	13-Z01*	13-Z02*	13-Z07*	13-Z11*	13-Z12*	13-Z31*	13-Z39*	13-Z40*	13-Z62*
	13-Z69*	13-Z70*	13-Z75*	13-Z76*	13-Z81*	13-Z82*	13-[04*	13-[12*	13-[13*	13-[18*	13-[23*	13-[24*	13-[37*	13-[41*
	13-[42*	13-[61*	13-[85*	13-[86*	13-\20*	13-\29*	13-\30*	13-\35*	13-\37*	13-\38*	13-\61*	13-\71*	13-\72*	13-\77*
	13-\81*	13-\82*	13-]02*	13-]12*	13-]13*	13-]36*	13-]45*	13-]46*	13-]51*	13-]52*	13-]57*	13-]58*	13-]81*	13-]91*
	13-]92*	13-]97*	13-^02*	13-^03*	13-^16*	13-^20*	13-^21*	13-^42*	13-^69*	13-^70*	13-_11*	13-_20*	13-_21*	13-_26*
	13-_28*	13-_29*	13-_52*	13-_62*	13-_63*	13-_68*	13-_72*	13-_73*	13-_96*	13-`06*	13-`07*	13-`31*	13-`40*	13-`41*
	13-`47*	13-`50*	13-`55*	13-`56*	13-`62*	13-`63*	13-`64*	13-`68*	13-`69*	13-`92*	13-a02*	13-a03*	13-a08*	13-a13*
	13-a14*	13-a27*	13-a31*	13-a32*	13-a52*	13-a61*	13-a62*	13-a68*	13-a71*	13-a76*	13-a77*	13-a84	13-a87*	13-a88*
	13-a94	13-a98*	13-b02*	13-b03*	13-b24*	13-b40*	13-b41*	13-b66*	13-b67*	13-c14*	13-c23*	13-c24*	13-c29*	13-c30*
	13-c34*	13-c35*	13-c42*	13-c43*	13-c44	13-c52	13-c58*	13-c84*	13-c93*	13-c94*	13-c99*	13-d03*	13-d04*	13-d24*
	13-d33*	13-d34*	13-d39*	13-d43*	13-d44*	13-d51*	13-d52*	13-d53	13-d61	13-d69*	13-d70*	13-d77*	13-d78*	13-d84

	13-d89	13-d96*	13-d97*	13-e16*	13-e25*	13-e26*	13-e31	13-e36	13-e43*	13-e44*	13-e76*	13-e85	13-e93	13-f03*
	13-f04*	13-f09*	13-f13*	13-f14*	13-f21*	13-f22*	13-f23	13-f3'	13-f48*	13-f57	13-f65	13-f72*	13-f73*	13-f78*
	'3-f79*	13-f90*	13-f91*	13-g07*	13-g08*	,3-g86*	13-g95	13-h03	13-h10*	13-h11*	13-h30*	13-h39	13-h47	13-h54*
	13-h55*	13-h60*	13-h64*	13-h65*	13-h71*	13-h72*	13-h77*	13-h81*	13-h82*	13-h89*	13 ·h90*	13-h91	13-h99	13-i18*
	13-i25	13-i33	13-i40*	13-i41*	13-i47*	13-i48*	13-i52*	13-i53*	13-i80*	13-i87	13-i95	13-j02*	13-j03*	13-j09*
	13-j10*	13-j14*	13-j15*	13-j20*	13-j24*	13-j25*	13-j30*	13-j34*	13-j35*	13-j42*	13-j43*	13-j44	13-j52	13-j64*
	13-j73	13-j81	13-j88*	13-j89*	13-j95*	13-j99*	13-k00*	13-k07*	13-k08*	13-k09	13-k17	13-k22*	13-k26*	13-k27*
	13-k34*	13-k35*	13-k38	13-k46	13-k99*	13-l06	13-l14	13-l39*	13-l40*	13-l52*	13-l53*	13-l95*	13-l96*	13-m09*
	13-m10*	13-m28*	13-m29*	16-15*	16-16*	17-14*								
RMMR1I	7-0#	13-149*	13-154	13-B21*	13-B22	13-B40								
RMMR10	7-0#													
RMMR2	4-904#	13-389*	13-F02	13-F15	13-F41	13-T80	13-T91	13-T95	13-U02	13-U20	13-U24	13-U38	13-U49	13-U53
	13-U77	13-U91	13-[68	13-[74	13-^51	13-^58	13-b48	13-b55	13-e47	13-e63	13-f99	13-g31	13-l22	13-l45
	13-l57	13-m02	13-m14	13-m34										
RMMR2I	7-0#													
RMMR20	7-0#													
RMOF	4-901#	13-81*	13-88	13-168*	13-175*	13-183	13-217*	13-223*	13-231	13-264*	13-271	13-385*	13-391	13-407*
	13-413	13-446*	13-452	13-470*	13-498*	13-504	13-784*	13-785	13-790	13-798	13-802*	13-803*	13-804	13-823*
	13-824*	13-825	13-850*	13-851*	13-852	13-877*	13-953*	13-B18*	13-B19	13-B97*	13-B98	13-C37*	13-C98*	13-C99
	'3-D50*	13-D51	13-E45*	13-N52*	13-N70	13-N82	13-N91*	i3-002	13-084*	13-P09	13-P14	13-P51*	13-Q21*	13-Q99*
	i3-R48*	13-S35*	13-S36	13-k89*										
RMOFI	7-0#	13-88*	13-99*	13-183*	13-190*	13-196*	13-202	13-231*	13-238*	13-244	13-271*	13-283	13-391*	13-395*
	13-396	13-413*	13-417*	13-418	13-452*	13-456*	13-457	13-504*	13-507*	13-804*	13-807	13-815	13-825*	13-828
	13-836													
RMOFO	7-0#	13-870*	13-877	13-953	13-B13*	13-B18	13-B25	13-B43	13-B63	13-B65*	13-B80*	13-B97	13-C37	13-C65
	13-C67*	13-C83*	13-C98	13-D24	13-D26*	13-D36*	13-D50	13-D77	13-D79*	13-E36*	13-E45	13-S24*	13-S35	13-S65
	13-S67*	13-k61*	13-k71*	13-k89	13-l80									
RMR	4-777#	13-687	13-716	13-S94	13-S99	13-T22	13-Y43							
RMSN	4-900#	13-388*	13-431*	13-977	13-981	13-985								
RMSNI	7-0#													
RMSNO	7-0#													
RMWC	4-968#	13-52	13-S39*	13-k95*										
RMWCI	7-0#													
RMWCO	7-0#	13-k63*	13-k95											
RQA	4-863#	13-T81	13-T92	13-U10	13-U21	13-U39	13-U50	13-U78	13-U82	13-[69	13-^52	13-b49	13-e55	13-g00
	13-l33	13-l71												
RQB	4-864#	13-T81	13-T92	13-U10	13-U21	13-U39	13-U50	13-U78	13-U84	13-[69	13-^52	13-b49	13-e55	13-g00
	13-l33	13-l71												
RTC	4-705#	13-H23	13-J42	13-093	13-W97									
SA1	4-742#													
SA16	4-738#													
SA2	4-741#													
SA4	4-740#													
SA8	4-739#													
SADMSK	4-746#													
SAVREG	26-13	29-1#												
SC	4-919#													
SC0	4-795#													
SC1	4-794#													
SC2	4-793#													
SC3	4-792#													
SC4	4-791#													
SCOPE	4-682#	13-1	13-73	13-106	13-137	13-159	13-329	13-521	13-546	13-557	13-610	13-677	13-781	13-794
	13-864	13-973	13-996	13-:49	13-:77	13-:97	13-;41	13-<03	13-=02	13-=19	13-=77	13->42	13-?03	13-@05
	13-@66	13-A26	13-B08	13-B76	13-C78	13-D31	13-D84	13-E64	13-E99	13-G20	13-H70	13-145	13-J84	13-K30
	13-K64	13-L96	13-M56	13-M92	13-N40	13-N79	13-009	13-048	13-074	13-P18	13-P39	13-P66	13-Q10	13-Q87
	13-R35	13-R85	13-S22	13-S72	13-T29	13-T47	13-T70	13-T99	13-U28	13-U58	13-U95	13-V33	13-V53	13-V86

```
TA1      4-735#  13-C28  13-g18
TA16     4-731#  11-63   13-g19
TA2      4-734#  11-63
TA4      4-733#  11-55
TA8      4-732#
TADMSK   4-745#
TAG      4-865#  13-[96  13-[97  13-[98  13-[99  13-\00  13-\01  13-^87  13-^88  13-^89  13-^90  13-^91  13-^92  13-b87
         13-b88  13-b90  13-b91  13-b92  13-b93  13-e55  13-g40  13-g41  13-g43  13-g44  13-g45  13-g46  13-m49  13-m50
         13-m51  13-m52  13-m53  13-m54  13-m60  13-m61  13-m62  13-m63  13-m64  13-m65  13-m92  13-m93  13-m94  13-m95
         13-m96
TAGADR   5-9#    6-0
TAP      4-840#
TBITVE   4-682#
TIME     7-0#    15-57*  15-70*  15-72*
TKVEC    4-682#  27-1*   27-1*
TPVEC    4-682#
TRAPVE   4-682#   9-23*   9-23*
TRE      4-920#
TRTVEC   4-682#
TST      4-866#  13-F16  13-F18  13-F19  13-F42  13-F69  13-F77  13-F85  13-F93  13-G01  13-G09  13-[69  13-^52  13-b49
         13-g00  13-L33  13-L71
TST1     13-1#   24-1
TST10    13-546# 24-1
TST100   13-U28# 24-1
TST101   13-U58# 24-1
TST102   13-U95# 24-1
TST103   13-V33# 24-1
TST104   13-V53# 24-1
TST105   13-V86# 24-1
TST106   13-W07# 24-1
TST107   13-W56# 24-1
TST11    13-557# 24-1
TST110   13-X01# 24-1
TST111   13-X71# 24-1
TST112   13-Y48# 24-1
TST113   13-\10# 24-1
TST114   13-_01# 24-1
TST115   13-`07# 24-1
TST116   13-c75# 24-1
TST117   13-g77# 24-1
TST12    13-610# 24-1
TST120   13-k51# 24-1
TST13    13-677# 24-1
TST14    13-781# 24-1
TST15    13-794# 24-1
TST16    13-864# 24-1
TST17    13-973# 24-1
TST2     13-73#  24-1
TST20    13-996# 24-1
TST21    13-:49# 24-1
TST22    13-:77# 24-1
TST23    13-:97# 24-1
TST24    13-;41# 24-1
TST25    13-<03# 24-1
TST26    13-=02# 24-1
TST27    13-=19# 24-1
TST3     13-106# 24-1
```

SEQ 0316

TST30	13-=77#	24-1												
TST31	13->42#	24-1												
TST32	13-?03#	24-1												
TST33	13-@05#	24-1												
TST34	13-@66#	24-1												
TST35	13-A26#	24-1												
TST36	13-B08#	24-1												
TST37	13-B76#	24-1												
TST4	13-137#	24-1												
TST40	13-C78#	24-1												
TST41	13-D31#	24-1												
TST42	13-D84#	24-1												
TST43	13-E64#	24-1												
TST44	13-E99#	24-1												
TST45	13-G20#	24-1												
TST46	13-H70#	24-1												
TST47	13-I45#	24-1												
TST5	13-159#	24-1												
TST50	13-J84#	24-1												
TST51	13-K30#	24-1												
TST52	13-K64#	24-1												
TST53	13-L96#	24-1												
TST54	13-M56#	24-1												
TST55	13-M92#	24-1												
TST56	13-N40#	24-1												
TST57	13-N79#	24-1												
TST6	13-329#	24-1												
TST60	13-0C9#	24-1												
TST61	13-048#	24-1												
TST62	13-074#	24-1												
TST63	13-P18#	24-1												
TST64	13-P39#	24-1												
TST65	13-P66#	24-1												
TST66	13-Q10#	24-1												
TST67	13-Q87#	24-1												
TST7	13-521#	24-1												
TST70	13-R35#	24-1												
TST71	13-R85#	24-1												
TST72	13-S22#	24-1												
TST73	13-S72#	24-1												
TST74	13-T29#	24-1												
TST75	13-T47#	24-1												
TST76	13-T70#	24-1												
TST77	13-T99#	24-1												
TSTNMB	26-44*	26-45*	26-47	26-159#										
TSTQUE	7-0#	11-8	11-9*	11-50	11-58	13-1	13-73	13-106	13-137	13-159	13-329	13-521	13-546	13-557
	13-610	13-677	13-781	13-794	13-864	13-973	13-996	13-:49	13-:77	13-:97	13-;41	13-<03	13- 02	13-=19
	13-=77	13->42	13-?03	13-@05	13-@66	13-A26	13-B08	13-B76	13-C78	13-D31	13-D84	13-E64	13-E99	13-G20
	13-H70	13-I45	13-J84	13-K30	13-K64	13-L96	13-M56	13-M92	13-N40	13-N79	13-0C9	13-048	13-074	13-P18
	13-P39	13-P66	13-Q10	13-Q87	13-R35	13-R85	13-S22	13-S72	13-T29	13-T47	13-T70	13-T99	13-U28	13-U58
	13-U95	13-V33	13-V53	13-V86	13-W07	13-W56	13-X01	13-X71	13-Y48	13-\10	13-_01	13-c07	13-c75	13-g77
	13-k51	14-11	14-13*	14-17	14-17*	18-11								
TYPBN	26-149	29-1#												
TYPDS	14-20	14-20	26-143	29-1#										
TYPE	9-6	9-28	9-42	9-48	9-53	9-59	9-60	9-77	9-81	9-83	9-108	9-114	9-117	9-123
	9-130	9-133	9-134	9-135	9-142	10-11	10-16	10-17	10-19	10-20	10-28	10-32	10-34	10-40
	10-44	10-48	10-54	10-61	10-62	10-64	10-69	10-75	10-86	10-87	10-92	10-101	11-3	11-6

	11-7	11-14	11-25	11-30	11-67	11-68	14-20	14-20	14-20	20-1	21-1	22-1	23-1	25-1
	25-1	26-20	26-21	26-39	26-40	26-46	26-51	26-53	26-61	26-104	26-110	26-114	26-126	26-132
	26-134	26-152	26-154	27-1	27-1	27-1	27-1	27-1	27-1	27-1	27-1	27-1	27-1	27-1
	27-1	27-1	27-1	27-1	27-1	27-1	27-1	29-1#	30-1					
TYPOC	9-8	10-33	10-47	26-146	27-1	29-1#								
TYPON	29-1#													
TYPOS	9-45	9-56	9-82	11-16	11-69	26-22	26-47	26-52	26-54	29-1#				
U0	4-945#													
U1	4-945#													
U2	4-945#													
UNS	4-765#	13-150	13-767	13-769	13-?21	13-?26	13-?31	13-?56	13-?59	13-?64	13-L42			
UNTMSK	4-949#	13-12	13-33	13-:01	13-V06									
UNTOFF	9-123	32-28#												
UNTON	9-133	32-29#												
UPE	4-932#													
USE	4-888#													
VV	4-759#	13->10	13-K79	13-L00	13-L68	13-M10	13-M28	13-M31	16-21	16-25				
WATCH	7-0#	13-K49*	13-K50	13-P84*	13-P90	13-Y15*	13-Y19	13-c59*	13-c63	13-e84*	13-e88	13-f56*	13-f60	13-g94*
	13-g98	13-h38*	13-h42	13-i24*	13-i28	13-i86*	13-i90	13-j72*	13-j76	13-l05*	13-l09	15-73*	15-75*	
WC	4-829#													
WCD	4-715#	13-F71	13-L14	13-Q03	13-Q76									
WCE	4-931#	33-77	33-78											
WCEHI	4-956#													
WCELO	4-957#													
WCF	4-774#	4-781	13-695	13-724										
WCH	4-716#	13-F79	13-Q67											
WD	4-719#	13-F87	13-Q73	13-S41	13-X50	13-X54	13-Y06							
WH	4-720#	13-F95	13-Q64											
WLE	4-768#	4-781	13-X28	13-X52	33-81	33-82								
WRL	4-754#	13->53	13->63	13->65	13->80	13->84								
XNUDC	4-858#	13-97	13-191	13-224	13-239	13-290	13-293	13-376	13-437	13-459	13-509	13-:90	13-;52	13-;62
	13-;72	13-;90	13-N68	13-041										
XNUER2	4-883#	13-125	13-192	13-226	13-240	13-300	13-303	13-398	13-420	13-475	13-512	13-882	13-931	13-939
	13-940	13-963	13-E49											
XNUOF	4-853#	13-81	13-99	13-190	13-223	13-238	13-284	13-287	13-395	13-417	13-456	13-507	13-808	13-816
	13-829	13-837	13-838	13-854	13-N71	13-003								
XSIZ	9-71#	10-71	10-102											
XXDP	7-0#	9-33*	9-36*	9-37	9-39*	9-44	9-55	9-126	9-128					
Y	32-31#													
ZEROS	35-38#													

CROSS REFERENCE TABLE (CREF V01-05)

	13-h39	13-h91	13-i25	13-i87	13-j44	13-j73	13-k09	13-k38	13-l06					
GETMR2	4-256#	13-F15	13-F41	13-T80	13-T91	13-U09	13-U20	13-U38	13-U49	13-U77	13-[68	13-^51	13-b48	13-e47
	13-f99	13-l22	13-l57	13-m14										
GETOF	4-264#	13-88	13-183	13-231	13-271	13-391	13-413	13-452	13-504	13-785	13-804	13-825	13-852	13-B19
	13-B98	13-C99	13-D51	13-N70	13-002	13-P09	13-S36							
GETPRI	4-682#	15-58												
GETSN	4-216#	13-981	13-985											
GETSWR	4-682#	9-28	9-28#											
GETWC	4-128#													
GETX	4-112#													
MSG	4-8#	13-1	13-73	13-106	13-137	13-159	13-324#	13-329	13-518#	13-521	13-546	13-557	13-610	13-677
	13-781	13-794	13-864	13-973	13-996	13-:49	13-:77	13-:97	13-;41	13-<03	13-=02	13-=19	13-=77	13->42
	13-?03	13-@05	13-@66	13-A26	13-A95#	13-B08	13-B73#	13-B76	13-C78	13-D31	13-D84	13-E64	13-E99	13-G20
	13-H70	13-I45	13-J84	13-K30	13-K64	13-L96	13-M56	13-M92	13-N40	13-N79	13-009	13-048	13-074	13-P18
	13-P39	13-P66	13-Q10	13-Q87	13-R35	13-R85	13-S22	13-S72	13-T29	13-T47	13-T70	13-T99	13-U28	13-U58
	13-U95	13-V33	13-V53	13-V86	13-W07	13-W56	13-X01	13-X71	13-Y48	13-\10	13-_01	13-c07	13-c75	13-g77
	13-k51													
MULT	4-682#													
NEWTST	4-682#	13-1	13-73	13-106	13-137	13-159	13-329	13-521	13-546	13-557	13-610	13-677	13-781	13-794
	13-864	13-973	13-996	13-:49	13-:77	13-:97	13-;41	13-<03	13-=02	13-=19	13-=77	13->42	13-?03	13-@05
	13-@66	13-A26	13-B08	13-B76	13-C78	13-D31	13-D84	13-E64	13-E99	13-G20	13-H70	13-I45	13-J84	13-K30
	13-K64	13-L96	13-M56	13-M92	13-N40	13-N79	13-009	13-048	13-074	13-P18	13-P39	13-P66	13-Q10	13-Q87
	13-R35	13-R85	13-S22	13-S72	13-T29	13-T47	13-T70	13-T99	13-U28	13-U58	13-U95	13-V33	13-V53	13-V86
	13-W07	13-W56	13-X01	13-X71	13-Y48	13-\10	13-_01	13-c07	13-c75	13-g77	13-k51			
NWTST	4-498#	13-1	13-73	13-106	13-137	13-159	13-329	13-521	13-546	13-557	13-610	13-677	13-781	13-794
	13-864	13-973	13-996	13-:49	13-:77	13-:97	13-;41	13-<03	13-=02	13-=19	13-=77	13->42	13-?03	13-@05
	13-@66	13-A26	13-B08	13-B76	13-C78	13-D31	13-D84	13-E64	13-E99	13-G20	13-H70	13-I45	13-J84	13-K30
	13-K64	13-L96	13-M56	13-M92	13-N40	13-N79	13-009	13-048	13-074	13-P18	13-P39	13-P66	13-Q10	13-Q87
	13-R35	13-R85	13-S22	13-S72	13-T29	13-T47	13-T70	13-T99	13-U28	13-U58	13-U95	13-V33	13-V53	13-V86
	13-W07	13-W56	13-X01	13-X71	13-Y48	13-\10	13-_01	13-c07	13-c75	13-g77	13-k51			
POP	4-682#	13-<33	13-<34	13-<38	13-<39	13-<98	13-<99	15-45	15-46	18-12	19-1	21-1	28-1	30-1
	30-1	31-1	31-1											
PUSH	4-682#	13-<24	13-<25	13-<67	13-<68	15-6	15-7	18-8	19-1	21-1	28-1	30-1	30-1	31-1
	31-1	31-1												
PUTAS	4-460#	13-387	13-U14	13-V09	13-V60	13-V74	13-V93	13-W16						
PUTBA	4-324#	13-S40	13-k96											
PUTBAE	4-468#													
PUTCS1	4-309#	13-78	13-111	13-165	13-172	13-221	13-430	13-617	13-630	13-631	13-643	13-644	13-663	13-664
	13-_08	13-C00	13-C41	13-D03	13-D67	13-E80	13-F29	13-G37	13-H88	13-I72	13-K09	13-K48	13-K72	13-K93
	13-L14	13-L83	13-M20	13-M68	13-N06	13-N55	13-N94	13-023	13-060	13-093	13-P56	13-P85	13-Q28	13-R05
	13-R52	13-R97	13-S41	13-S87	13-T77	13-U06	13-U35	13-U68	13-V42	13-W68	13-X17	13-Y06	13-Y58	13-Y96
	13-Z34	13-Z65	13-[07	13-[64	13-\25	13-\66	13-]07	13-]41	13-]86	13-^47	13-_16	13-_57	13-`01	13-`36
	13-`97	13-a57	13-b29	13-c19	13-c89	13-d29	13-e21	13-e81	13-f53	13-g91	13-h35	13-i21	13-i83	13-j69
	13-l02	16-19	17-17											
PUTCS2	4-340#	13-:24												
PUTDA	4-364#	13-79	13-166	13-173	13-216	13-222	13-263	13-384	13-406	13-467	13-484	13-503	13-568	13-569
	13-579	13-580	13-595	13-596	13-:82	13-:84	13-;03	13-;04	13-;14	13-;15	13-;28	13-;29	13-B20	13-B96
	13-C38	13-D00	13-D49	13-N50	13-019	13-P52	13-Q19	13-Q97	13-R47	13-R93	13-S38	13-U43	13-Y03	13-\21
	13-\62	13-]03	13-]37	13-]82	13-^43	13-_12	13-_53	13-`97	13-`32	13-`93	13-a53	13-b25	13-c18	13-c87
	13-d27	13-e19	13-e79	13-f51	13-g89	13-h34	13-i12	13-i74	13-j67	13-k87				
PUTDB	4-332#													
PUTDC	4-372#	13-80	13-176	13-218	13-224	13-265	13-369	13-429	13-447	13-485	13-499	13-:83	13-:85	13-;50
	13-;59	13-;60	13-;69	13-;70	13-;87	13-;88	13-C89	13-D48	13-N51	13-020	13-056	13-P30	13-Q20	13-Q98
	13-R46	13-R92	13-S37	13-Y04	13-\22	13-\63	13-]04	13-]38	13-]83	13-^44	13-_13	13-_54	13-`98	13-`33
	13-`94	13-a54	13-b26	13-c17	13-c88	13-d28	13-e20	13-e80	13-f52	13-g90	13-h33	13-i11	13-i75	13-j68
	13-k88													
PUTDS	4-348#	13-469	13-501											